# Repository Documentation

This document provides a comprehensive overview of the repository's structure and contents.The first section, titled 'Directory/File Tree', displays the repository's hierarchy in a tree format.In this section, directories and files are listed using tree branches to indicate their structure and relationships.Following the tree representation, the 'File Content' section details the contents of each file in the repository.Each file's content is introduced with a '[File Begins]' marker followed by the file's relative path,and the content is displayed verbatim. The end of each file's content is marked with a '[File Ends]' marker.This format ensures a clear and orderly presentation of both the structure and the detailed contents of the repository.

## Directory/File Tree Begins -->

sem4pi-22-23-61-master/

└── sem4pi-22-23-61-master

├── base.app.backoffice.console

│ ├── pom.xml

│ └── src

│ ├── main

│ │ ├── java

│ │ │ └── eapli

│ │ │ └── base

│ │ │ └── app

│ │ │ └── backoffice

│ │ │ └── console

│ │ │ ├── BaseBackoffice.java

│ │ │ └── presentation

│ │ │ ├── Classe

│ │ │ │ ├── ScheduleClassAction.java

│ │ │ │ ├── ScheduleClassUI.java

│ │ │ │ ├── ScheduleExtraClassAction.java

│ │ │ │ ├── ScheduleExtraClassUI.java

│ │ │ │ ├── UpdateClassAction.java

│ │ │ │ └── UpdateClassScheduleUI.java

│ │ │ ├── MainMenu.java

│ │ │ ├── SharedBoard

│ │ │ │ ├── CreateBoardPostItAction.java

│ │ │ │ ├── CreateBoardPostItUI.java

│ │ │ │ ├── SharedBoardAction.java

│ │ │ │ ├── SharedBoardArchiveAction.java

│ │ │ │ ├── SharedBoardArchiveUI.java

│ │ │ │ ├── SharedBoardListAction.java

│ │ │ │ ├── SharedBoardListUI.java

│ │ │ │ ├── SharedBoardPrinter.java

│ │ │ │ ├── SharedBoardUI.java

│ │ │ │ ├── UndoPostItAction.java

│ │ │ │ ├── UndoPostItUI.java

│ │ │ │ ├── UpdateBoardPostItAction.java

│ │ │ │ └── UpdateBoardPostItUI.java

│ │ │ ├── authz

│ │ │ │ ├── AddUserAction.java

│ │ │ │ ├── AddUserUI.java

│ │ │ │ ├── DeactivateUserAction.java

│ │ │ │ ├── DeactivateUserUI.java

│ │ │ │ ├── ListUsersAction.java

│ │ │ │ ├── ListUsersUI.java

│ │ │ │ └── SystemUserPrinter.java

│ │ │ ├── clientuser

│ │ │ │ ├── AcceptRefuseSignupRequestAction.java

│ │ │ │ ├── AcceptRefuseSignupRequestUI.java

│ │ │ │ └── SignupRequestPrinter.java

│ │ │ ├── course

│ │ │ │ ├── CoursePrinter.java

│ │ │ │ ├── CreateCourseAction.java

│ │ │ │ ├── CreateCourseUI.java

│ │ │ │ ├── OpenCloseCourseAction.java

│ │ │ │ ├── OpenCloseCourseUI.java

│ │ │ │ ├── OpenCloseEnrollementsAction.java

│ │ │ │ ├── OpenCloseEnrollementsUI.java

│ │ │ │ └── TeacherPrinter.java

│ │ │ ├── enrollment

│ │ │ │ ├── ApproveEnrollmentAction.java

│ │ │ │ ├── ApproveEnrollmentUI.java

│ │ │ │ ├── BulkEnrollmentUI.java

│ │ │ │ ├── EnrollmentPrinter.java

│ │ │ │ ├── RequestEnrollmentAction.java

│ │ │ │ └── RequestEnrollmentUI.java

│ │ │ ├── exam

│ │ │ │ ├── CreateExamAction.java

│ │ │ │ ├── CreateExamUI.java

│ │ │ │ ├── ListCourseExamsAction.java

│ │ │ │ ├── ListCourseExamsUI.java

│ │ │ │ ├── ListFutureExamsAction.java

│ │ │ │ └── ListFutureExamsUI.java

│ │ │ └── meeting

│ │ │ ├── AcceptRefuseInvitesAction.java

│ │ │ ├── AcceptRefuseInvitesUI.java

│ │ │ ├── CancelMeetingAction.java

│ │ │ ├── CancelMeetingUI.java

│ │ │ ├── InvitePrinter.java

│ │ │ ├── ListParticipantAction.java

│ │ │ ├── ListParticipantUI.java

│ │ │ ├── MeetingPrinter.java

│ │ │ ├── ScheduleMeetingAction.java

│ │ │ └── ScheduleMeetingUI.java

│ │ └── resources

│ │ ├── application.properties

│ │ └── logback.xml

│ └── test

│ └── java

│ └── eapli

│ └── base

│ └── app

│ └── other

│ └── console

│ └── BaseBackofficeTest.java

├── base.app.bootstrap

│ ├── pom.xml

│ └── src

│ └── main

│ ├── java

│ │ └── eapli

│ │ └── base

│ │ └── app

│ │ └── bootstrap

│ │ └── BaseBootstrap.java

│ └── resources

│ ├── application.properties

│ └── logback.xml

├── base.app.common.console

│ ├── pom.xml

│ └── src

│ ├── main

│ │ └── java

│ │ └── eapli

│ │ └── base

│ │ └── app

│ │ └── common

│ │ └── console

│ │ ├── BaseApplication.java

│ │ └── presentation

│ │ └── authz

│ │ ├── ChangePasswordUI.java

│ │ ├── LoginAction.java

│ │ ├── LoginUI.java

│ │ ├── LogoutAction.java

│ │ ├── LogoutUI.java

│ │ └── MyUserMenu.java

│ └── test

│ └── java

│ └── eapli

│ └── base

│ └── consoleapp

│ └── common

│ └── AppTest.java

├── base.app.other.console

│ ├── pom.xml

│ └── src

│ ├── main

│ │ ├── java

│ │ │ └── eapli

│ │ │ └── base

│ │ │ └── app

│ │ │ └── other

│ │ │ └── console

│ │ │ ├── OtherApp.java

│ │ │ └── presentation

│ │ │ └── MainMenu.java

│ │ └── resources

│ │ ├── application.properties

│ │ └── logback.xml

│ └── test

│ └── java

│ └── eapli

│ └── base

│ └── app

│ └── other

│ └── console

│ └── OtherAppTest.java

├── base.app.user.console

│ ├── pom.xml

│ └── src

│ ├── main

│ │ ├── java

│ │ │ └── eapli

│ │ │ └── base

│ │ │ └── app

│ │ │ └── user

│ │ │ └── console

│ │ │ ├── BaseUserApp.java

│ │ │ └── presentation

│ │ │ ├── ClientUserBaseUI.java

│ │ │ ├── FrontMenu.java

│ │ │ ├── MainMenu.java

│ │ │ └── myuser

│ │ │ ├── SignupRequestAction.java

│ │ │ ├── SignupRequestUI.java

│ │ │ └── UserDataWidget.java

│ │ └── resources

│ │ ├── application.properties

│ │ └── logback.xml

│ └── test

│ └── java

│ └── eapli

│ └── base

│ └── utente

│ └── consoleapp

│ └── AppTest.java

├── base.bootstrappers

│ ├── pom.xml

│ └── src

│ ├── main

│ │ ├── java

│ │ │ └── eapli

│ │ │ └── base

│ │ │ └── infrastructure

│ │ │ ├── bootstrapers

│ │ │ │ ├── BaseBootstrapper.java

│ │ │ │ ├── MasterUsersBootstrapper.java

│ │ │ │ ├── TestDataConstants.java

│ │ │ │ ├── UsersBootstrapperBase.java

│ │ │ │ └── demo

│ │ │ │ ├── BackofficeUsersBootstrapper.java

│ │ │ │ ├── BaseDemoBootstrapper.java

│ │ │ │ └── ClientUserBootstrapper.java

│ │ │ └── smoketests

│ │ │ └── BaseDemoSmokeTester.java

│ │ └── resources

│ │ └── application.properties

│ └── test

│ └── java

│ └── eapli

│ └── base

│ └── bootstrapapp

│ └── AppTest.java

├── base.core

│ ├── machinet.conf

│ ├── pom.xml

│ └── src

│ ├── main

│ │ ├── java

│ │ │ ├── META-INF

│ │ │ │ └── application.xml

│ │ │ └── eapli

│ │ │ └── base

│ │ │ ├── ANTLR

│ │ │ │ ├── ExamValidation

│ │ │ │ │ ├── Exam.txt

│ │ │ │ │ ├── ExamFile.txt

│ │ │ │ │ ├── Exame.g4

│ │ │ │ │ ├── Exame.interp

│ │ │ │ │ ├── Exame.tokens

│ │ │ │ │ ├── ExameBaseListener.txt

│ │ │ │ │ ├── ExameBaseVisitor.txt

│ │ │ │ │ ├── ExameLexer.interp

│ │ │ │ │ ├── ExameLexer.txt

│ │ │ │ │ ├── ExameListener.txt

│ │ │ │ │ ├── ExameParser.txt

│ │ │ │ │ └── ExameVisitor.txt

│ │ │ │ ├── FormativeExamValidation

│ │ │ │ │ ├── ExameFormativo.g4

│ │ │ │ │ └── ExameFormativoFile.txt

│ │ │ │ └── QuestionValidation

│ │ │ │ ├── Question.g4

│ │ │ │ └── QuestionFile.txt

│ │ │ ├── AppSettings.java

│ │ │ ├── Application.java

│ │ │ ├── Classe

│ │ │ │ ├── aplication

│ │ │ │ │ ├── ClassController.java

│ │ │ │ │ ├── ClasseDTO.java

│ │ │ │ │ └── ScheduleClassService.java

│ │ │ │ ├── domain

│ │ │ │ │ ├── Classe.java

│ │ │ │ │ ├── Classe\_Finish\_Date.java

│ │ │ │ │ ├── Classe\_Finish\_Time.java

│ │ │ │ │ ├── Classe\_Start\_Date.java

│ │ │ │ │ ├── Classe\_Start\_Time.java

│ │ │ │ │ ├── Classe\_Title.java

│ │ │ │ │ └── DayOfWeek.java

│ │ │ │ └── repository

│ │ │ │ └── ClassRepository.java

│ │ │ ├── Course

│ │ │ │ ├── Application

│ │ │ │ │ ├── CourseController.java

│ │ │ │ │ ├── CourseService.java

│ │ │ │ │ ├── CreateCourseController.java

│ │ │ │ │ ├── ListCoursesController.java

│ │ │ │ │ ├── ListCoursesService.java

│ │ │ │ │ ├── OpenCloseCourseController.java

│ │ │ │ │ └── OpenCloseEnrollementsController.java

│ │ │ │ ├── Domain

│ │ │ │ │ ├── Course.java

│ │ │ │ │ ├── Course\_ID.java

│ │ │ │ │ ├── Course\_Name.java

│ │ │ │ │ ├── Maximum\_Number\_Of\_Students.java

│ │ │ │ │ ├── Minimum\_Number\_Of\_Students.java

│ │ │ │ │ └── Small\_Textual\_Description.java

│ │ │ │ └── Repository

│ │ │ │ └── CourseRepository.java

│ │ │ ├── Enrollment

│ │ │ │ ├── Application

│ │ │ │ │ ├── ApproveEnrollmentController.java

│ │ │ │ │ ├── BulkEnrollmentController.java

│ │ │ │ │ ├── ListEnrollmentsController.java

│ │ │ │ │ ├── ListEnrollmentsService.java

│ │ │ │ │ └── RequestEnrollmentController.java

│ │ │ │ ├── Domain

│ │ │ │ │ ├── BulkEnrollment

│ │ │ │ │ │ └── BulkEnrollment.java

│ │ │ │ │ ├── Enrollment.java

│ │ │ │ │ ├── EnrollmentDescription.java

│ │ │ │ │ ├── EnrollmentID.java

│ │ │ │ │ └── EnrollmentState.java

│ │ │ │ └── Repository

│ │ │ │ └── EnrollmentRepository.java

│ │ │ ├── ExtraClasse

│ │ │ │ ├── aplication

│ │ │ │ │ ├── ExtraClassController.java

│ │ │ │ │ ├── ExtraClasseDTO.java

│ │ │ │ │ └── ScheduleExtraClasseService.java

│ │ │ │ ├── domain

│ │ │ │ │ ├── ExtraClasse.java

│ │ │ │ │ ├── ExtraClasse\_Day.java

│ │ │ │ │ ├── ExtraClasse\_Finish\_Time.java

│ │ │ │ │ ├── ExtraClasse\_Start\_Time.java

│ │ │ │ │ └── ExtraClasse\_Title.java

│ │ │ │ └── repository

│ │ │ │ └── ExtraClasseRepository.java

│ │ │ ├── Manager

│ │ │ │ └── Manager.java

│ │ │ ├── SharedBoard

│ │ │ │ ├── aplication

│ │ │ │ │ ├── BoardUpdateService.java

│ │ │ │ │ ├── CreateBoardPostItController.java

│ │ │ │ │ ├── CreateBoardPostItService.java

│ │ │ │ │ ├── SharedBoardArchiveController.java

│ │ │ │ │ ├── SharedBoardArchiveService.java

│ │ │ │ │ ├── SharedBoardController.java

│ │ │ │ │ ├── SharedBoardService.java

│ │ │ │ │ ├── UpdateBoardPostItController.java

│ │ │ │ │ └── UpdateBoardPostItService.java

│ │ │ │ ├── domain

│ │ │ │ │ ├── Board.java

│ │ │ │ │ ├── CreateBoardPostItThread.java

│ │ │ │ │ ├── MoveBoardPostItThread.java

│ │ │ │ │ ├── NumberofColumns.java

│ │ │ │ │ ├── NumberofRows.java

│ │ │ │ │ ├── PostIt.java

│ │ │ │ │ ├── SharedBoard.java

│ │ │ │ │ ├── SharedBoardThread.java

│ │ │ │ │ ├── Shared\_Board\_Title.java

│ │ │ │ │ ├── UpdateBoardPostItThread.java

│ │ │ │ │ └── sharedBoardHistory.java

│ │ │ │ └── repository

│ │ │ │ └── SharedBoardRepository.java

│ │ │ ├── Student\_Teacher

│ │ │ │ ├── Date\_Of\_Birth.java

│ │ │ │ ├── Student

│ │ │ │ │ ├── Repository

│ │ │ │ │ │ └── StudentRepository.java

│ │ │ │ │ └── domain

│ │ │ │ │ ├── MechanographicNumber.java

│ │ │ │ │ └── Student.java

│ │ │ │ ├── Tax\_Payer\_Number.java

│ │ │ │ └── Teacher

│ │ │ │ ├── Domain

│ │ │ │ │ ├── Acronym.java

│ │ │ │ │ └── Teacher.java

│ │ │ │ └── Repository

│ │ │ │ └── TeacherRepository.java

│ │ │ ├── clientusermanagement

│ │ │ │ ├── application

│ │ │ │ │ ├── AcceptRefuseSignupFactory.java

│ │ │ │ │ ├── AcceptRefuseSignupRequestController.java

│ │ │ │ │ ├── AcceptRefuseSignupRequestControllerEventfullImpl.java

│ │ │ │ │ ├── AcceptRefuseSignupRequestControllerTxImpl.java

│ │ │ │ │ ├── ClientUserService.java

│ │ │ │ │ ├── ListClientUsersController.java

│ │ │ │ │ └── eventhandlers

│ │ │ │ │ ├── AddClientUserOnSignupAcceptedController.java

│ │ │ │ │ └── NewUserRegisteredFromSignupWatchDog.java

│ │ │ │ ├── domain

│ │ │ │ │ ├── ApprovalStatus.java

│ │ │ │ │ ├── ClientUser.java

│ │ │ │ │ ├── ClientUserBuilder.java

│ │ │ │ │ ├── SignupRequest.java

│ │ │ │ │ ├── SignupRequestBuilder.java

│ │ │ │ │ └── events

│ │ │ │ │ ├── NewUserRegisteredFromSignupEvent.java

│ │ │ │ │ └── SignupAcceptedEvent.java

│ │ │ │ └── repositories

│ │ │ │ ├── ClientUserRepository.java

│ │ │ │ └── SignupRequestRepository.java

│ │ │ ├── exammanagement

│ │ │ │ ├── application

│ │ │ │ │ ├── CreateExamController.java

│ │ │ │ │ ├── CreateExamService.java

│ │ │ │ │ ├── ListCourseExamsController.java

│ │ │ │ │ ├── ListCourseExamsService.java

│ │ │ │ │ ├── ListFutureExamsController.java

│ │ │ │ │ └── ListFutureExamsService.java

│ │ │ │ ├── domain

│ │ │ │ │ ├── Exam.java

│ │ │ │ │ ├── ExamDate.java

│ │ │ │ │ ├── ExamResult.java

│ │ │ │ │ ├── ExamTime.java

│ │ │ │ │ └── ExamTitle.java

│ │ │ │ └── repository

│ │ │ │ └── ExamRepository.java

│ │ │ ├── infrastructure

│ │ │ │ └── persistence

│ │ │ │ ├── PersistenceContext.java

│ │ │ │ └── RepositoryFactory.java

│ │ │ ├── meetingmanagement

│ │ │ │ ├── application

│ │ │ │ │ ├── AcceptRefuseInvitesController.java

│ │ │ │ │ ├── AcceptRefuseInvitesService.java

│ │ │ │ │ ├── CancelMeetingController.java

│ │ │ │ │ ├── CancelMeetingService.java

│ │ │ │ │ ├── ListParticipantController.java

│ │ │ │ │ ├── ListParticipantService.java

│ │ │ │ │ └── ScheduleMeetingController.java

│ │ │ │ ├── domain

│ │ │ │ │ ├── Invite.java

│ │ │ │ │ ├── InviteState.java

│ │ │ │ │ ├── Meeting.java

│ │ │ │ │ ├── MeetingDate.java

│ │ │ │ │ ├── MeetingDuration.java

│ │ │ │ │ ├── MeetingStatus.java

│ │ │ │ │ ├── MeetingTime.java

│ │ │ │ │ └── MeetingTitle.java

│ │ │ │ └── repository

│ │ │ │ ├── InviteRepository.java

│ │ │ │ └── MeetingRepository.java

│ │ │ ├── myclientuser

│ │ │ │ └── application

│ │ │ │ ├── MyClientUserService.java

│ │ │ │ └── SignupController.java

│ │ │ ├── systemUserManagement

│ │ │ │ └── SystemUserRepository.java

│ │ │ └── usermanagement

│ │ │ ├── application

│ │ │ │ ├── AddUserController.java

│ │ │ │ ├── DeactivateUserController.java

│ │ │ │ ├── ListUsersController.java

│ │ │ │ └── eventhandlers

│ │ │ │ ├── AddUserOnSignupAcceptedController.java

│ │ │ │ └── SignupAcceptedWatchDog.java

│ │ │ └── domain

│ │ │ ├── BasePasswordPolicy.java

│ │ │ ├── BaseRoles.java

│ │ │ └── UserBuilderHelper.java

│ │ └── resources

│ │ └── ecafeteria.sample.properties

│ └── test

│ ├── eapli

│ │ └── base

│ │ ├── Classe

│ │ │ └── Domain

│ │ │ └── ClasseTest.java

│ │ ├── SharedBoard

│ │ │ ├── Domain

│ │ │ │ └── SharedBoardTest.java

│ │ │ └── aplication

│ │ │ └── SharedBoardServiceTest.java

│ │ ├── exammanagement

│ │ │ └── domain

│ │ │ ├── CreateExamControllerTest.java

│ │ │ └── ExamTest.java

│ │ ├── meetingmanagement

│ │ │ ├── aplication

│ │ │ │ └── CancelMeetingTest.java

│ │ │ └── domain

│ │ │ ├── InviteTest.java

│ │ │ └── MeetingTest.java

│ │ └── sharedboardmanagement

│ │ └── domain

│ │ ├── CreatePostItTest.java

│ │ └── UpdatePostItTest.java

│ └── java

│ └── eapli

│ └── base

│ └── clientusermanagement

│ └── domain

│ └── ClientUserTest.java

├── base.infrastructure.application

│ ├── pom.xml

│ └── src

│ └── main

│ ├── java

│ │ └── eapli

│ │ └── base

│ │ ├── AppSettings.java

│ │ ├── Application.java

│ │ └── infrastructure

│ │ └── authz

│ │ ├── AuthenticationCredentialHandler.java

│ │ └── CredentialHandler.java

│ └── resources

│ └── application.properties.sample

├── base.persistence.impl

│ ├── pom.xml

│ └── src

│ └── main

│ ├── java

│ │ └── eapli

│ │ └── base

│ │ └── persistence

│ │ └── impl

│ │ ├── inmemory

│ │ │ ├── InMemoryClientUserRepository.java

│ │ │ ├── InMemoryExamRepository.java

│ │ │ ├── InMemoryInitializer.java

│ │ │ ├── InMemoryInviteRepository.java

│ │ │ ├── InMemoryRepositoryFactory.java

│ │ │ ├── InMemorySharedBoardRepository.java

│ │ │ └── InMemorySignupRequestRepository.java

│ │ └── jpa

│ │ ├── BaseJpaReportingRepositoryBase.java

│ │ ├── BasepaRepositoryBase.java

│ │ ├── JpaClassRepository.java

│ │ ├── JpaClientUserRepository.java

│ │ ├── JpaCourseRepository.java

│ │ ├── JpaExamRepository.java

│ │ ├── JpaExtraClassRepository.java

│ │ ├── JpaInviteRepository.java

│ │ ├── JpaMeetingRepository.java

│ │ ├── JpaRepositoryFactory.java

│ │ ├── JpaSharedBoardRepository.java

│ │ ├── JpaSignupRequestRepository.java

│ │ ├── JpaStudentRepository.java

│ │ ├── JpaSystemUserRepository.java

│ │ ├── JpaTeacherRepository.java

│ │ └── PersistenceSettings.java

│ └── resources

│ └── META-INF

│ └── persistence.xml

├── base.rcomp

│ ├── Makefile

│ ├── SSL

│ │ ├── Makefile

│ │ ├── check-TLS

│ │ │ ├── CheckServerTLS.java

│ │ │ └── Makefile

│ │ ├── https-server-ajax-voting

│ │ │ ├── HTTPmessage.java

│ │ │ ├── HttpsServerAjaxVoting.java

│ │ │ ├── Makefile

│ │ │ ├── httpAjaxVotingRequest.java

│ │ │ ├── make\_cert

│ │ │ └── www

│ │ │ ├── index.html

│ │ │ └── rcomp-ajax.js

│ │ └── tcp-cli-srv

│ │ ├── Makefile

│ │ ├── TcpCliSumTLS.java

│ │ ├── TcpSrvSumTLS.java

│ │ └── make\_certs

│ ├── http-server-ajax-voting

│ │ ├── DemoConsumer.java

│ │ ├── HttpAjaxVotingRequest.java

│ │ ├── HttpServerAjaxVoting.java

│ │ ├── Httpmessage.java

│ │ ├── Makefile

│ │ └── www

│ │ ├── index.html

│ │ └── rcomp-ajax.js

│ ├── http-server-chat

│ │ ├── HttpChatConsumer.java

│ │ ├── HttpChatRequest.java

│ │ ├── HttpServerChat.java

│ │ ├── Makefile

│ │ ├── README

│ │ ├── httpmessage.java

│ │ └── www

│ │ ├── index.html

│ │ └── rcomp-chat.js

│ ├── http-server-form-file-upload

│ │ ├── HTTP.java

│ │ ├── HttpRequest.java

│ │ ├── HttpServerFormFileUpload.java

│ │ ├── Makefile

│ │ └── www

│ │ └── index.html

│ ├── tcp-chat

│ │ ├── Makefile

│ │ ├── TcpChatCli.java

│ │ ├── TcpChatCliGui.java

│ │ ├── TcpChatSrv.java

│ │ └── TcpChatSrvSingleThread.java

│ ├── tcp-cli-srv

│ │ ├── Makefile

│ │ ├── TcpCliSum.java

│ │ └── TcpSrvSum.java

│ ├── udp-chat

│ │ ├── Makefile

│ │ └── UdpChat.java

│ └── udp-cli-srv

│ ├── Makefile

│ ├── UdpCli.java

│ ├── UdpCliBcast.java

│ ├── UdpCliTo.java

│ ├── UdpSrv.java

│ └── UdpSrvMport.java

├── bitbucket-pipelines.yml

├── docs

│ ├── Divisao\_Tarefas

│ │ └── Divisao\_Tarefas

│ ├── Domain Model

│ │ └── US-G002

│ │ ├── domain-model.puml

│ │ └── readme.md

│ ├── Group\_Elements

│ │ ├── 1200614

│ │ │ └── readme.md

│ │ ├── 1200801

│ │ │ └── readme.md

│ │ ├── 1200874

│ │ │ └── readme.md

│ │ ├── 1201718

│ │ │ └── readme.md

│ │ └── 1210965

│ │ └── readme.md

│ ├── US1007

│ │ ├── US1007-ANALYSIS.md

│ │ ├── US1007-CD.puml

│ │ ├── US1007-DM.puml

│ │ ├── US1007-SD.puml

│ │ ├── US1007-SSD.puml

│ │ └── readme.md

│ ├── US1008

│ │ ├── US-1008-ANALYSIS.md

│ │ ├── US1008-CD.puml

│ │ ├── US1008-DM.puml

│ │ ├── US1008-SD.puml

│ │ ├── US1008-SSD.puml

│ │ └── readme.md

│ ├── US1009

│ │ ├── US1009-ANALYSIS.md

│ │ ├── US1009-CD.puml

│ │ ├── US1009-DM.puml

│ │ ├── US1009-SD.puml

│ │ ├── US1009-SSD.puml

│ │ └── readme.md

│ ├── US1010

│ │ ├── ANALYSIS.txt

│ │ ├── DESIGN.txt

│ │ ├── US1010 - TEXT PLAN.txt

│ │ ├── US1010-SD.puml

│ ├── US1011

│ │ ├── ANALYSIS.txt

│ │ ├── DESIGN.txt

│ │ ├── US1011- TEXT PLAN.txt

│ │ └── US1011-SD.puml

│ ├── US1012

│ │ ├── ANALYSIS.txt

│ │ ├── DESIGN.txt

│ │ └── US1012 -TEXT PLAN.txt

│ ├── US2001

│ │ ├── Analysis.txt

│ │ ├── US2001-SD.puml

│ ├── US2002

│ │ ├── US2002-ANALYSIS.md

│ │ ├── US2002-SD.puml

│ ├── US2003

│ │ ├── US2003-ANALYSIS.md

│ │ ├── US2003-SD.puml

│ ├── US3004

│ │ ├── US3004-ANALYSIS.md

│ │ ├── US3004-CD.puml

│ │ └── US3004-SD.puml

│ ├── US3006

│ │ ├── US3006(ONLY\_READ\_PERMISSION)-SD.puml

│ │ ├── US3006-ANALYSIS.md

│ │ └── US3006-SD(WRITE\_PERMISSION).puml

│ ├── US3007

│ │ ├── US3007-ANALYSIS.md

│ │ ├── US3007-SD(MOVE\_POST-IT).puml

│ │ ├── US3007-SD(UPDATE\_CREATED\_POST-IT).puml

│ ├── US3010

│ │ ├── US3010-ANALYSIS.md

│ │ ├── US3010-CD.puml

│ │ └── US3010-SD.puml

│ ├── US4001

│ │ ├── US4001-ANALYSIS.md

│ │ ├── US4001-SD.puml

│ ├── US4002

│ │ ├── US4002-ANALYSIS.md

│ │ ├── US4002-CD.puml

│ │ ├── US4002-DM.puml

│ │ └── US4002-SD.puml

│ ├── US4003

│ │ ├── US4003-ANALYSIS.md

│ │ ├── US4003-SD(ACCEPT\_INVITE).puml

│ │ ├── US4003-SD(REFUSE\_INVITE).puml

│ ├── US4004

│ │ ├── US4004-ANALYSIS.md

│ │ ├── US4004-SD.puml

│ ├── US5002

│ └── readme.md

├── lib

├── libs

│ ├── ModeloConceptual (1).drawio.html

├── license.txt

├── mvnw

├── mvnw.cmd

├── pom.xml

├── readme.md

## <-- Directory/File Tree Ends

## File Content Begins -->

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
   
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
   
 <artifactId>base.app.backoffice.console</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.app.backoffice.console</name>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 </properties>  
  
  
  
 <dependencies>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.app.common.console</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.persistence.impl</artifactId>  
 <version>${project.parent.version}</version>  
 <scope>runtime</scope>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\BaseBackoffice.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console;  
  
import eapli.base.app.backoffice.console.presentation.MainMenu;  
import eapli.base.app.common.console.BaseApplication;  
import eapli.base.app.common.console.presentation.authz.LoginUI;  
import eapli.base.clientusermanagement.application.eventhandlers.NewUserRegisteredFromSignupWatchDog;  
import eapli.base.clientusermanagement.domain.events.NewUserRegisteredFromSignupEvent;  
import eapli.base.clientusermanagement.domain.events.SignupAcceptedEvent;  
import eapli.base.infrastructure.authz.AuthenticationCredentialHandler;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.application.eventhandlers.SignupAcceptedWatchDog;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.pubsub.EventDispatcher;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra Sousa  
 \*/  
@SuppressWarnings("squid:S106")  
public final class BaseBackoffice extends BaseApplication {  
  
 /\*\*  
 \* avoid instantiation of this class.  
 \*/  
 private BaseBackoffice() {  
 }  
  
 /\*\*  
 \* @param args  
 \* the command line arguments  
 \*/  
 public static void main(final String[] args) {  
  
 AuthzRegistry.configure(PersistenceContext.repositories().users(),  
 new BasePasswordPolicy(), new PlainTextEncoder());  
  
 new BaseBackoffice().run(args);  
 }  
  
 @Override  
 protected void doMain(final String[] args) {  
 // login and go to main menu  
 if (new LoginUI(new AuthenticationCredentialHandler()).show()) {  
 // go to main menu  
 final MainMenu menu = new MainMenu();  
 menu.mainLoop();  
 }  
 }  
  
 @Override  
 protected String appTitle() {  
 return "Base Back Office";  
 }  
  
 @Override  
 protected String appGoodbye() {  
 return "Base Back Office";  
 }  
  
 @SuppressWarnings("unchecked")  
 @Override  
 protected void doSetupEventHandlers(final EventDispatcher dispatcher) {  
 dispatcher.subscribe(new NewUserRegisteredFromSignupWatchDog(),  
 NewUserRegisteredFromSignupEvent.class);  
 dispatcher.subscribe(new SignupAcceptedWatchDog(), SignupAcceptedEvent.class);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\BaseBackoffice.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleClassAction.java

package eapli.base.app.backoffice.console.presentation.Classe;  
  
import eapli.base.Classe.aplication.ClassController;  
import eapli.base.app.backoffice.console.presentation.exam.ListFutureExamsUI;  
import eapli.framework.actions.Action;  
  
public class ScheduleClassAction implements Action {  
  
  
  
 @Override  
 public boolean execute() {  
  
 return new ScheduleClassUI().show();  
  
 }  
}  
  
 /\*  
  
 private final ClassController classController;  
  
 public ScheduleClassAction(ClassController classController) {  
 this.classController = classController;  
 }  
  
 public boolean execute(){  
 try {  
 new ScheduleClassUI().run();  
 } catch (Exception e) {  
 throw new RuntimeException(e);  
 }  
 return true; // Return a boolean indicating if the operation was successful.  
 }  
  
  
 \*/

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleClassAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleClassUI.java

package eapli.base.app.backoffice.console.presentation.Classe;  
  
import eapli.base.Classe.aplication.ClassController;  
import eapli.base.Classe.domain.\*;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import java.time.LocalDate;  
import java.time.LocalTime;  
import java.util.InputMismatchException;  
import java.util.List;  
import java.util.Scanner;  
  
public class ScheduleClassUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final UserSession session = authz.session().orElseThrow(IllegalStateException::new);  
 private final SystemUser authenticatedUser = session.authenticatedUser();  
  
 private final ClassController classController = new ClassController(  
 AuthzRegistry.authorizationService(),  
 PersistenceContext.repositories().classRepository(),  
 PersistenceContext.repositories().teacherRepository());  
  
 public boolean show() {  
 try {  
 Scanner scanner = new Scanner(System.in);  
  
 System.out.println("Enter class title: ");  
 Classe\_Title title = Classe\_Title.valueOf(scanner.nextLine());  
  
 System.out.println("Enter class start time (HH:mm): ");  
 LocalTime startTime = LocalTime.parse(scanner.nextLine());  
 Classe\_Start\_Time userInputStartTime = new Classe\_Start\_Time(startTime);  
  
 System.out.println("Enter class finish time (HH:mm): ");  
 LocalTime endTime = LocalTime.parse(scanner.nextLine());  
 Classe\_Finish\_Time userInputEndTime = new Classe\_Finish\_Time(endTime);  
  
 System.out.println("Enter class start date (YYYY-MM-DD): ");  
 LocalDate startDate = LocalDate.parse(scanner.nextLine());  
 Classe\_Start\_Date start\_date = new Classe\_Start\_Date(startDate);  
  
 System.out.println("Enter class finish date (YYYY-MM-DD): ");  
 LocalDate finishDate = LocalDate.parse(scanner.nextLine());  
 Classe\_Finish\_Date finish\_date = new Classe\_Finish\_Date(finishDate);  
  
 System.out.println("Enter day of the week (1-7): ");  
 int dayOfWeek = scanner.nextInt();  
 DayOfWeek userInputDayOfWeek = DayOfWeek.valueOf(dayOfWeek);  
  
 scanner.nextLine(); // Consume the newline character  
  
 System.out.println("Enter teacher acronym: ");  
 Acronym teacherAcronym = Acronym.valueOf(scanner.nextLine());  
  
 TeacherRepository teacherRepository = PersistenceContext.repositories().teacherRepository();  
 List<Teacher> teacherOptional = teacherRepository.findTeacherByAcronym(teacherAcronym);  
 if (teacherOptional.isEmpty()) {  
 throw new IllegalArgumentException("No teacher found with the provided acronym");  
 }  
  
 boolean hasConflict = classController.checkClassConflict(new Classe(  
 title, start\_date, finish\_date, userInputStartTime, userInputEndTime, userInputDayOfWeek, teacherAcronym));  
 if (hasConflict) {  
 System.out.println("Class conflicts with an existing class.");  
 // Handle class conflict, display an error message or take any appropriate action  
 } else {  
 // Proceed with scheduling the class  
 classController.scheduleClass(  
 title, userInputStartTime, userInputEndTime, start\_date, finish\_date, userInputDayOfWeek, teacherAcronym);  
 System.out.println("Class scheduled successfully.");  
 }  
 } catch (InputMismatchException e) {  
 System.out.println("Invalid input. Please try again.");  
 } catch (IllegalArgumentException e) {  
 System.out.println(e.getMessage());  
 }  
 return false;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleClassUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleExtraClassAction.java

package eapli.base.app.backoffice.console.presentation.Classe;  
  
import eapli.base.Classe.aplication.ClassController;  
import eapli.base.ExtraClasse.aplication.ExtraClassController;  
import eapli.framework.actions.Action;  
  
public class ScheduleExtraClassAction implements Action {  
  
  
  
 @Override  
 public boolean execute() {  
 try {  
 return new ScheduleExtraClassUI().show();  
 } catch (Exception e) {  
 throw new RuntimeException(e);  
 }  
 }  
}  
  
 /\*  
 private final ExtraClassController extraClassController;  
  
 public ScheduleExtraClassAction(ExtraClassController extraClassController) {  
 this.extraClassController = extraClassController;  
 }  
  
 public boolean execute(){  
 try {  
 new ScheduleExtraClassUI(extraClassController).run();  
 } catch (Exception e) {  
 throw new RuntimeException(e);  
 }  
 return true; // Return a boolean indicating if the operation was successful.  
 }  
  
  
 \*/

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleExtraClassAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleExtraClassUI.java

package eapli.base.app.backoffice.console.presentation.Classe;  
  
import eapli.base.Classe.aplication.ClassController;  
import eapli.base.Classe.domain.\*;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Day;  
import eapli.base.ExtraClasse.aplication.ExtraClassController;  
import eapli.base.ExtraClasse.domain.ExtraClasse;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Finish\_Time;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Start\_Time;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Title;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import java.time.LocalDate;  
import java.time.LocalTime;  
import java.util.\*;  
  
public class ScheduleExtraClassUI {  
  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
  
 private final ExtraClassController extraClassController = new ExtraClassController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().extraClassRepostory(), PersistenceContext.repositories().teacherRepository()) ;  
  
 public boolean show() throws Exception {  
  
  
 Scanner scanner = new Scanner(System.in);  
 scanner.nextLine();  
  
  
 System.out.println("Enter class title: ");  
 ExtraClasse\_Title title = ExtraClasse\_Title.valueOf(scanner.nextLine());  
  
 System.out.println("Enter class start time (HH:mm): ");  
 ExtraClasse\_Start\_Time startTime = ExtraClasse\_Start\_Time.valueOf(LocalTime.parse(scanner.nextLine()));  
  
 System.out.println("Enter class finish time (HH:mm): ");  
 ExtraClasse\_Finish\_Time finishTime = ExtraClasse\_Finish\_Time.valueOf(LocalTime.parse(scanner.nextLine()));  
  
  
 System.out.println("Enter day of the week (1-7): ");  
 int dayOfWeek = scanner.nextInt();  
 ExtraClasse\_Day dayInput = (ExtraClasse\_Day) ExtraClasse\_Day.valueOf(dayOfWeek);  
  
 System.out.println("Enter teacher Acronym: ");  
 Acronym Acronym = eapli.base.Student\_Teacher.Teacher.Domain.Acronym.valueOf(scanner.nextLine());  
 Acronym teacherAcronym = Acronym.valueOf(scanner.nextLine());  
  
 List<Teacher> teacherOptional = extraClassController.findTeacherByAcronym(teacherAcronym);  
 if (teacherOptional.isEmpty()) {  
 throw new IllegalArgumentException("No teacher found with provided acronym");  
 }  
  
 boolean hasConflict = extraClassController.checkClassConflict(new ExtraClasse(title, startTime, finishTime,dayInput, teacherAcronym));  
 if (hasConflict) {  
 System.out.println("Class conflicts with an existing class.");  
  
 } else {  
  
 extraClassController.scheduleExtraClass( title, startTime, finishTime, dayInput, teacherAcronym);  
 System.out.println("Class scheduled successfully.");  
 }  
 return false;  
 }  
 }  
  
  
  
  
  
  
  
  
  
  
  
 /\*  
 private final ExtraClassController extraClassController;  
  
 public ScheduleExtraClassUI(ExtraClassController extraClassController) {  
 this.extraClassController = extraClassController;  
 }  
  
 public void run() throws Exception {  
 Scanner scanner = new Scanner(System.in);  
 System.out.println("Enter class ID: ");  
 Long id = scanner.nextLong();  
 scanner.nextLine();  
  
 System.out.println("Enter class title: ");  
 ExtraClasse\_Title title = ExtraClasse\_Title.from(scanner.nextLine());  
  
 System.out.println("Enter class start time (hh:mm): ");  
 String startTimeInput = scanner.nextLine();  
 ExtraClasse\_Start\_Time startTime = ExtraClasse\_Start\_Time.from(LocalTime.parse(startTimeInput));  
  
 System.out.println("Enter class finish time (hh:mm): ");  
 String finishTimeInput = scanner.nextLine();  
 ExtraClasse\_Finish\_Time finishTime = ExtraClasse\_Finish\_Time.from(LocalTime.parse(finishTimeInput));  
  
 System.out.println("Enter day of the week (1-7): ");  
 int dayOfWeek = scanner.nextInt();  
 DayOfWeek dayInput = DayOfWeek.valueOf(dayOfWeek);  
  
  
 System.out.println("Enter teacher acronym: ");  
 String teacherAcronymInput = scanner.nextLine();  
 Acronym teacherAcronym = Acronym.valueOf(teacherAcronymInput);  
  
 boolean hasConflict = extraClassController.checkClassConflict(new ExtraClasse(id, title, startTime, finishTime, (ExtraClasse\_Day) dayInput, teacherAcronym));  
 if (hasConflict) {  
 System.out.println("Class conflicts with an existing class.");  
 // Handle class conflict, display an error message or take any appropriate action  
 } else {  
 // Proceed with scheduling the class  
 extraClassController.scheduleExtraClass(id, title, startTime, finishTime, (ExtraClasse\_Day) dayInput, teacherAcronym);  
 System.out.println("Class scheduled successfully.");  
 }  
 }  
  
 \*/

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\ScheduleExtraClassUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\UpdateClassAction.java

package eapli.base.app.backoffice.console.presentation.Classe;  
  
import eapli.base.Classe.aplication.ClassController;  
import eapli.framework.actions.Action;  
  
public class UpdateClassAction implements Action {  
 @Override  
 public boolean execute() {  
 try {  
 return new UpdateClassScheduleUI().show();  
 } catch (Exception e) {  
 throw new RuntimeException(e);  
 }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\UpdateClassAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\UpdateClassScheduleUI.java

package eapli.base.app.backoffice.console.presentation.Classe;  
  
import eapli.base.Classe.aplication.ClassController;  
import eapli.base.Classe.domain.\*;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import java.time.LocalTime;  
import java.util.List;  
import java.util.Scanner;  
  
public class UpdateClassScheduleUI {  
  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final ClassController classController = new ClassController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().classRepository(), PersistenceContext.repositories().teacherRepository());  
  
  
 public boolean show() throws Exception {  
 Teacher teacher = classController.findTeacherBySystemUser(myUser);  
 List<Classe> Classes = classController.getAllClasses(teacher.getAcronym());  
 for (Classe e : Classes) {  
 System.out.println("Class title: " + e.identity() + "; Class start time : " + e.getStartTime() + "; Class end time : " + e.getFinishTime());  
 }  
  
 System.out.println("Enter class Title: ");  
 Scanner scanner = new Scanner(System.in);  
 Classe\_Title title = Classe\_Title.valueOf(scanner.nextLine());  
 scanner.nextLine();  
  
 System.out.println("Enter new class start time (HH:mm): ");  
 Classe\_Start\_Time newStartTime = Classe\_Start\_Time.valueOf(LocalTime.parse(scanner.nextLine()));  
  
 System.out.println("Enter new class finish time (HH:mm): ");  
 Classe\_Finish\_Time newFinishTime = Classe\_Finish\_Time.valueOf(LocalTime.parse(scanner.nextLine()));  
  
 classController.updateClassSchedule(title, newStartTime, newFinishTime);  
  
 System.out.println("Class schedule updated successfully.");  
 return false;  
 }  
}  
  
  
/\*  
 public class NoClassesException extends Exception {  
 public NoClassesException(String message) {  
 super(message);  
 }  
 }  
  
 \*/  
/\*  
 public void displayAllClasses() throws NoClassesException {  
 List<Classe> allClasses = classController.getAllClasses();  
  
 if (allClasses.isEmpty()) {  
 throw new NoClassesException("No classes found!");  
 }  
  
 for (Classe classe : allClasses) {  
 System.out.println(classe);  
 }  
 }  
  
  
 public void run() throws Exception {  
 Scanner scanner = new Scanner(System.in);  
 try {  
 System.out.println("Existing classes: ");  
 displayAllClasses();  
  
  
 System.out.println("Enter class ID: ");  
 Long id = scanner.nextLong();  
  
 scanner.nextLine(); // Consume newline left-over  
  
 System.out.println("Enter new class start time (HH:mm): ");  
 Classe\_Start\_Time newStartTime = Classe\_Start\_Time.valueOf(LocalTime.parse(scanner.nextLine()));  
  
 System.out.println("Enter new class finish time (HH:mm): ");  
 Classe\_Finish\_Time newFinishTime = Classe\_Finish\_Time.valueOf(LocalTime.parse(scanner.nextLine()));  
  
 classController.updateClassSchedule(id, newStartTime, newFinishTime);  
  
 System.out.println("Class schedule updated successfully.");  
 // Display all classes after updating  
 System.out.println("Updated classes: ");  
 displayAllClasses();  
  
 } catch (NoClassesException e) {  
 System.err.println(e.getMessage());  
 } catch (Exception e) {  
 }  
 }  
  
 \*/

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\Classe\UpdateClassScheduleUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\MainMenu.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation;  
  
import eapli.base.app.backoffice.console.presentation.Classe.ScheduleClassAction;  
import eapli.base.app.backoffice.console.presentation.Classe.ScheduleExtraClassAction;  
import eapli.base.app.backoffice.console.presentation.Classe.UpdateClassAction;  
import eapli.base.app.backoffice.console.presentation.SharedBoard.\*;  
import eapli.base.app.backoffice.console.presentation.authz.AddUserAction;  
import eapli.base.app.backoffice.console.presentation.course.CreateCourseAction;  
import eapli.base.app.backoffice.console.presentation.course.OpenCloseCourseAction;  
import eapli.base.app.backoffice.console.presentation.course.OpenCloseEnrollementsAction;  
import eapli.base.app.backoffice.console.presentation.enrollment.ApproveEnrollmentAction;  
import eapli.base.app.backoffice.console.presentation.enrollment.RequestEnrollmentAction;  
import eapli.base.app.backoffice.console.presentation.exam.CreateExamAction;  
import eapli.base.app.backoffice.console.presentation.exam.ListCourseExamsAction;  
import eapli.base.app.backoffice.console.presentation.exam.ListFutureExamsAction;  
import eapli.base.app.backoffice.console.presentation.meeting.AcceptRefuseInvitesAction;  
import eapli.base.app.backoffice.console.presentation.meeting.CancelMeetingAction;  
import eapli.base.app.backoffice.console.presentation.meeting.ListParticipantAction;  
import eapli.base.app.backoffice.console.presentation.meeting.ScheduleMeetingAction;  
import eapli.base.app.common.console.presentation.authz.MyUserMenu;  
import eapli.base.Application;  
import eapli.base.app.backoffice.console.presentation.authz.DeactivateUserAction;  
import eapli.base.app.backoffice.console.presentation.authz.ListUsersAction;  
import eapli.base.app.backoffice.console.presentation.clientuser.AcceptRefuseSignupRequestAction;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.actions.Actions;  
import eapli.framework.actions.menu.Menu;  
import eapli.framework.actions.menu.MenuItem;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.ExitWithMessageAction;  
import eapli.framework.presentation.console.ShowMessageAction;  
import eapli.framework.presentation.console.menu.HorizontalMenuRenderer;  
import eapli.framework.presentation.console.menu.MenuItemRenderer;  
import eapli.framework.presentation.console.menu.MenuRenderer;  
import eapli.framework.presentation.console.menu.VerticalMenuRenderer;  
  
/\*\*  
 \* TODO split this class in more specialized classes for each menu  
 \*  
 \* @author Paulo Gandra Sousa  
 \*  
 \* Menu do usuário do backoffice  
 \*/  
public class MainMenu extends AbstractUI {  
  
 private static final String RETURN\_LABEL = "Return ";  
  
 private static final int EXIT\_OPTION = 0;  
  
 // USERS - MENU  
 private static final int ADD\_USER\_OPTION = 1;  
 private static final int LIST\_USERS\_OPTION = 2;  
 private static final int DEACTIVATE\_USER\_OPTION = 3;  
 private static final int ACCEPT\_REFUSE\_SIGNUP\_REQUEST\_OPTION = 4;  
  
 // SETTINGS - MENU  
 private static final int UNDERDEVELOPMENT\_MESSAGE = 1;  
  
 // COURSE - MENU  
 private static final int CREATE\_COURSE = 1;  
 private static final int OpenCloseEnrollement = 2;  
 private static final int OpenCloseCourse = 3;  
  
 // EXAMS - MENU  
 private static final int CREATE\_EXAM = 1;  
 private static final int SEE\_COURSE\_EXAMS = 2;  
 private static final int SEE\_FUTURE\_EXAMS = 3;  
 private static final int ADMIN\_SEE\_FUTURE\_EXAMS = 3;  
  
 // CLASS - MENU  
 private static final int SCHEDULE\_CLASSES = 1;  
 private static final int UPDATE\_CLASSES = 2;  
 private static final int SCHEDULE\_EXTRA\_CLASS = 3;  
  
 // SHARED BOARD - MENU  
 private static final int SHARE\_THE\_BOARD = 1;  
 private static final int LIST\_THE\_BOARDS = 2;  
 private static final int ARCHIVE\_BOARD = 3;  
 private static final int CREATE\_POSTIT = 4;  
 private static final int UPDATE\_POSTIT = 5;  
 private static final int UNDO\_POSTIT = 6;  
  
 // ENROLLMENT - MENU  
 private static final int REQUEST\_ENROLLMENT = 1;  
 private static final int ACCEPT\_REJECT\_ENROLLMENT = 2;  
  
 // MAIN MENU - GERAL  
 private static final int MY\_USER\_OPTION = 1;  
  
 // MAIN MENU - ADMINISTRADOR  
 private static final int USERS\_OPTION = 2;  
 private static final int COURSE\_MENU = 3;  
 private static final int EXAM\_MENU = 4;  
 private static final int CLASS\_MENU = 5;  
 private static final int MEETINGS\_MENU = 6;  
 private static final int ENROLLMENT\_MENU = 7;  
 private static final int SHARED\_BOARD\_MENU = 8;  
 private static final int SETTINGS\_OPTION = 9;  
 private static final int SCHEDULE\_MEETING = 1;  
 private static final int LIST\_MEETING\_INVITES = 2;  
 private static final int LIST\_PARTICIPANTS = 3;  
  
 private static final int CANCEL\_MEETING = 3;  
  
 // MAIN MENU - MANAGER  
 private static final int USERS\_OPTION\_MANAGER = 2;  
 private static final int COURSE\_MENU\_MANAGER = 3;  
 private static final int MEETINGS\_MENU\_MANAGER = 4;  
 private static final int SHARED\_BOARD\_MENU\_MANAGER = 5;  
 private static final int SETTINGS\_OPTION\_MANAGER = 6;  
  
 // MAIN MENU - TEACHER  
 private static final int EXAM\_MENU\_TEACHER = 2;  
 private static final int CLASS\_MENU\_TEACHER = 3;  
 private static final int MEETINGS\_MENU\_TEACHER = 4;  
 private static final int SHARED\_BOARD\_MENU\_TEACHER = 5;  
  
 // MAIN MENU - STUDENT  
 private static final int EXAM\_MENU\_STUDENT = 2;  
 private static final int CLASS\_MENU\_STUDENT = 3;  
 private static final int MEETINGS\_MENU\_STUDENT = 4;  
 private static final int ENROLLMENT\_MENU\_STUDENT = 5;  
 private static final int SHARED\_BOARD\_MENU\_STUDENT = 6;  
  
 private static final String SEPARATOR\_LABEL = "--------------";  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 @Override  
 public boolean show() {  
 drawFormTitle();  
 return doShow();  
 }  
  
 /\*\*  
 \* @return true if the user selected the exit option  
 \*/  
 @Override  
 public boolean doShow() {  
 final Menu menu = buildMainMenu();  
 final MenuRenderer renderer;  
 if (Application.settings().isMenuLayoutHorizontal()) {  
 renderer = new HorizontalMenuRenderer(menu, MenuItemRenderer.DEFAULT);  
 } else {  
 renderer = new VerticalMenuRenderer(menu, MenuItemRenderer.DEFAULT);  
 }  
 return renderer.render();  
 }  
  
 @Override  
 public String headline() {  
  
 return authz.session().map(s -> "Base [ @" + s.authenticatedUser().identity() + " ]")  
 .orElse("Base [ ==Anonymous== ]");  
 }  
  
 private Menu buildMainMenu() {  
 final Menu mainMenu = new Menu();  
  
 // Opções do Usuário  
 final Menu myUserMenu = new MyUserMenu();  
 mainMenu.addSubMenu(MY\_USER\_OPTION, myUserMenu);  
  
  
 if (!Application.settings().isMenuLayoutHorizontal()) {  
 mainMenu.addItem(MenuItem.separator(SEPARATOR\_LABEL));  
 }  
  
 // Opções do Administrador  
 if (authz.isAuthenticatedUserAuthorizedTo(BaseRoles.ADMIN)) {  
 final Menu usersMenu = buildUsersMenu();  
 mainMenu.addSubMenu(USERS\_OPTION, usersMenu);  
 final Menu courseMenu = buildCourseMenu();  
 mainMenu.addSubMenu(COURSE\_MENU, courseMenu);  
 final Menu examMenu = buildExamMenu();  
 mainMenu.addSubMenu(EXAM\_MENU, examMenu);  
 final Menu classMenu = buildClassMenu();  
 mainMenu.addSubMenu(CLASS\_MENU, classMenu);  
 final Menu meetingMenu = buildMeetingMenu();  
 mainMenu.addSubMenu(MEETINGS\_MENU, meetingMenu);  
 final Menu enrollmentMenu = buildEnrollmentMenu();  
 mainMenu.addSubMenu(ENROLLMENT\_MENU, enrollmentMenu);  
 final Menu sharedMenu = buildSharedBoardMenu();  
 mainMenu.addSubMenu(SHARED\_BOARD\_MENU,sharedMenu);  
 final Menu settingsMenu = buildAdminSettingsMenu();  
 mainMenu.addSubMenu(SETTINGS\_OPTION, settingsMenu);  
 }  
  
 // Opções do Manager  
 if (authz.isAuthenticatedUserAuthorizedTo(BaseRoles.MANAGER)) {  
 final Menu usersMenu = buildUsersMenu();  
 mainMenu.addSubMenu(USERS\_OPTION\_MANAGER, usersMenu);  
 final Menu courseMenu = buildCourseMenu();  
 mainMenu.addSubMenu(COURSE\_MENU\_MANAGER, courseMenu);  
 final Menu meetingMenu = buildMeetingMenu();  
 mainMenu.addSubMenu(MEETINGS\_MENU\_MANAGER, meetingMenu);  
 final Menu sharedMenu = buildSharedBoardMenu();  
 mainMenu.addSubMenu(SHARED\_BOARD\_MENU\_MANAGER,sharedMenu);  
 final Menu settingsMenu = buildAdminSettingsMenu();  
 mainMenu.addSubMenu(SETTINGS\_OPTION\_MANAGER, settingsMenu);  
 }  
  
 // Opções do Teacher  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.TEACHER)){  
 final Menu examMenu = buildExamMenu();  
 mainMenu.addSubMenu(EXAM\_MENU\_TEACHER, examMenu);  
 final Menu classMenu = buildClassMenu();  
 mainMenu.addSubMenu(CLASS\_MENU\_TEACHER, classMenu);  
 final Menu meetingMenu = buildMeetingMenu();  
 mainMenu.addSubMenu(MEETINGS\_MENU\_TEACHER, meetingMenu);  
 final Menu sharedMenu = buildSharedBoardMenu();  
 mainMenu.addSubMenu(SHARED\_BOARD\_MENU\_TEACHER,sharedMenu);  
 }  
  
 // Opções do Student  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.STUDENT)){  
 final Menu examMenu = buildExamMenu();  
 mainMenu.addSubMenu(EXAM\_MENU\_STUDENT, examMenu);  
 final Menu classMenu = buildClassMenu();  
 mainMenu.addSubMenu(CLASS\_MENU\_STUDENT, classMenu);  
 final Menu meetingMenu = buildMeetingMenu();  
 mainMenu.addSubMenu(MEETINGS\_MENU\_STUDENT, meetingMenu);  
 final Menu enrollmentMenu = buildEnrollmentMenu();  
 mainMenu.addSubMenu(ENROLLMENT\_MENU\_STUDENT, enrollmentMenu);  
 final Menu sharedMenu = buildSharedBoardMenu();  
 mainMenu.addSubMenu(SHARED\_BOARD\_MENU\_STUDENT,sharedMenu);  
 }  
  
 if (!Application.settings().isMenuLayoutHorizontal()) {  
 mainMenu.addItem(MenuItem.separator(SEPARATOR\_LABEL));  
 }  
  
 mainMenu.addItem(EXIT\_OPTION, "Exit", new ExitWithMessageAction("Exiting the application... Bye!"));  
  
 return mainMenu;  
 }  
  
 //Menu do Exam  
 private Menu buildExamMenu(){  
 final Menu menu = new Menu("Exam >");  
  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.TEACHER, BaseRoles.ADMIN)){  
 menu.addItem(CREATE\_EXAM, "Create a new exam", new CreateExamAction());  
 }  
  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.TEACHER, BaseRoles.ADMIN)){  
 menu.addItem(SEE\_COURSE\_EXAMS, "View list of all exams in a course", new ListCourseExamsAction());  
 }  
  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.STUDENT)){  
 menu.addItem(SEE\_FUTURE\_EXAMS, "View list of all my future exams", new ListFutureExamsAction());  
 }  
  
 // ID Option diferente para o admin para evitar conflitos e erros de execução  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.ADMIN)){  
 menu.addItem(ADMIN\_SEE\_FUTURE\_EXAMS, "View list of all my future exams", new ListFutureExamsAction());  
 }  
  
 menu.addItem(EXIT\_OPTION, RETURN\_LABEL, Actions.SUCCESS);  
  
 return menu;  
 }  
  
 //Menu das Classes  
 private Menu buildClassMenu(){  
 final Menu menu = new Menu("Class >");  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.TEACHER, BaseRoles.ADMIN)){  
 menu.addItem(SCHEDULE\_CLASSES, "Schedule a new class",new ScheduleClassAction());  
  
 }  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.TEACHER, BaseRoles.ADMIN)){  
 menu.addItem(UPDATE\_CLASSES, "Update an existing class", new UpdateClassAction());  
  
 }  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.TEACHER, BaseRoles.ADMIN)){  
 menu.addItem(SCHEDULE\_EXTRA\_CLASS, "Schedule a new extra class", new ScheduleExtraClassAction());  
  
 }  
 menu.addItem(EXIT\_OPTION, RETURN\_LABEL, Actions.SUCCESS);  
 return menu;  
 }  
  
 // Menu do Course  
 private Menu buildCourseMenu() {  
 final Menu menu = new Menu("Course >");  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.MANAGER, BaseRoles.ADMIN)){  
 menu.addItem(CREATE\_COURSE, "Create a new Course", new CreateCourseAction());  
 menu.addItem(OpenCloseEnrollement, "Open/Close enrollements", new OpenCloseEnrollementsAction());  
 menu.addItem(OpenCloseCourse, "Open/Close course", new OpenCloseCourseAction());  
 }  
 menu.addItem(EXIT\_OPTION, RETURN\_LABEL, Actions.SUCCESS);  
  
 return menu;  
 }  
  
 private Menu buildSharedBoardMenu(){  
 final Menu menu = new Menu("SharedBoard >");  
  
 menu.addItem(SHARE\_THE\_BOARD, "Share the board with someone",new SharedBoardAction());  
 menu.addItem(LIST\_THE\_BOARDS, "List all the boards",new SharedBoardListAction());  
 menu.addItem(ARCHIVE\_BOARD, "Archive the board",new SharedBoardArchiveAction());  
 menu.addItem(CREATE\_POSTIT, "Create a post-it in a board", new CreateBoardPostItAction());  
 menu.addItem(UPDATE\_POSTIT, "Update a post-it in a board", new UpdateBoardPostItAction());  
 menu.addItem(UNDO\_POSTIT, "Undo the last change in a post-it", new UndoPostItAction());  
  
  
 return menu;  
 }  
  
 // Menu do Meeting  
 private Menu buildMeetingMenu(){  
 final Menu menu = new Menu("Meeting >");  
  
 menu.addItem(SCHEDULE\_MEETING, "Schedule a meeting", new ScheduleMeetingAction());  
 menu.addItem(LIST\_MEETING\_INVITES, "View list of meetings I've been invited to", new AcceptRefuseInvitesAction());  
 menu.addItem(LIST\_PARTICIPANTS, "List participants", new ListParticipantAction());  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.ADMIN)) {  
 menu.addItem(CANCEL\_MEETING, "Cancel the Meeting", new CancelMeetingAction());  
 }  
 menu.addItem(EXIT\_OPTION, RETURN\_LABEL, Actions.SUCCESS);  
  
 return menu;  
 }  
  
 private Menu buildEnrollmentMenu() {  
 final Menu menu = new Menu("Enrollment >");  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.STUDENT, BaseRoles.ADMIN)) {  
 menu.addItem(REQUEST\_ENROLLMENT, "Request Enrollment", new RequestEnrollmentAction());  
 }  
 if(authz.isAuthenticatedUserAuthorizedTo(BaseRoles.MANAGER, BaseRoles.ADMIN)) {  
 menu.addItem(ACCEPT\_REJECT\_ENROLLMENT, "Accept / Reject enrollment requests", new ApproveEnrollmentAction());  
 }  
 menu.addItem(EXIT\_OPTION, RETURN\_LABEL, Actions.SUCCESS);  
  
 return menu;  
 }  
  
 // Settings do Administrador  
 private Menu buildAdminSettingsMenu() {  
 final Menu menu = new Menu("Settings >");  
  
 menu.addItem(UNDERDEVELOPMENT\_MESSAGE, "Testing Option", new ShowMessageAction("This option is under development"));  
 menu.addItem(EXIT\_OPTION, RETURN\_LABEL, Actions.SUCCESS);  
  
 return menu;  
 }  
  
 private Menu buildUsersMenu() {  
 final Menu menu = new Menu("Users >");  
  
 menu.addItem(ADD\_USER\_OPTION, "Add User", new AddUserAction());  
 menu.addItem(LIST\_USERS\_OPTION, "List all Users", new ListUsersAction());  
 menu.addItem(DEACTIVATE\_USER\_OPTION, "Deactivate User", new DeactivateUserAction());  
 menu.addItem(ACCEPT\_REFUSE\_SIGNUP\_REQUEST\_OPTION, "Accept/Refuse Signup Request", new AcceptRefuseSignupRequestAction());  
 menu.addItem(EXIT\_OPTION, RETURN\_LABEL, Actions.SUCCESS);  
  
 return menu;  
 }  
  
  
  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\MainMenu.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\CreateBoardPostItAction.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.framework.actions.Action;  
  
public class CreateBoardPostItAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new CreateBoardPostItUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\CreateBoardPostItAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\CreateBoardPostItUI.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.base.SharedBoard.aplication.CreateBoardPostItController;  
import eapli.base.SharedBoard.aplication.SharedBoardController;  
import eapli.base.SharedBoard.domain.NumberofColumns;  
import eapli.base.SharedBoard.domain.NumberofRows;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
public class CreateBoardPostItUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final CreateBoardPostItController controller = new CreateBoardPostItController(  
 PersistenceContext.repositories().sharedBoardRepository());  
 private final SharedBoardRepository sharedBoardRepository = PersistenceContext.repositories().sharedBoardRepository();  
  
 @Override  
 protected boolean doShow() {  
// Iterable<SharedBoard> sharedBoards = controller.findBoardsBySystemUser(myUser);  
// final SelectWidget<SharedBoard> selector = new SelectWidget<>("Shared Boards:", sharedBoards, new SharedBoardPrinter());  
// selector.show();  
// final SharedBoard sharedBoard = selector.selectedElement();  
 SharedBoard sharedBoard = new SharedBoard(myUser, new Shared\_Board\_Title("Shared Board 1"), new NumberofRows(10), new NumberofColumns(10));  
 sharedBoard.shareBoard(myUser,myUser,SharedBoard.AccessType.valueOf("WRITE"));  
 sharedBoard = sharedBoardRepository.save(sharedBoard);  
  
// if(sharedBoard!=null){  
 if(controller.getBoardAccessType(sharedBoard, myUser)){  
 String content = Console.readLine("Type the content of the post-it.");  
 controller.createPostIt(content, myUser);  
 int row = Console.readInteger("Type the row of the board you wish to put the post-it.");  
 int column = Console.readInteger("Type the column of the board you wish to put the post-it.");  
 try {  
 controller.addPostIt(sharedBoard, row, column);  
 } catch (InterruptedException e) {  
 throw new RuntimeException(e);  
 }  
  
 }else{  
 System.out.println("You don't have permission to write on this board.");  
 }  
  
// }  
  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Create a post-it in a board";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\CreateBoardPostItUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardAction.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.framework.actions.Action;  
  
  
public class SharedBoardAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new SharedBoardUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardArchiveAction.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.base.SharedBoard.aplication.SharedBoardArchiveController;  
import eapli.base.SharedBoard.aplication.SharedBoardController;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
import javax.swing.\*;  
import eapli.framework.actions.Action;  
import java.awt.event.ActionEvent;  
import java.beans.PropertyChangeListener;  
  
public class SharedBoardArchiveAction implements Action{  
 @Override  
 public boolean execute() {  
 return new SharedBoardArchiveUI().show();  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardArchiveAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardArchiveUI.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.base.SharedBoard.aplication.SharedBoardArchiveController;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
public class SharedBoardArchiveUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
  
 private final SharedBoardArchiveController controller = new SharedBoardArchiveController(authz, PersistenceContext.repositories().sharedBoardRepository(), PersistenceContext.repositories().systemUserRepository());  
  
 @Override  
 public String headline() {  
 return "Archive a board";  
 }  
  
 @Override  
 protected boolean doShow() {  
 Iterable<SharedBoard> sharedBoards = controller.findSharedBoardsByOwner(myUser);  
 final SelectWidget<SharedBoard> selector = new SelectWidget<>("Shared Boards:", sharedBoards, new SharedBoardPrinter());  
 selector.show();  
 final SharedBoard sharedBoard = selector.selectedElement();  
 if(sharedBoard!=null){  
 Shared\_Board\_Title title = Shared\_Board\_Title.valueOf(Console.readLine("Enter the title of the board you want to archive: "));  
 try {  
 controller.archiveBoard(title, myUser.identity());  
 System.out.println("Board with title " + title + " has been archived.");  
 } catch (IllegalStateException e) {  
 System.out.println("Failed to archive board: " + e.getMessage());  
 }  
 }else{  
 System.out.println("No shared board selected. Please select a board to archive.");  
 }  
  
 return false;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardArchiveUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardListAction.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
  
import eapli.framework.actions.Action;  
  
  
public class SharedBoardListAction implements Action {  
 @Override  
 public boolean execute() {  
 return new SharedBoardListUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardListAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardListUI.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
 import eapli.base.SharedBoard.domain.SharedBoard;  
 import eapli.base.infrastructure.persistence.PersistenceContext;  
 import eapli.base.SharedBoard.aplication.SharedBoardController;  
  
 import eapli.base.systemUserManagement.SystemUserRepository;  
 import eapli.framework.infrastructure.authz.application.AuthorizationService;  
 import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
 import eapli.framework.infrastructure.authz.application.UserSession;  
 import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
 import eapli.framework.presentation.console.AbstractUI;  
  
 public class SharedBoardListUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final SharedBoardController controller = new SharedBoardController(PersistenceContext.repositories().sharedBoardRepository(), PersistenceContext.repositories().systemUserRepository());  
  
  
 @Override  
 protected boolean doShow() {  
 final Iterable<SharedBoard> sharedBoards = this.controller.findSharedBoardsByOwner(myUser);  
 for(SharedBoard sb : sharedBoards){  
 System.out.println("Shared Board title: " + sb.getTitle());  
 }  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "List shared boards";  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardListUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardPrinter.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.framework.visitor.Visitor;  
  
public class SharedBoardPrinter implements Visitor<SharedBoard> {  
  
 @Override  
 public void visit(SharedBoard visitee) {  
 System.out.printf(visitee.getTitle().getTitle());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardPrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardUI.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.base.SharedBoard.aplication.SharedBoardController;  
import eapli.base.SharedBoard.aplication.SharedBoardService;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.io.util.Console;  
  
public class SharedBoardUI extends AbstractUI {  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final SharedBoardController controller = new SharedBoardController(PersistenceContext.repositories().sharedBoardRepository(), PersistenceContext.repositories().systemUserRepository());  
  
 @Override  
 public String headline() {  
 return "Share a board";  
 }  
  
 @Override  
 protected boolean doShow() {  
  
 String ownerUsername = myUser.username().toString();  
 String sharedBoardTitle = Console.readNonEmptyLine("Enter the title of the shared board", "Shared board title cannot be empty");  
 int numberOfRows = Console.readInteger("Enter the number of rows of the board");  
 int numberOfColumns = Console.readInteger("Enter the number of columns for the board");  
 String userToShareUsername = Console.readNonEmptyLine("Enter the username of the user you want to share the board with: ", "Username cannot be empty");  
 String accessType = Console.readNonEmptyLine("Enter the access type for the user (READ/WRITE): ", "Access type cannot be empty");  
 try {  
 controller.createSharedBoard(myUser, sharedBoardTitle, numberOfRows, numberOfColumns);  
 controller.shareBoard(ownerUsername, userToShareUsername, SharedBoard.AccessType.valueOf(accessType.toUpperCase()));  
 System.out.println("Shared board with user: " + userToShareUsername + " with " + accessType + " access.");  
 } catch (IllegalStateException e) {  
 System.out.println("Failed to share board: " + e.getMessage());  
 } catch (InterruptedException e) {  
 throw new RuntimeException(e);  
 }  
  
 return false;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\SharedBoardUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UndoPostItAction.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.framework.actions.Action;  
  
public class UndoPostItAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new UndoPostItUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UndoPostItAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UndoPostItUI.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.base.SharedBoard.aplication.CreateBoardPostItController;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
public class UndoPostItUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final CreateBoardPostItController controller = new CreateBoardPostItController(  
 PersistenceContext.repositories().sharedBoardRepository());  
 @Override  
 protected boolean doShow() {  
 Iterable<SharedBoard> sharedBoards = controller.findBoardsBySystemUser(myUser);  
 final SelectWidget<SharedBoard> selector = new SelectWidget<>("Shared Boards:", sharedBoards, new SharedBoardPrinter());  
 selector.show();  
 final SharedBoard sharedBoard = selector.selectedElement();  
 if(sharedBoard!=null){  
 if(controller.getBoardAccessType(sharedBoard, myUser)) {  
  
 try {  
 controller.undoPostIt();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 else{  
 System.out.println("You don't have permission to undo a post it in this board.");  
 }  
  
 }  
  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Undo Post it";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UndoPostItUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UpdateBoardPostItAction.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.framework.actions.Action;  
  
public class UpdateBoardPostItAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new UpdateBoardPostItUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UpdateBoardPostItAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UpdateBoardPostItUI.java

package eapli.base.app.backoffice.console.presentation.SharedBoard;  
  
import eapli.base.SharedBoard.aplication.CreateBoardPostItController;  
import eapli.base.SharedBoard.aplication.UpdateBoardPostItController;  
import eapli.base.SharedBoard.domain.\*;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
public class UpdateBoardPostItUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final UpdateBoardPostItController controller = new UpdateBoardPostItController(  
 PersistenceContext.repositories().sharedBoardRepository());  
  
 private final SharedBoardRepository sharedBoardRepository = PersistenceContext.repositories().sharedBoardRepository();  
  
 @Override  
 protected boolean doShow() {  
// Iterable<SharedBoard> sharedBoards = controller.findBoardsBySystemUser(myUser);  
// final SelectWidget<SharedBoard> selector = new SelectWidget<>("Shared Boards:", sharedBoards, new SharedBoardPrinter());  
// selector.show();  
// final SharedBoard sharedBoard = selector.selectedElement();  
 SharedBoard sharedBoard = new SharedBoard(myUser, new Shared\_Board\_Title("Shared Board 1"), new NumberofRows(10), new NumberofColumns(10));  
 sharedBoard.shareBoard(myUser,myUser,SharedBoard.AccessType.valueOf("WRITE"));  
 sharedBoard = sharedBoardRepository.save(sharedBoard);  
 sharedBoard.getBoard().insertCell(1,1,new PostIt("test",myUser));  
  
 if(sharedBoard!=null){  
 if(controller.getBoardAccessType(sharedBoard, myUser)){  
 System.out.println("Type \"0\" to update a created post-it or \"1\" to move a created post-it to a new position.");  
 int option=Console.readOption(0,1,0);  
 if(option==0){  
 int row = Console.readInteger("Type the row of the board where the post-it you wish to update is present.");  
 int column = Console.readInteger("Type the column of the board where the post-it you wish to update is present.");  
 String content = Console.readLine("Type the content of the post-it you wish to update.");  
 controller.createPostIt(content, myUser);  
 try {  
 controller.updatePostIt(sharedBoard, row, column);  
 } catch (InterruptedException e) {  
 throw new RuntimeException(e);  
 }  
 }else{  
 int row = Console.readInteger("Type the current row of the board where the post-it is present.");  
 int column = Console.readInteger("Type the current column of the board where the post-it is present.");  
 int newRow = Console.readInteger("Type the new row of the board you wish to move the post-it.");  
 int newColumn = Console.readInteger("Type the new column of the board you wish to move the post-it.");  
 try {  
 controller.movePostIt(sharedBoard, row, column, newRow, newColumn, myUser);  
 } catch (InterruptedException e) {  
 throw new RuntimeException(e);  
 }  
 }  
  
 }else{  
 System.out.println("You don't have permission to write on this board.");  
 }  
  
 }  
  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Update a post-it in a board";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\SharedBoard\UpdateBoardPostItUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\AddUserAction.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.authz;  
  
import eapli.framework.actions.Action;  
  
/\*\*  
 \* Menu action for adding a new user to the application. Created by nuno on  
 \* 22/03/16.  
 \*/  
public class AddUserAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new AddUserUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\AddUserAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\AddUserUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.authz;  
  
import java.util.HashSet;  
import java.util.Set;  
  
import eapli.base.usermanagement.application.AddUserController;  
import eapli.framework.actions.Actions;  
import eapli.framework.actions.menu.Menu;  
import eapli.framework.actions.menu.MenuItem;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.menu.MenuItemRenderer;  
import eapli.framework.presentation.console.menu.MenuRenderer;  
import eapli.framework.presentation.console.menu.VerticalMenuRenderer;  
  
/\*\*  
 \* UI for adding a user to the application.  
 \* Created by nuno on 22/03/16.  
 \*/  
public class AddUserUI extends AbstractUI {  
  
 private final AddUserController theController = new AddUserController();  
  
 @Override  
 protected boolean doShow() {  
 // FIXME avoid duplication with SignUpUI. reuse UserDataWidget from  
 // UtenteApp  
 final String username = Console.readLine("Username");  
 final String password = Console.readLine("Password");  
 final String firstName = Console.readLine("First Name");  
 final String lastName = Console.readLine("Last Name");  
 final String email = Console.readLine("E-Mail");  
  
 final Set<Role> roleTypes = new HashSet<>();  
 boolean show;  
 do {  
 show = showRoles(roleTypes);  
 } while (!show);  
  
 try {  
 this.theController.addUser(username, password, firstName, lastName, email, roleTypes);  
 } catch (final IntegrityViolationException | ConcurrencyException e) {  
 System.out.println("That username is already in use.");  
 }  
  
 return false;  
 }  
  
 private boolean showRoles(final Set<Role> roleTypes) {  
 // TODO we could also use the "widget" classes from the framework...  
 final Menu rolesMenu = buildRolesMenu(roleTypes);  
 final MenuRenderer renderer = new VerticalMenuRenderer(rolesMenu, MenuItemRenderer.DEFAULT);  
 return renderer.render();  
 }  
  
 private Menu buildRolesMenu(final Set<Role> roleTypes) {  
 final Menu rolesMenu = new Menu();  
 int counter = 0;  
 rolesMenu.addItem(MenuItem.of(counter++, "USER / NO ROLE", Actions.SUCCESS));  
 for (final Role roleType : theController.getRoleTypes()) {  
 rolesMenu.addItem(MenuItem.of(counter++, roleType.toString(), () -> roleTypes.add(roleType)));  
 }  
 return rolesMenu;  
 }  
  
 @Override  
 public String headline() {  
 return "Add User";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\AddUserUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\DeactivateUserAction.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.authz;  
  
import eapli.framework.actions.Action;  
  
/\*\*  
 \*  
 \* @author Fernando  
 \*/  
public class DeactivateUserAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new DeactivateUserUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\DeactivateUserAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\DeactivateUserUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.authz;  
  
import java.util.ArrayList;  
import java.util.List;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import eapli.base.usermanagement.application.DeactivateUserController;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
/\*\*  
 \*  
 \* @author Fernando  
 \*/  
@SuppressWarnings("squid:S106")  
public class DeactivateUserUI extends AbstractUI {  
 private static final Logger LOGGER = LoggerFactory.getLogger(DeactivateUserUI.class);  
  
 private final DeactivateUserController theController = new DeactivateUserController();  
  
 @Override  
 protected boolean doShow() {  
 final List<SystemUser> list = new ArrayList<>();  
 final Iterable<SystemUser> iterable = this.theController.activeUsers();  
 if (!iterable.iterator().hasNext()) {  
 System.out.println("There is no registered User");  
 } else {  
 int cont = 1;  
 System.out.println("SELECT User to deactivate\n");  
 // FIXME use select widget, see, ChangeDishTypeUI  
 System.out.printf("%-6s%-10s%-30s%-30s%n", "Nº:", "Username", "Firstname", "Lastname");  
 for (final SystemUser user : iterable) {  
 list.add(user);  
 System.out.printf("%-6d%-10s%-30s%-30s%n", cont, user.username(), user.name().firstName(),  
 user.name().lastName());  
 cont++;  
 }  
 final int option = Console.readInteger("Enter user nº to deactivate or 0 to finish ");  
 if (option == 0) {  
 System.out.println("No user selected");  
 } else {  
 try {  
 this.theController.deactivateUser(list.get(option - 1));  
 } catch (IntegrityViolationException | ConcurrencyException ex) {  
 LOGGER.error("Error performing the operation", ex);  
 System.out.println("Unfortunatelly there was an unexpected error in the application. Please try again and if the problem persists, contact your system admnistrator.");  
 }  
 }  
 }  
 return true;  
 }  
  
 @Override  
 public String headline() {  
 return "Deactivate User";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\DeactivateUserUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\ListUsersAction.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.authz;  
  
import eapli.framework.actions.Action;  
  
/\*\*  
 \*  
 \* @author losa  
 \*/  
public class ListUsersAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new ListUsersUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\ListUsersAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\ListUsersUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.authz;  
  
import eapli.base.usermanagement.application.ListUsersController;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.presentation.console.AbstractListUI;  
import eapli.framework.visitor.Visitor;  
  
/\*\*  
 \*  
 \* @author losa  
 \*/  
@SuppressWarnings({ "squid:S106" })  
public class ListUsersUI extends AbstractListUI<SystemUser> {  
 private ListUsersController theController = new ListUsersController();  
  
 @Override  
 public String headline() {  
 return "List Users";  
 }  
  
 @Override  
 protected String emptyMessage() {  
 return "No data.";  
 }  
  
 @Override  
 protected Iterable<SystemUser> elements() {  
 return theController.allUsers();  
 }  
  
 @Override  
 protected Visitor<SystemUser> elementPrinter() {  
 return new SystemUserPrinter();  
 }  
  
 @Override  
 protected String elementName() {  
 return "User";  
 }  
  
 @Override  
 protected String listHeader() {  
 return String.format("# %-10s%-30s%-30s", "USERNAME", "F. NAME", "L. NAME");  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\ListUsersUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\SystemUserPrinter.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.authz;  
  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.visitor.Visitor;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*  
 \*/  
@SuppressWarnings({ "squid:S106" })  
public class SystemUserPrinter implements Visitor<SystemUser> {  
  
 @Override  
 public void visit(final SystemUser visitee) {  
 System.out.printf("%-10s%-30s%-30s", visitee.username(), visitee.name().firstName(), visitee.name().lastName());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\authz\SystemUserPrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\clientuser\AcceptRefuseSignupRequestAction.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.clientuser;  
  
import eapli.framework.actions.Action;  
  
/\*\*  
 \* Created by AJS on 08/04/2016.  
 \*/  
public class AcceptRefuseSignupRequestAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new AcceptRefuseSignupRequestUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\clientuser\AcceptRefuseSignupRequestAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\clientuser\AcceptRefuseSignupRequestUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.clientuser;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import eapli.base.clientusermanagement.application.AcceptRefuseSignupFactory;  
import eapli.base.clientusermanagement.application.AcceptRefuseSignupRequestController;  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
/\*\*  
 \*  
 \* Created by AJS on 08/04/2016.  
 \*/  
@SuppressWarnings("squid:S106")  
public class AcceptRefuseSignupRequestUI extends AbstractUI {  
 private static final Logger LOGGER = LoggerFactory.getLogger(AcceptRefuseSignupRequestUI.class);  
  
 private final AcceptRefuseSignupRequestController theController = AcceptRefuseSignupFactory  
 .build();  
  
 @Override  
 protected boolean doShow() {  
 final SelectWidget<SignupRequest> selector = new SelectWidget<>("Pending signups", this.theController.listPendingSignupRequests(), new SignupRequestPrinter());  
 selector.show();  
 final SignupRequest theSignupRequest = selector.selectedElement();  
 if (theSignupRequest != null) {  
 System.out.println("1. Accept Signup Request");  
 System.out.println("2. Refuse Signup Request");  
 System.out.println("0. Exit");  
  
 final int option = Console.readOption(1, 2, 0);  
 try {  
 switch (option) {  
 case 1:  
 this.theController.acceptSignupRequest(theSignupRequest);  
 break;  
 case 2:  
 this.theController.refuseSignupRequest(theSignupRequest);  
 break;  
 default:  
 System.out.println("No valid option selected");  
 break;  
 }  
 } catch (IntegrityViolationException | ConcurrencyException ex) {  
 LOGGER.error("Error performing the operation", ex);  
 System.out.println("Unfortunatelly there was an unexpected error in the application. Please try again and if the problem persists, contact your system admnistrator.");  
 }  
 }  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Accept of Refuse Signup Requests";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\clientuser\AcceptRefuseSignupRequestUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\clientuser\SignupRequestPrinter.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.backoffice.console.presentation.clientuser;  
  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.framework.visitor.Visitor;  
  
/\*\*  
 \* Created by AJS on 08/04/2016.  
 \*  
 \*/  
@SuppressWarnings("squid:S106")  
class SignupRequestPrinter implements Visitor<SignupRequest> {  
  
 @Override  
 public void visit(final SignupRequest visitee) {  
 System.out.printf("%-10s%-20s%-10s%n", visitee.identity(), visitee.name(),  
 visitee.mecanographicNumber());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\clientuser\SignupRequestPrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\CoursePrinter.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import eapli.base.Course.Domain.Course;  
import eapli.framework.visitor.Visitor;  
  
public class CoursePrinter implements Visitor<Course> {  
  
 @Override  
 public void visit(final Course visitee) {  
 System.out.printf("%-10s%-30s%-4s", visitee.identity(), visitee.getSmallTextualDescription(), visitee.getTeacherInCharge().toString());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\CoursePrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\CreateCourseAction.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import eapli.framework.actions.Action;  
  
public class CreateCourseAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new CreateCourseUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\CreateCourseAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\CreateCourseUI.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import eapli.base.Course.Application.CreateCourseController;  
import eapli.base.Course.Domain.Course;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
public class CreateCourseUI extends AbstractUI {  
  
 // Necessário completar esta US, ainda está incompleta, seguir um padrão semelhante ao das outras US  
 //falta validar os cursos. Verificar se já nao existem  
 private final CreateCourseController createCourseController = new CreateCourseController(PersistenceContext.repositories().courseRepository(), PersistenceContext.repositories().teacherRepository());  
 Course course = new Course();  
 @Override  
 protected boolean doShow() {  
 String name = Console.readLine("Name of the Course:");  
 int minNumOfStud = Console.readInteger("Minimum number of students:");  
 int maxNumOfStud = Console.readInteger("Maximum number of students:");  
 String description = Console.readLine("Course Description:");  
 final Iterable<Teacher> teachers = this.createCourseController.listTeachers();  
 final SelectWidget<Teacher> selector = new SelectWidget<>("Teachers:", teachers, new TeacherPrinter());  
 selector.show();  
 final Teacher teacher = selector.selectedElement();  
 if (teacher != null) {  
 try {  
 createCourseController.createCourse(name,minNumOfStud,maxNumOfStud,description, teacher);  
 } catch (final IntegrityViolationException e) {  
 System.out.println("Invalid Course");  
 }  
 }  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Create Course";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\CreateCourseUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseCourseAction.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import javax.swing.\*;  
import eapli.framework.actions.Action;  
  
public class OpenCloseCourseAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new OpenCloseCourseUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseCourseAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseCourseUI.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import eapli.base.Course.Application.OpenCloseCourseController;  
import eapli.base.Course.Application.OpenCloseEnrollementsController;  
import eapli.base.Course.Domain.Course\_ID;  
import eapli.base.Course.Domain.Course\_Name;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
import java.sql.SQLOutput;  
import java.util.Scanner;  
  
public class OpenCloseCourseUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final OpenCloseCourseController controller = new OpenCloseCourseController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().courseRepository());  
  
  
 @Override  
 protected boolean doShow() {  
 Scanner scanner = new Scanner(System.in);  
  
 String courseName = Console.readLine("Digite o nome do curso que deseja abrir ou fechar:");  
 Course\_Name courseName1 = new Course\_Name(courseName);  
  
 System.out.println("Digite 'abrir' para abrir ou 'fechar' para fechar o curso");  
 String option = scanner.next();  
  
 if (option.equalsIgnoreCase("abrir")) {  
 controller.openCourse(courseName1);  
 System.out.println("O curso " + courseName + " foi aberto com sucesso.");  
 } else if (option.equalsIgnoreCase("fechar")) {  
 controller.closeCourse(courseName1);  
 System.out.println("O curso " + courseName + " foi fechado com sucesso.");  
 } else {  
 System.out.println("Opção inválida.");  
 }  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Open/Close Course";  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseCourseUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseEnrollementsAction.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import eapli.base.app.backoffice.console.presentation.exam.ListCourseExamsUI;  
import eapli.framework.actions.Action;  
import javax.swing.\*;  
  
public class OpenCloseEnrollementsAction implements Action {  
 @Override  
 public boolean execute() {  
 return new OpenCloseEnrollementsUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseEnrollementsAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseEnrollementsUI.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import eapli.base.Course.Application.CreateCourseController;  
import eapli.base.Course.Application.OpenCloseEnrollementsController;  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Domain.Course\_ID;  
import eapli.base.Course.Domain.Course\_Name;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.application.ListFutureExamsController;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
import java.util.Scanner;  
  
public class OpenCloseEnrollementsUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final OpenCloseEnrollementsController controller = new OpenCloseEnrollementsController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().courseRepository());  
  
  
 @Override  
 protected boolean doShow() {  
 Scanner scanner = new Scanner(System.in);  
  
 String courseName = Console.readLine("Digite o nome do curso para abrir ou fechar as inscrições: ");  
 Course\_Name courseName1 = new Course\_Name(courseName);  
  
 System.out.print("Digite 'abrir' para abrir as inscrições ou 'fechar' para fechá-las: ");  
 String option = scanner.next();  
  
 if (option.equalsIgnoreCase("abrir")) {  
 controller.openEnrollments(courseName1);  
 System.out.println("As inscrições para o curso " + courseName + " foram abertas com sucesso.");  
 } else if (option.equalsIgnoreCase("fechar")) {  
 controller.closeEnrollments(courseName1);  
 System.out.println("As inscrições para o curso " + courseName + " foram fechadas com sucesso.");  
 } else {  
 System.out.println("Opção inválida.");  
 }  
  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Open/Close enrollements";  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\OpenCloseEnrollementsUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\TeacherPrinter.java

package eapli.base.app.backoffice.console.presentation.course;  
  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.framework.visitor.Visitor;  
  
public class TeacherPrinter implements Visitor<Teacher> {  
  
 @Override  
 public void visit(Teacher visitee) {  
 System.out.printf("Teacher acronym: %s, Date of birth: %s, Tax payer number: %s\n",  
 visitee.getAcronym(), visitee.getDate\_of\_Birth().getDateOfBirth(), visitee.getTax\_payer\_number().getNumber());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\course\TeacherPrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\ApproveEnrollmentAction.java

package eapli.base.app.backoffice.console.presentation.enrollment;  
  
import eapli.base.app.backoffice.console.presentation.exam.ListCourseExamsUI;  
import eapli.framework.actions.Action;  
  
public class ApproveEnrollmentAction implements Action {  
 @Override  
 public boolean execute() {  
 return new ApproveEnrollmentUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\ApproveEnrollmentAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\ApproveEnrollmentUI.java

package eapli.base.app.backoffice.console.presentation.enrollment;  
  
import eapli.base.Enrollment.Application.ApproveEnrollmentController;  
import eapli.base.Enrollment.Application.ListEnrollmentsController;  
import eapli.base.Enrollment.Domain.Enrollment;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
  
  
public class ApproveEnrollmentUI extends AbstractUI {  
  
 private final ListEnrollmentsController listEnrollmentsController = new ListEnrollmentsController(PersistenceContext.repositories().enrollmentRepository());  
  
 private final ApproveEnrollmentController controller = new ApproveEnrollmentController(PersistenceContext.repositories().enrollmentRepository());  
  
  
 @Override  
 protected boolean doShow() {  
  
 final Iterable<Enrollment> enrollments = this.listEnrollmentsController.listEnrollments();  
 final SelectWidget<Enrollment> selector = new SelectWidget<>("Enrollments:", enrollments, new EnrollmentPrinter());  
 selector.show();  
 final Enrollment chosenEnrollment = selector.selectedElement();  
 int isToBeAccepted = Console.readInteger("Do you want to accept this enrollment?(1-Accept/0-Reject)");  
 controller.approveEnrollment(chosenEnrollment,isToBeAccepted);  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "--Approve Enrollment UI--";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\ApproveEnrollmentUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\BulkEnrollmentUI.java

package eapli.base.app.backoffice.console.presentation.enrollment;  
  
import eapli.base.Enrollment.Application.BulkEnrollmentController;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
public class BulkEnrollmentUI extends AbstractUI {  
 private final BulkEnrollmentController bulkEnrollmentController = new BulkEnrollmentController(PersistenceContext.repositories().enrollmentRepository());  
  
 @Override  
 protected boolean doShow() {  
 String csvPath = Console.readLine("Please, indicate the CSV file path which you pretend to bulk Enroll:");  
 bulkEnrollmentController.bulkEnrollment(csvPath);  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "--Bulk Enrollment UI--";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\BulkEnrollmentUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\EnrollmentPrinter.java

package eapli.base.app.backoffice.console.presentation.enrollment;  
  
import eapli.base.Enrollment.Domain.Enrollment;  
import eapli.framework.visitor.Visitor;  
  
public class EnrollmentPrinter implements Visitor<Enrollment> {  
  
 @Override  
 public void visit(final Enrollment visitee) {  
 System.out.printf("%-10s%-30s%-4s", visitee.identity(),visitee.getStudent().toString(), visitee.getCourse().toString(),visitee.getDescription());  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\EnrollmentPrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\RequestEnrollmentAction.java

package eapli.base.app.backoffice.console.presentation.enrollment;  
  
import eapli.framework.actions.Action;  
  
public class RequestEnrollmentAction implements Action {  
 @Override  
 public boolean execute() {  
 return new RequestEnrollmentUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\RequestEnrollmentAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\RequestEnrollmentUI.java

package eapli.base.app.backoffice.console.presentation.enrollment;  
  
import eapli.base.Course.Application.ListCoursesController;  
import eapli.base.Course.Domain.Course;  
import eapli.base.Enrollment.Application.RequestEnrollmentController;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.app.backoffice.console.presentation.course.CoursePrinter;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
public class RequestEnrollmentUI extends AbstractUI {  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private RequestEnrollmentController controller = new RequestEnrollmentController(PersistenceContext.repositories().enrollmentRepository(), PersistenceContext.repositories().studentRepository());  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
 private final ListCoursesController listCoursesController = new ListCoursesController(PersistenceContext.repositories().courseRepository());  
  
  
 @Override  
 protected boolean doShow() {  
 Student student = controller.findStudentBySystemUser(myUser);  
  
 final Iterable<Course> courses = this.listCoursesController.listCourses();  
 final SelectWidget<Course> selector = new SelectWidget<>("Courses:", courses, new CoursePrinter());  
 selector.show();  
 final Course choosenCourse = selector.selectedElement();  
  
 String description = Console.readLine("What is the Enrollment description:");  
 controller.requestEnrollment(student, choosenCourse, description);  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "--Request Enrollment UI--";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\enrollment\RequestEnrollmentUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\CreateExamAction.java

package eapli.base.app.backoffice.console.presentation.exam;  
  
import eapli.framework.actions.Action;  
  
public class CreateExamAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new CreateExamUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\CreateExamAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\CreateExamUI.java

package eapli.base.app.backoffice.console.presentation.exam;  
  
import eapli.base.exammanagement.application.CreateExamController;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import org.springframework.expression.ParseException;  
  
import java.sql.Time;  
import java.text.DateFormat;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
public class CreateExamUI extends AbstractUI {  
  
 private final CreateExamController controller = new CreateExamController();  
  
 @Override  
 protected boolean doShow() {  
 final String examTitle = Console.readLine("Exam Title");  
 final Date examDate = Console.readDate("Exam Date (YYYY/MM/DD)", "YYYY/MM/dd");  
 final Time openTime = readTime("Open Time (HH:mm)");  
 final Time closeTime = readTime("Close Time (HH:mm)");  
  
 System.out.println("\n\nSYSTEM: It seems that the create a Course system is not fully implemented, we will skip this part, it will introduced as 'null', sorry.\n\n");  
  
 Exam newExam = controller.createExam(examTitle, java.sql.Date.valueOf(String.valueOf(examDate)), Time.valueOf(String.valueOf(openTime)), Time.valueOf(String.valueOf(closeTime)), null);  
 if (newExam != null) {  
 System.out.println("Exam created successfully:");  
 System.out.println(newExam);  
 } else {  
 System.out.println("Failed to create exam.");  
 }  
  
 return false;  
 }  
  
 private static Time readTime(String message) {  
 String format = "HH:mm";  
 DateFormat dateFormat = new SimpleDateFormat(format);  
 dateFormat.setLenient(false); // Disallow lenient parsing  
  
 while (true) {  
 try {  
 String input = Console.readLine(message);  
 Date parsedDate = dateFormat.parse(input);  
 return new Time(parsedDate.getTime());  
 } catch (ParseException | java.text.ParseException e) {  
 System.out.println("\nInvalid time format. Please enter the time in the format: " + format + " (Hours : Minutes)");  
 System.out.println("Example: 10:00\n");  
 }  
 }  
 }  
  
 @Override  
 public String headline() {  
 return "Create Exam";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\CreateExamUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListCourseExamsAction.java

package eapli.base.app.backoffice.console.presentation.exam;  
  
import eapli.framework.actions.Action;  
  
public class ListCourseExamsAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new ListCourseExamsUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListCourseExamsAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListCourseExamsUI.java

package eapli.base.app.backoffice.console.presentation.exam;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.app.backoffice.console.presentation.course.CoursePrinter;  
import eapli.base.exammanagement.application.ListCourseExamsController;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
import java.util.Iterator;  
  
public class ListCourseExamsUI extends AbstractUI {  
  
 private final ListCourseExamsController controller = new ListCourseExamsController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().examRepository(), PersistenceContext.repositories().courseRepository());  
  
 @Override  
 public String headline() {  
 return "List all exams in a course";  
 }  
  
 @Override  
 protected boolean doShow() {  
 final Iterable<Course> courses = this.controller.listCourses();  
 final SelectWidget<Course> selector = new SelectWidget<>("Courses:", courses, new CoursePrinter());  
 selector.show();  
 final Course course = selector.selectedElement();  
 if (course != null) {  
 Iterable<Exam> courseExams = controller.listCourseExams(course);  
 for(Exam e : courseExams){  
 System.out.println("Exam id: " + e.identity() + "; exam course: " + e.getExamCourse().getCourseName());  
 }  
 }  
 return false;  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListCourseExamsUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListFutureExamsAction.java

package eapli.base.app.backoffice.console.presentation.exam;  
  
import eapli.framework.actions.Action;  
  
public class ListFutureExamsAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new ListFutureExamsUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListFutureExamsAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListFutureExamsUI.java

package eapli.base.app.backoffice.console.presentation.exam;  
  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.application.ListFutureExamsController;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.presentation.console.AbstractUI;  
  
import java.util.Optional;  
  
  
public class ListFutureExamsUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final ListFutureExamsController controller = new ListFutureExamsController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().examRepository(), PersistenceContext.repositories().studentRepository());  
  
  
 @Override  
 protected boolean doShow() {  
 Student student = controller.findStudentBySystemUser(myUser);  
 Iterable<Exam> futureExams = controller.listFutureExams(student);  
 for(Exam e : futureExams){  
 System.out.println("Exam id: " + e.identity() + "; exam course: " + e.getExamCourse().getCourseName());  
 }  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "List all future exams";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\exam\ListFutureExamsUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\AcceptRefuseInvitesAction.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.framework.actions.Action;  
  
public class AcceptRefuseInvitesAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new AcceptRefuseInvitesUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\AcceptRefuseInvitesAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\AcceptRefuseInvitesUI.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.meetingmanagement.application.AcceptRefuseInvitesController;  
import eapli.base.meetingmanagement.domain.\*;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
import java.text.ParseException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
public class AcceptRefuseInvitesUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final AcceptRefuseInvitesController controller = new AcceptRefuseInvitesController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().inviteRepository());  
  
 @Override  
 protected boolean doShow() {  
 int option;  
 do {  
// final Iterable<Invite> invites = this.controller.invitesReceived(myUser.username());  
// final SelectWidget<Invite> selector = new SelectWidget<>("Invites:", invites, new InvitePrinter());  
// selector.show();  
// final Invite invite = selector.selectedElement();  
 SimpleDateFormat simpleDateFormat = new SimpleDateFormat("hh:mm");  
 SimpleDateFormat simpleDateFormat2 = new SimpleDateFormat("dd-MM-yyyy");  
 Date time, date;  
 try {  
 time = simpleDateFormat.parse("10:30");  
 date=simpleDateFormat2.parse("25-06-2023");  
 } catch (ParseException e) {  
 throw new RuntimeException(e);  
 }  
 Meeting meeting = new Meeting(new MeetingTitle("meeting1"), myUser, new MeetingDuration(30), new MeetingDate(date), new MeetingTime(time));  
 Invite invite = new Invite(myUser, myUser, meeting);  
 System.out.println(invite.toString());  
 if (invite != null) {  
 System.out.println("Type \"1\" to accept the meeting invite or \"0\" to refuse it.");  
 int option2 = Console.readOption(0, 1, 0);  
 if (option2 == 1){  
 controller.acceptInvite(invite, myUser);  
 System.out.println("Meeting invite accepted.");  
 }  
 if (option2 == 0){  
 controller.refuseInvite(invite);  
 System.out.println("Meeting invite refused");  
 }  
 }  
  
 System.out.println("Type \"1\" to accept or refuse another meeting or \"0\" to leave.");  
 option = Console.readOption(0, 1, 0);  
 } while (option != 0);  
  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Accept/Refuse meeting invites";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\AcceptRefuseInvitesUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\CancelMeetingAction.java

package eapli.base.app.backoffice.console.presentation.meeting;  
import eapli.framework.actions.Action;  
  
  
public class CancelMeetingAction implements Action {  
 @Override  
 public boolean execute() {  
 return new CancelMeetingUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\CancelMeetingAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\CancelMeetingUI.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.meetingmanagement.application.CancelMeetingController;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
public class CancelMeetingUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final CancelMeetingController cancelMeetingController = new CancelMeetingController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().meetingRepository());  
  
 @Override  
 public String headline() {  
 return "Cancel a meeting";  
 }  
  
 @Override  
 protected boolean doShow() {  
 final Iterable<Meeting> meetings = cancelMeetingController.allMeetings();  
 final SelectWidget<Meeting> selector = new SelectWidget<>("Meetings:", meetings, new MeetingPrinter());  
 selector.show();  
 final Meeting meeting = selector.selectedElement();  
 if (meeting != null) {  
 cancelMeetingController.cancelMeeting(meeting.getMeetingTitle());  
 System.out.println("Meeting with ID " + meeting.identity() + " has been cancelled.");  
 }  
 return false;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\CancelMeetingUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\InvitePrinter.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.framework.visitor.Visitor;  
  
public class InvitePrinter implements Visitor<Invite> {  
  
 @Override  
 public void visit(Invite visitee) {  
 System.out.printf(visitee.toString());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\InvitePrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ListParticipantAction.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.framework.actions.Action;  
  
public class ListParticipantAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new ListParticipantUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ListParticipantAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ListParticipantUI.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.meetingmanagement.application.ListParticipantController;  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.InviteState;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.domain.MeetingTitle;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
import java.util.AbstractMap;  
import java.util.List;  
  
public class ListParticipantUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
 MeetingRepository meetingRepository = PersistenceContext.repositories().meetingRepository();  
 InviteRepository inviteRepository = PersistenceContext.repositories().inviteRepository();  
  
  
 private final ListParticipantController listParticipantController = new ListParticipantController(meetingRepository, inviteRepository);  
  
 @Override  
 public String headline() {  
 return "View participants of a meeting";  
 }  
  
 @Override  
 protected boolean doShow() {  
  
 final String meetingTitle = Console.readLine("Please enter the meeting title:");  
 Meeting meeting = listParticipantController.getMeetingById(new MeetingTitle(meetingTitle));  
  
 if (meeting == null) {  
 System.out.println("No meeting found with the given ID.");  
 return false;  
 }  
  
 Iterable<Invite> participantsStatus = listParticipantController.findInvitesByMeeting(meeting);  
  
 System.out.println("Participants and their status:");  
  
 while(participantsStatus.iterator().hasNext()){  
 Invite invite = participantsStatus.iterator().next();  
 System.out.println("User: " + invite.getReceiver().username().toString() + ", Status: " + invite.getState());  
 }  
  
 return false;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ListParticipantUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\MeetingPrinter.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.framework.visitor.Visitor;  
  
public class MeetingPrinter implements Visitor<Meeting> {  
  
 @Override  
 public void visit(Meeting visitee) {  
 System.out.printf("Meeting ID: %s, Date: %s, Time: %s, Duration: %s minutes, Status: %s\n",  
 visitee.identity(), visitee.getMeetingDate(), visitee.getMeetingTime(), visitee.getMeetingDuration(), visitee.getStatus());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\MeetingPrinter.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ScheduleMeetingAction.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.framework.actions.Action;  
  
public class ScheduleMeetingAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new ScheduleMeetingUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ScheduleMeetingAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ScheduleMeetingUI.java

package eapli.base.app.backoffice.console.presentation.meeting;  
  
import eapli.base.app.backoffice.console.presentation.authz.SystemUserPrinter;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.meetingmanagement.application.ScheduleMeetingController;  
import eapli.base.meetingmanagement.domain.\*;  
import eapli.base.usermanagement.application.ListUsersController;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.SelectWidget;  
  
import java.util.Date;  
  
public class ScheduleMeetingUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
  
 private final ListUsersController usersController = new ListUsersController();  
  
 private final ScheduleMeetingController scheduleMeetingController = new ScheduleMeetingController(  
 AuthzRegistry.authorizationService(), PersistenceContext.repositories().meetingRepository(), PersistenceContext.repositories().inviteRepository());  
  
 @Override  
 public String headline() {  
 return "Schedule a meeting";  
 }  
  
 @Override  
 protected boolean doShow() {  
 final String title = Console.readLine("Type the meeting title.");  
 final Date date = Console.readDate("Type the meeting date (dd-MM-yyyy):", "dd-MM-yyyy");  
 final Date time = Console.readDate("Type the meeting time (HH:mm):", "HH:mm");  
 final int duration = Console.readInteger("Type the meeting duration in minutes:");  
 Meeting meeting = scheduleMeetingController.saveMeeting(new MeetingTitle(title), myUser, new MeetingDuration(duration), new MeetingDate(date), new MeetingTime(time));  
  
 final Iterable<SystemUser> systemUsers = this.usersController.allUsers();  
 int option;  
 do{  
 final SelectWidget<SystemUser> selector = new SelectWidget<>("Users:", systemUsers, new SystemUserPrinter());  
 selector.show();  
 final SystemUser systemUser = selector.selectedElement();  
 if (systemUser != null) {  
 scheduleMeetingController.saveInvite(myUser, systemUser, meeting);  
 System.out.println("Invite sent to " + systemUser.username());  
 }  
 option=Console.readInteger("Type \"0\" to exit or any other button to invite another user.");  
 }while(option!=0);  
  
  
 return false;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\java\eapli\base\app\backoffice\console\presentation\meeting\ScheduleMeetingUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\resources\application.properties

#####################################################  
##  
## Base  
##  
#####################################################  
  
persistence.repositoryFactory=eapli.base.persistence.impl.jpa.JpaRepositoryFactory  
#persistence.repositoryFactory=eapli.base.persistence.impl.inmemory.InMemoryRepositoryFactory  
ui.menu.layout=horizontal  
persistence.persistenceUnit=eapli.base  
HighCaloriesDishLimit=300

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\resources\application.properties

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\resources\logback.xml

<?xml version="1.0" encoding="UTF-8" ?>  
  
<configuration scan="true" scanPeriod="3 seconds">  
  
 <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <pattern>  
 %d{HH:mm:ss.SSS} [%thread] %-5level %logger{32} - %msg%n  
 </pattern>  
 </encoder>  
 </appender>  
  
 <appender name="FILE"  
 class="ch.qos.logback.core.rolling.RollingFileAppender">  
 <File>logFile.log</File>  
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">  
 <FileNamePattern>  
 logFile.%d{yyyy-MM-dd\_HH-mm}.log.zip  
 </FileNamePattern>  
 </rollingPolicy>  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <Pattern>  
 %d{HH:mm:ss,SSS} [%thread] %-5level %logger{32} - %msg%n  
 </Pattern>  
 </encoder>  
 </appender>  
  
 <!-- LOG everything at INFO level -->  
 <root level="error">  
 <appender-ref ref="STDOUT" />  
 </root>  
  
 <!-- specific LOG level -->  
 <logger name="eapli.framework" level="debug" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
 <logger name="eapli.base" level="info" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
  
 <Logger name="org.hibernate" level="WARN">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.SQL" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.type" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.util.DTDEntityResolver" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
</configuration>

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\main\resources\logback.xml

### [File Begins] sem4pi-22-23-61-master\base.app.backoffice.console\src\test\java\eapli\base\app\other\console\BaseBackofficeTest.java

package eapli.base.app.other.console;  
  
/\*\*  
 \* Created by Nuno Bettencourt [NMB] on 03/04/16.  
 \*/  
public class BaseBackofficeTest {  
  
}

### [File Ends] sem4pi-22-23-61-master\base.app.backoffice.console\src\test\java\eapli\base\app\other\console\BaseBackofficeTest.java

### [File Begins] sem4pi-22-23-61-master\base.app.bootstrap\pom.xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
   
   
 <artifactId>base.app.bootstrap</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.app.bootstrap</name>  
  
 <url>http://maven.apache.org</url>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.core</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.bootstrappers</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.persistence.impl</artifactId>  
 <version>${project.parent.version}</version>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.app.common.console</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.app.bootstrap\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.app.bootstrap\src\main\java\eapli\base\app\bootstrap\BaseBootstrap.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.bootstrap;  
  
import eapli.base.app.common.console.BaseApplication;  
import eapli.base.clientusermanagement.application.eventhandlers.NewUserRegisteredFromSignupWatchDog;  
import eapli.base.clientusermanagement.domain.events.NewUserRegisteredFromSignupEvent;  
import eapli.base.clientusermanagement.domain.events.SignupAcceptedEvent;  
import eapli.base.infrastructure.bootstrapers.BaseBootstrapper;  
import eapli.base.infrastructure.bootstrapers.demo.BaseDemoBootstrapper;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.infrastructure.smoketests.BaseDemoSmokeTester;  
import eapli.base.usermanagement.application.eventhandlers.SignupAcceptedWatchDog;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.pubsub.EventDispatcher;  
import eapli.framework.util.ArrayPredicates;  
  
/\*\*  
 \* Base Bootstrapping data app  
 \*  
 \*/  
@SuppressWarnings("squid:S106")  
public final class BaseBootstrap extends BaseApplication {  
 /\*\*  
 \* avoid instantiation of this class.  
 \*/  
 private BaseBootstrap() {  
 }  
  
 private boolean isToBootstrapDemoData;  
 private boolean isToRunSampleE2E;  
  
 public static void main(final String[] args) {  
  
 AuthzRegistry.configure(PersistenceContext.repositories().users(), new BasePasswordPolicy(),  
 new PlainTextEncoder());  
  
 new BaseBootstrap().run(args);  
 }  
  
 @Override  
 protected void doMain(final String[] args) {  
 handleArgs(args);  
  
 System.out.println("\n\n------- MASTER DATA -------");  
 new BaseBootstrapper().execute();  
  
 if (isToBootstrapDemoData) {  
 System.out.println("\n\n------- DEMO DATA -------");  
 new BaseDemoBootstrapper().execute();  
 }  
 if (isToRunSampleE2E) {  
 System.out.println("\n\n------- BASIC SCENARIO -------");  
 new BaseDemoSmokeTester().execute();  
 }  
 }  
  
 private void handleArgs(final String[] args) {  
 isToRunSampleE2E = ArrayPredicates.contains(args, "-smoke:basic");  
 if (isToRunSampleE2E) {  
 isToBootstrapDemoData = true;  
 } else {  
 isToBootstrapDemoData = ArrayPredicates.contains(args, "-bootstrap:demo");  
 }  
 }  
  
 @Override  
 protected String appTitle() {  
 return "Bootstrapping Base data ";  
 }  
  
 @Override  
 protected String appGoodbye() {  
 return "Bootstrap data done.";  
 }  
  
 @SuppressWarnings("unchecked")  
 @Override  
 protected void doSetupEventHandlers(final EventDispatcher dispatcher) {  
 dispatcher.subscribe(new NewUserRegisteredFromSignupWatchDog(), NewUserRegisteredFromSignupEvent.class);  
 dispatcher.subscribe(new SignupAcceptedWatchDog(), SignupAcceptedEvent.class);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.bootstrap\src\main\java\eapli\base\app\bootstrap\BaseBootstrap.java

### [File Begins] sem4pi-22-23-61-master\base.app.bootstrap\src\main\resources\application.properties

#####################################################  
##  
## Base  
##  
#####################################################  
  
persistence.repositoryFactory=eapli.base.persistence.impl.jpa.JpaRepositoryFactory  
#persistence.repositoryFactory=eapli.base.persistence.impl.inmemory.InMemoryRepositoryFactory  
  
ui.menu.layout=horizontal  
  
persistence.persistenceUnit=eapli.base  
  
javax.persistence.schema-generation.database.action=drop-and-create  
  
HighCaloriesDishLimit=300

### [File Ends] sem4pi-22-23-61-master\base.app.bootstrap\src\main\resources\application.properties

### [File Begins] sem4pi-22-23-61-master\base.app.bootstrap\src\main\resources\logback.xml

<?xml version="1.0" encoding="UTF-8" ?>  
  
<configuration scan="true" scanPeriod="3 seconds">  
  
 <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <pattern>  
 %d{HH:mm:ss.SSS} [%thread] %-5level %logger{32} - %msg%n  
 </pattern>  
 </encoder>  
 </appender>  
  
 <appender name="FILE"  
 class="ch.qos.logback.core.rolling.RollingFileAppender">  
 <File>logFile.log</File>  
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">  
 <FileNamePattern>  
 logFile.%d{yyyy-MM-dd\_HH-mm}.log.zip  
 </FileNamePattern>  
 </rollingPolicy>  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <Pattern>  
 %d{HH:mm:ss,SSS} [%thread] %-5level %logger{32} - %msg%n  
 </Pattern>  
 </encoder>  
 </appender>  
  
 <!-- LOG everything at INFO level -->  
 <root level="error">  
 <appender-ref ref="STDOUT" />  
 </root>  
  
 <!-- specific LOG level -->  
 <logger name="eapli.framework" level="debug" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
 <logger name="eapli.base" level="info" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
  
 <Logger name="org.hibernate" level="WARN">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.SQL" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.type" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.util.DTDEntityResolver" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
</configuration>

### [File Ends] sem4pi-22-23-61-master\base.app.bootstrap\src\main\resources\logback.xml

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\pom.xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
  
 <artifactId>base.app.common.console</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.app.common.console</name>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 <slf4jVersion>1.7.20</slf4jVersion>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>3.8.1</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.core</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\BaseApplication.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.common.console;  
  
import org.apache.logging.log4j.LogManager;  
import org.apache.logging.log4j.Logger;  
  
import eapli.base.Application;  
import eapli.framework.infrastructure.pubsub.EventDispatcher;  
import eapli.framework.infrastructure.pubsub.impl.inprocess.service.InProcessPubSub;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra Sousa  
 \*/  
@SuppressWarnings("squid:S106")  
public abstract class BaseApplication {  
  
 // we are assuming we will always use the in process event  
 // dispatcher. check the Spring version of the Base project  
 // for an alternative  
 final EventDispatcher dispatcher = InProcessPubSub.dispatcher();  
  
 protected static final String SEPARATOR\_HR = "=====================================";  
 private static final Logger LOGGER = LogManager.getLogger(BaseApplication.class);  
  
 /\*\*  
 \* @param args  
 \* the command line arguments  
 \*/  
 public void run(final String[] args) {  
 printHeader();  
  
 try {  
 setupEventHandlers();  
  
 doMain(args);  
  
 printFooter();  
 } catch (final Exception e) {  
 System.out.println(  
 "Something unexpected has happened and the application will terminate. Please check the logs.\n");  
 LOGGER.error(e);  
 } finally {  
 clearEventHandlers();  
 }  
  
 // exiting the application, closing all threads  
 System.exit(0);  
 }  
  
 protected void printFooter() {  
 System.out.println("\n");  
 System.out.println(SEPARATOR\_HR);  
 System.out.println(appGoodbye());  
 System.out.println(SEPARATOR\_HR);  
 }  
  
 protected void printHeader() {  
 System.out.println(SEPARATOR\_HR);  
 System.out.println(appTitle() + " " + Application.VERSION);  
 System.out.println(Application.COPYRIGHT);  
 System.out.println(SEPARATOR\_HR);  
 }  
  
 private final void clearEventHandlers() {  
 try {  
 doClearEventHandlers(dispatcher);  
  
 dispatcher.shutdown();  
 } catch (final Exception e) {  
 LOGGER.error("Unable to cleanup event handlers", e);  
 }  
 }  
  
 private final void setupEventHandlers() {  
 doSetupEventHandlers(dispatcher);  
 }  
  
 protected abstract void doMain(final String[] args);  
  
 protected abstract String appTitle();  
  
 protected abstract String appGoodbye();  
  
 protected void doClearEventHandlers(final EventDispatcher dispatcher) {  
 // nothing to do  
 }  
  
 protected abstract void doSetupEventHandlers(EventDispatcher dispatcher);  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\BaseApplication.java

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\ChangePasswordUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.common.console.presentation.authz;  
  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.application.AuthenticationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
/\*\*  
 \* UI for user login action. Created by nuno on 21/03/16.  
 \*/  
@SuppressWarnings("squid:S106")  
public class ChangePasswordUI extends AbstractUI {  
  
 private final AuthenticationService authenticationService = AuthzRegistry.authenticationService();  
  
 @Override  
 protected boolean doShow() {  
 final String oldPassword = Console.readLine("Old Password:");  
 final String newPassword = Console.readLine("New Password:");  
  
 try {  
 if (this.authenticationService.changePassword(oldPassword, newPassword)) {  
 System.out.println("Password Successfuly changed");  
 return true;  
 } else {  
 System.out.println("Invalid authentication");  
 return false;  
 }  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 System.out.println("An error has occurred> " + e.getLocalizedMessage());  
 return false;  
 }  
 }  
  
 @Override  
 public String headline() {  
 return "Change Password";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\ChangePasswordUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LoginAction.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.common.console.presentation.authz;  
  
import eapli.base.infrastructure.authz.AuthenticationCredentialHandler;  
import eapli.framework.actions.Action;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
  
/\*\*  
 \* Menu action for user login. Created by nuno on 20/03/16.  
 \*/  
public class LoginAction implements Action {  
  
 private final Role onlyWithThis;  
  
 public LoginAction() {  
 onlyWithThis = null;  
 }  
  
 /\*\*  
 \*  
 \* @param onlyWithThis  
 \* only if the user has this specific action right will be  
 \* allowed to login  
 \*/  
 public LoginAction(final Role onlyWithThis) {  
 this.onlyWithThis = onlyWithThis;  
 }  
  
 @Override  
 public boolean execute() {  
 return new LoginUI(new AuthenticationCredentialHandler(), onlyWithThis).show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LoginAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LoginUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.common.console.presentation.authz;  
  
import eapli.base.infrastructure.authz.CredentialHandler;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
/\*\*   
 \* UI for user login action.   
 \*  
 \* @author nuno on 21/03/16.  
 \*/  
@SuppressWarnings("squid:S106")  
public class LoginUI extends AbstractUI {  
  
 private Role onlyWithThis;  
 private static final int DEFAULT\_MAX\_ATTEMPTS = 3;  
 private final int maxAttempts;  
  
 private final CredentialHandler credentialHandler;  
  
 public LoginUI(CredentialHandler credentialHandler) {  
 maxAttempts = DEFAULT\_MAX\_ATTEMPTS;  
 this.credentialHandler = credentialHandler;  
 }  
  
 public LoginUI(CredentialHandler credentialHandler, final Role onlyWithThis) {  
 this.onlyWithThis = onlyWithThis;  
 maxAttempts = DEFAULT\_MAX\_ATTEMPTS;  
 this.credentialHandler = credentialHandler;  
 }  
  
 public LoginUI(CredentialHandler credentialHandler, final Role onlyWithThis, final int maxAttempts) {  
 this.onlyWithThis = onlyWithThis;  
 this.maxAttempts = maxAttempts;  
 this.credentialHandler = credentialHandler;  
 }  
  
 public LoginUI(CredentialHandler credentialHandler, final int maxAttempts) {  
 this.maxAttempts = maxAttempts;  
 this.credentialHandler = credentialHandler;  
 }  
  
 @Override  
 protected boolean doShow() {  
 var attempt = 1;  
 while (attempt <= maxAttempts) {  
 final String userName = Console.readNonEmptyLine("Username:", "Please provide a username");  
 final String password = Console.readLine("Password:");  
  
 if (credentialHandler.authenticated(userName, password, onlyWithThis)) {  
 return true;  
 }  
 System.out.printf("Wrong username or password. You have %d attempts left.%n%n»»»»»»»»»%n",  
 maxAttempts - attempt);  
 attempt++;  
 }  
 System.out.println("Sorry, we are unable to authenticate you. Please contact your system admnistrator.");  
 return false;  
 }  
  
 @Override  
 public String headline() {  
 return "Login";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LoginUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LogoutAction.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.common.console.presentation.authz;  
  
import eapli.framework.actions.Action;  
  
/\*\*  
 \* Menu action for user logout. Created by nuno on 20/03/16.  
 \*/  
public class LogoutAction implements Action {  
  
 @Override  
 public boolean execute() {  
 // TODO call controller to execute logout  
 return false;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LogoutAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LogoutUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.common.console.presentation.authz;  
  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.presentation.console.AbstractUI;  
  
/\*\*  
 \*  
 \* @author mcn  
 \*/  
public class LogoutUI extends AbstractUI {  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 @Override  
 public String headline() {  
 return "Logout sucessful!!\n Make a new Login";  
 }  
  
 @Override  
 protected boolean doShow() {  
 authz.clearSession();  
 return true;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\LogoutUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\MyUserMenu.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.common.console.presentation.authz;  
  
import eapli.base.infrastructure.authz.AuthenticationCredentialHandler;  
import eapli.framework.actions.Actions;  
import eapli.framework.actions.menu.Menu;  
import eapli.framework.actions.menu.MenuItem;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
  
public class MyUserMenu extends Menu {  
  
 private static final String MENU\_TITLE = "My account >";  
  
 private static final int EXIT\_OPTION = 0;  
  
 // MY USER  
 private static final int CHANGE\_PASSWORD\_OPTION = 1;  
 private static final int LOGIN\_OPTION = 2;  
 private static final int LOGOUT\_OPTION = 3;  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 public MyUserMenu() {  
 super(MENU\_TITLE);  
 buildMyUserMenu(null);  
 }  
  
 public MyUserMenu(final Role onlyWithThis) {  
 super(MENU\_TITLE);  
 buildMyUserMenu(onlyWithThis);  
 }  
  
 private void buildMyUserMenu(final Role onlyWithThis) {  
 if (authz.hasSession()) {  
 addItem(MenuItem.of(CHANGE\_PASSWORD\_OPTION, "Change password", new ChangePasswordUI()::show));  
 addItem(MenuItem.of(LOGIN\_OPTION, "Change user", new LoginUI(new AuthenticationCredentialHandler(), onlyWithThis)::show));  
 addItem(MenuItem.of(LOGOUT\_OPTION, "Logout", new LogoutUI()::show));  
 } else {  
 addItem(MenuItem.of(LOGIN\_OPTION, "Login", new LoginUI(new AuthenticationCredentialHandler(), onlyWithThis)::show));  
 }  
  
 addItem(MenuItem.of(EXIT\_OPTION, "Return ", Actions.SUCCESS));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\main\java\eapli\base\app\common\console\presentation\authz\MyUserMenu.java

### [File Begins] sem4pi-22-23-61-master\base.app.common.console\src\test\java\eapli\base\consoleapp\common\AppTest.java

package eapli.base.consoleapp.common;  
  
import junit.framework.Test;  
import junit.framework.TestCase;  
import junit.framework.TestSuite;  
  
/\*\*  
 \* Unit test for simple App.  
 \*/  
public class AppTest extends TestCase {  
  
 /\*\*  
 \* @return the suite of tests being tested  
 \*/  
 public static Test suite() {  
 return new TestSuite(AppTest.class);  
 }  
  
 /\*\*  
 \* Create the test case  
 \*  
 \* @param testName name of the test case  
 \*/  
 public AppTest(String testName) {  
 super(testName);  
 }  
  
 /\*\*  
 \* Rigourous Test :-)  
 \*/  
 public void testApp() {  
 assertTrue(true);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.common.console\src\test\java\eapli\base\consoleapp\common\AppTest.java

### [File Begins] sem4pi-22-23-61-master\base.app.other.console\pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
   
 <artifactId>base.app.other.console</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.app.other.console</name>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 </properties>  
  
  
 <dependencies>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.app.common.console</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.persistence.impl</artifactId>  
 <version>${project.parent.version}</version>  
 <scope>runtime</scope>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.app.other.console\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.app.other.console\src\main\java\eapli\base\app\other\console\OtherApp.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.other.console;  
  
import eapli.base.app.common.console.presentation.authz.LoginAction;  
import eapli.base.app.other.console.presentation.MainMenu;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra Sousa  
 \*/  
@SuppressWarnings("squid:S106")  
public final class OtherApp {  
  
 /\*\*  
 \* Empty constructor is private to avoid instantiation of this class.  
 \*/  
 private OtherApp() {  
 }  
  
 public static void main(final String[] args) {  
 System.out.println("=====================================");  
 System.out.println("Base POS");  
 System.out.println("(C) 2016, 2017, 2018");  
 System.out.println("=====================================");  
  
 AuthzRegistry.configure(PersistenceContext.repositories().users(),  
 new BasePasswordPolicy(), new PlainTextEncoder());  
  
 // login and go to main menu  
 if (new LoginAction(BaseRoles.ADMIN).execute()) {  
 final MainMenu menu = new MainMenu();  
 menu.mainLoop();  
 }  
  
 // exiting the application, closing all threads  
 System.exit(0);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.other.console\src\main\java\eapli\base\app\other\console\OtherApp.java

### [File Begins] sem4pi-22-23-61-master\base.app.other.console\src\main\java\eapli\base\app\other\console\presentation\MainMenu.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.other.console.presentation;  
  
import eapli.base.app.common.console.presentation.authz.MyUserMenu;  
import eapli.base.Application;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.actions.Actions;  
import eapli.framework.actions.menu.Menu;  
import eapli.framework.actions.menu.MenuItem;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.ExitWithMessageAction;  
import eapli.framework.presentation.console.menu.HorizontalMenuRenderer;  
import eapli.framework.presentation.console.menu.MenuItemRenderer;  
import eapli.framework.presentation.console.menu.MenuRenderer;  
import eapli.framework.presentation.console.menu.VerticalMenuRenderer;  
  
/\*\*  
 \* TODO split this class in more specialized classes for each menu  
 \*  
 \* @author Paulo Gandra Sousa  
 \*/  
public class MainMenu extends AbstractUI {  
  
 private static final String SEPARATOR\_LABEL = "--------------";  
  
 private static final int EXIT\_OPTION = 0;  
  
 // MAIN MENU  
 private static final int MY\_USER\_OPTION = 1;  
 private static final int SALES\_OPTION = 7;  
  
 private static final int RECHARGE\_USER\_CARD\_OPTION = 1;  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 private final Menu menu;  
 private final MenuRenderer renderer;  
  
 public MainMenu() {  
 menu = buildMainMenu();  
 renderer = getRenderer(menu);  
 }  
  
 private MenuRenderer getRenderer(final Menu menu) {  
 final MenuRenderer theRenderer;  
 if (Application.settings().isMenuLayoutHorizontal()) {  
 theRenderer = new HorizontalMenuRenderer(menu, MenuItemRenderer.DEFAULT);  
 } else {  
 theRenderer = new VerticalMenuRenderer(menu, MenuItemRenderer.DEFAULT);  
 }  
 return theRenderer;  
 }  
  
 @Override  
 public boolean doShow() {  
 return renderer.render();  
 }  
  
 @Override  
 public boolean show() {  
 drawFormTitle();  
 return doShow();  
 }  
  
 @Override  
 public String headline() {  
  
 return authz.session().map(s -> "Base [ @" + s.authenticatedUser().identity() + " ]")  
 .orElse("Base [ ==Anonymous== ]");  
 }  
  
 private Menu buildMainMenu() {  
 final Menu mainMenu = new Menu();  
  
 final Menu myUserMenu = new MyUserMenu(BaseRoles.ADMIN);  
 mainMenu.addSubMenu(MY\_USER\_OPTION, myUserMenu);  
  
 if (!Application.settings().isMenuLayoutHorizontal()) {  
 mainMenu.addItem(MenuItem.separator(SEPARATOR\_LABEL));  
 }  
  
 if (authz.isAuthenticatedUserAuthorizedTo(BaseRoles.ADMIN)) {  
 final Menu cashierMenu = buildCashierMenu();  
 mainMenu.addSubMenu(SALES\_OPTION, cashierMenu);  
 }  
  
 if (!Application.settings().isMenuLayoutHorizontal()) {  
 mainMenu.addItem(MenuItem.separator(SEPARATOR\_LABEL));  
 }  
  
 mainMenu.addItem(EXIT\_OPTION, "Exit", new ExitWithMessageAction("Bye, Bye"));  
  
 return mainMenu;  
 }  
  
 private Menu buildCashierMenu() {  
 final Menu cashierMenu = new Menu("Sales >");  
  
 cashierMenu.addItem(EXIT\_OPTION, "Return", Actions.SUCCESS);  
  
 return cashierMenu;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.other.console\src\main\java\eapli\base\app\other\console\presentation\MainMenu.java

### [File Begins] sem4pi-22-23-61-master\base.app.other.console\src\main\resources\application.properties

#####################################################  
##  
## Base  
##  
#####################################################  
  
persistence.repositoryFactory=eapli.base.persistence.impl.jpa.JpaRepositoryFactory  
#persistence.repositoryFactory=eapli.base.persistence.impl.inmemory.InMemoryRepositoryFactory  
ui.menu.layout=horizontal  
persistence.persistenceUnit=eapli.base  
HighCaloriesDishLimit=300

### [File Ends] sem4pi-22-23-61-master\base.app.other.console\src\main\resources\application.properties

### [File Begins] sem4pi-22-23-61-master\base.app.other.console\src\main\resources\logback.xml

<?xml version="1.0" encoding="UTF-8" ?>  
  
<configuration scan="true" scanPeriod="3 seconds">  
  
 <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <pattern>  
 %d{HH:mm:ss.SSS} [%thread] %-5level %logger{32} - %msg%n  
 </pattern>  
 </encoder>  
 </appender>  
  
 <appender name="FILE"  
 class="ch.qos.logback.core.rolling.RollingFileAppender">  
 <File>logFile.log</File>  
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">  
 <FileNamePattern>  
 logFile.%d{yyyy-MM-dd\_HH-mm}.log.zip  
 </FileNamePattern>  
 </rollingPolicy>  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <Pattern>  
 %d{HH:mm:ss,SSS} [%thread] %-5level %logger{32} - %msg%n  
 </Pattern>  
 </encoder>  
 </appender>  
  
 <!-- LOG everything at INFO level -->  
 <root level="error">  
 <appender-ref ref="STDOUT" />  
 </root>  
  
 <!-- specific LOG level -->  
 <logger name="eapli.framework" level="debug" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
 <logger name="eapli.base" level="info" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
  
 <Logger name="org.hibernate" level="WARN">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.SQL" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.type" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.util.DTDEntityResolver" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
</configuration>

### [File Ends] sem4pi-22-23-61-master\base.app.other.console\src\main\resources\logback.xml

### [File Begins] sem4pi-22-23-61-master\base.app.other.console\src\test\java\eapli\base\app\other\console\OtherAppTest.java

package eapli.base.app.other.console;  
  
/\*\*  
 \* Created by Nuno Bettencourt [NMB] on 03/04/16.  
 \*/  
public class OtherAppTest {  
  
}

### [File Ends] sem4pi-22-23-61-master\base.app.other.console\src\test\java\eapli\base\app\other\console\OtherAppTest.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\pom.xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
   
 <artifactId>base.app.user.console</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.app.user.console</name>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.app.common.console</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.core</artifactId>  
 <version>${project.parent.version}</version>  
 <type>jar</type>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.persistence.impl</artifactId>  
 <version>${project.parent.version}</version>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.1</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\BaseUserApp.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.user.console;  
  
import eapli.base.app.user.console.presentation.FrontMenu;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
  
/\*\*  
 \* Base User App.  
 \*/  
@SuppressWarnings("squid:S106")  
public final class BaseUserApp {  
  
 /\*\*  
 \* Empty constructor is private to avoid instantiation of this class.  
 \*/  
 private BaseUserApp() {  
 }  
  
 public static void main(final String[] args) {  
 System.out.println("=====================================");  
 System.out.println("Base User App");  
 System.out.println("(C) 2016 - 2019");  
 System.out.println("=====================================");  
  
 AuthzRegistry.configure(PersistenceContext.repositories().users(),  
 new BasePasswordPolicy(), new PlainTextEncoder());  
  
 new FrontMenu().show();  
  
 // exiting the application, closing all threads  
 System.exit(0);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\BaseUserApp.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\ClientUserBaseUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.user.console.presentation;  
  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.presentation.console.AbstractUI;  
  
/\*\*  
 \*  
 \* @author mcn  
 \*/  
@SuppressWarnings("squid:S106")  
public abstract class ClientUserBaseUI extends AbstractUI {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 @Override  
 public String headline() {  
  
 return authz.session().map(s -> "Base [ @" + s.authenticatedUser().identity() + " ] ")  
 .orElse("Base [ ==Anonymous== ]");  
 }  
  
 @Override  
 protected void drawFormTitle(final String title) {  
 final String titleBorder = BORDER.substring(0, 2) + " " + title;  
 System.out.println(titleBorder);  
 drawFormBorder();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\ClientUserBaseUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\FrontMenu.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.user.console.presentation;  
  
import eapli.base.app.common.console.presentation.authz.LoginUI;  
import eapli.base.app.user.console.presentation.myuser.SignupRequestAction;  
import eapli.base.infrastructure.authz.AuthenticationCredentialHandler;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.actions.ChainedAction;  
import eapli.framework.actions.menu.Menu;  
import eapli.framework.presentation.console.AbstractUI;  
import eapli.framework.presentation.console.ExitWithMessageAction;  
import eapli.framework.presentation.console.menu.MenuItemRenderer;  
import eapli.framework.presentation.console.menu.MenuRenderer;  
import eapli.framework.presentation.console.menu.VerticalMenuRenderer;  
  
/\*\*  
 \* @author Paulo Gandra Sousa  
 \*/  
public class FrontMenu extends AbstractUI {  
  
 private static final int EXIT\_OPTION = 0;  
  
 private static final int LOGIN\_OPTION = 1;  
 private static final int SIGNUP\_OPTION = 2;  
  
 @Override  
 public boolean show() {  
 drawFormTitle();  
 return doShow();  
 }  
  
 /\*\*  
 \* @return true if the user selected the exit option  
 \*/  
 @Override  
 public boolean doShow() {  
 final Menu menu = new Menu();  
 menu.addItem(LOGIN\_OPTION, "Login", new ChainedAction(new LoginUI(new AuthenticationCredentialHandler(),   
 BaseRoles.CLIENT\_USER)::show, () -> {  
 new MainMenu().mainLoop();  
 return true;  
 }));  
 //TODO: instead of leaving the app, return to the main menu again  
 menu.addItem(SIGNUP\_OPTION, "Sign up", new SignupRequestAction());  
 menu.addItem(EXIT\_OPTION, "Exit", new ExitWithMessageAction("Bye, Bye"));  
  
 final MenuRenderer renderer = new VerticalMenuRenderer(menu, MenuItemRenderer.DEFAULT);  
 return renderer.render();  
 }  
  
 @Override  
 public String headline() {  
 return "Base";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\FrontMenu.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\MainMenu.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.app.user.console.presentation;  
  
import eapli.base.app.common.console.presentation.authz.MyUserMenu;  
import eapli.framework.actions.menu.Menu;  
import eapli.framework.actions.menu.MenuItem;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.presentation.console.ExitWithMessageAction;  
import eapli.framework.presentation.console.menu.MenuItemRenderer;  
import eapli.framework.presentation.console.menu.MenuRenderer;  
import eapli.framework.presentation.console.menu.VerticalMenuRenderer;  
  
/\*\*  
 \* @author Paulo Gandra Sousa  
 \*/  
class MainMenu extends ClientUserBaseUI {  
  
 private static final String SEPARATOR\_LABEL = "--------------";  
  
 private static final String RETURN = "Return ";  
  
 private static final String NOT\_IMPLEMENTED\_YET = "Not implemented yet";  
  
 // EXIT  
 private static final int EXIT\_OPTION = 0;  
  
 // MAIN MENU  
 private static final int MY\_USER\_OPTION = 1;  
 private static final int ACCOUNT\_OPTION = 3;  
 private static final int SETTINGS\_OPTION = 4;  
  
 //EXAMS  
 private static final int EXAMS\_OPTION = 2;  
  
 // BOOKINGS MENU  
 private static final int BOOK\_A\_MEAL\_OPTION = 2;  
 private static final int LIST\_MY\_BOOKINGS\_OPTION = 3;  
  
 // ACCOUNT MENU  
 private static final int LIST\_MOVEMENTS\_OPTION = 1;  
  
 // SETTINGS  
 private static final int SET\_USER\_ALERT\_LIMIT\_OPTION = 1;  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 @Override  
 public boolean show() {  
 drawFormTitle();  
 return doShow();  
 }  
  
 /\*\*  
 \* @return true if the user selected the exit option  
 \*/  
 @Override  
 public boolean doShow() {  
 final Menu menu = buildMainMenu();  
 final MenuRenderer renderer = new VerticalMenuRenderer(menu, MenuItemRenderer.DEFAULT);  
 return renderer.render();  
 }  
  
 private Menu buildMainMenu() {  
 final Menu mainMenu = new Menu();  
  
 final Menu myUserMenu = new MyUserMenu();  
 mainMenu.addSubMenu(MY\_USER\_OPTION, myUserMenu);  
  
 mainMenu.addItem(MenuItem.separator(SEPARATOR\_LABEL));  
  
 mainMenu.addItem(EXIT\_OPTION, "Exit", new ExitWithMessageAction("Bye, Bye"));  
  
 return mainMenu;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\MainMenu.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\myuser\SignupRequestAction.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.user.console.presentation.myuser;  
  
import eapli.framework.actions.Action;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt  
 \*/  
public class SignupRequestAction implements Action {  
  
 @Override  
 public boolean execute() {  
 return new SignupRequestUI().show();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\myuser\SignupRequestAction.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\myuser\SignupRequestUI.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.user.console.presentation.myuser;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import eapli.base.myclientuser.application.SignupController;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.io.util.Console;  
import eapli.framework.presentation.console.AbstractUI;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt  
 \*/  
@SuppressWarnings("squid:S106")  
public class SignupRequestUI extends AbstractUI {  
 private static final Logger LOGGER = LoggerFactory.getLogger(SignupRequestUI.class);  
  
 private final SignupController theController = new SignupController();  
  
 @Override  
 protected boolean doShow() {  
 final UserDataWidget userData = new UserDataWidget();  
  
 userData.show();  
  
 final String mecanographicNumber = Console.readLine("Mecanographic Number");  
  
 try {  
 this.theController.signup(userData.username(), userData.password(),  
 userData.firstName(), userData.lastName(), userData.email(),  
 mecanographicNumber);  
 } catch (final IntegrityViolationException | ConcurrencyException e) {  
 LOGGER.error("Error performing the operation", e);  
 System.out.println(  
 "Unfortunatelly there was an unexpected error in the application. Please try again and if the problem persists, contact your system admnistrator.");  
 }  
  
 return true;  
 }  
  
 @Override  
 public String headline() {  
 return "Sign Up";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\myuser\SignupRequestUI.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\myuser\UserDataWidget.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.app.user.console.presentation.myuser;  
  
import eapli.framework.io.util.Console;  
  
/\*\*  
 \* TODO move to console.common to allow reuse from both backoffice and UtenteApp  
 \*  
 \* widget for reading user data Jorge Santos ajs@isp.ipp.pt  
 \*/  
class UserDataWidget {  
  
 private String username;  
 private String password;  
 private String firstName;  
 private String lastName;  
 private String email;  
  
 public void show() {  
 this.username = Console.readLine("Username");  
 this.password = Console.readLine("Password");  
 this.firstName = Console.readLine("First Name");  
 this.lastName = Console.readLine("Last Name");  
 this.email = Console.readLine("E-Mail");  
 }  
  
 public String username() {  
 return this.username;  
 }  
  
 public String password() {  
 return this.password;  
 }  
  
 public String firstName() {  
 return this.firstName;  
 }  
  
 public String lastName() {  
 return this.lastName;  
 }  
  
 public String email() {  
 return this.email;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\java\eapli\base\app\user\console\presentation\myuser\UserDataWidget.java

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\resources\application.properties

#####################################################  
##  
## Base  
##  
#####################################################  
  
persistence.repositoryFactory=eapli.base.persistence.impl.jpa.JpaRepositoryFactory  
#persistence.repositoryFactory=eapli.base.persistence.impl.inmemory.InMemoryRepositoryFactory  
persistence.persistenceUnit=eapli.base  
ui.menu.layout=vertical  
HighCaloriesDishLimit=300

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\resources\application.properties

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\main\resources\logback.xml

<?xml version="1.0" encoding="UTF-8" ?>  
  
<configuration scan="true" scanPeriod="3 seconds">  
  
 <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <pattern>  
 %d{HH:mm:ss.SSS} [%thread] %-5level %logger{32} - %msg%n  
 </pattern>  
 </encoder>  
 </appender>  
  
 <appender name="FILE"  
 class="ch.qos.logback.core.rolling.RollingFileAppender">  
 <File>logFile.log</File>  
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">  
 <FileNamePattern>  
 logFile.%d{yyyy-MM-dd\_HH-mm}.log.zip  
 </FileNamePattern>  
 </rollingPolicy>  
 <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">  
 <Pattern>  
 %d{HH:mm:ss,SSS} [%thread] %-5level %logger{32} - %msg%n  
 </Pattern>  
 </encoder>  
 </appender>  
  
 <!-- LOG everything at INFO level -->  
 <root level="error">  
 <appender-ref ref="STDOUT" />  
 </root>  
  
 <!-- specific LOG level -->  
 <logger name="eapli.framework" level="debug" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
 <logger name="eapli.base" level="info" additivity="false">  
 <appender-ref ref="STDOUT" />  
 </logger>  
  
 <Logger name="org.hibernate" level="WARN">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.SQL" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.type" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
 <Logger name="org.hibernate.util.DTDEntityResolver" level="error">  
 <appender-ref ref="STDOUT" />  
 </Logger>  
</configuration>

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\main\resources\logback.xml

### [File Begins] sem4pi-22-23-61-master\base.app.user.console\src\test\java\eapli\base\utente\consoleapp\AppTest.java

package eapli.base.utente.consoleapp;  
  
import junit.framework.Test;  
import junit.framework.TestCase;  
import junit.framework.TestSuite;  
  
/\*\*  
 \* Unit test for simple App.  
 \*/  
public class AppTest extends TestCase {  
  
 /\*\*  
 \* Create the test case  
 \*  
 \* @param testName name of the test case  
 \*/  
 public AppTest(String testName) {  
 super(testName);  
 }  
  
 /\*\*  
 \* @return the suite of tests being tested  
 \*/  
 public static Test suite() {  
 return new TestSuite(AppTest.class);  
 }  
  
 /\*\*  
 \* Rigourous Test :-)  
 \*/  
 public void testApp() {  
 assertTrue(true);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.app.user.console\src\test\java\eapli\base\utente\consoleapp\AppTest.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\pom.xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
   
 <artifactId>base.bootstrappers</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.bootstrapper</name>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.12</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.core</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\BaseBootstrapper.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.infrastructure.bootstrapers;  
  
import eapli.base.Classe.domain.\*;  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.base.Course.Domain.\*;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.SharedBoard.domain.NumberofColumns;  
import eapli.base.SharedBoard.domain.NumberofRows;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.Student\_Teacher.Date\_Of\_Birth;  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.Student\_Teacher.Tax\_Payer\_Number;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.domain.ExamDate;  
import eapli.base.exammanagement.domain.ExamTime;  
import eapli.base.exammanagement.domain.ExamTitle;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.base.usermanagement.domain.UserBuilderHelper;  
import eapli.framework.actions.Action;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.application.AuthenticationService;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
import eapli.framework.infrastructure.authz.domain.repositories.UserRepository;  
import eapli.framework.strings.util.Strings;  
import eapli.framework.validations.Invariants;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import javax.persistence.Persistence;  
import java.text.ParseException;  
import java.text.SimpleDateFormat;  
import java.time.LocalDate;  
import java.time.LocalTime;  
import java.util.Date;  
  
/\*\*  
 \* Base Bootstrapping data app  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*  
 \* Classe destinada a criar um utilizador pelo bootstrap.  
 \*/  
@SuppressWarnings("squid:S106")  
public class BaseBootstrapper implements Action {  
 private static final Logger LOGGER = LoggerFactory.getLogger(BaseBootstrapper.class);  
  
 private static final String ADMIN = "admin";  
 private static final String ADMIN\_PWD = "Password1";  
 private static final String USER = "user";  
 private static final String USER\_PWD = "userA1";  
 private static final String MANAGER = "manager";  
 private static final String MANAGER\_PWD = "managerA1";  
 private static final String TEACHER = "teacher";  
 private static final String TEACHER\_PWD = "teacherA1";  
 private static final String STUDENT = "student";  
 private static final String STUDENT\_PWD = "studentA1";  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final AuthenticationService authenticationService = AuthzRegistry.authenticationService();  
 private final UserRepository userRepository = PersistenceContext.repositories().users();  
 private final TeacherRepository teacherRepository = PersistenceContext.repositories().teacherRepository();  
 private final StudentRepository studentRepository = PersistenceContext.repositories().studentRepository();  
 private final CourseRepository courseRepository = PersistenceContext.repositories().courseRepository();  
 private final ExamRepository examRepository = PersistenceContext.repositories().examRepository();  
 private final ClassRepository classeRepository = PersistenceContext.repositories().classRepository();  
 private final SharedBoardRepository sharedBoardRepository = PersistenceContext.repositories().sharedBoardRepository();  
  
 private SystemUser user1;  
 private SystemUser user2;  
 private Teacher teacher;  
  
 @Override  
 public boolean execute() {  
 // declare bootstrap actions  
 final Action[] actions = { new MasterUsersBootstrapper(), };  
  
 registerAdmin();  
 registerUser();  
 registerManager();  
 registerTeacher();  
 registerStudent();  
 registerSharedBoard();  
 try {  
 registerCourseExams();  
 registerClass1();  
 } catch (ParseException e) {  
 throw new RuntimeException(e);  
 }  
 authenticateForBootstrapping();  
  
 // execute all bootstrapping  
 boolean ret = true;  
 for (final Action boot : actions) {  
 System.out.println("Bootstrapping " + nameOfEntity(boot) + "...");  
 ret &= boot.execute();  
 }  
 return ret;  
 }  
  
 /\*\*  
 \* register a power user directly in the persistence layer as we need to  
 \* circumvent authorisations in the Application Layer  
 \*/  
 private boolean registerUser() {  
 final SystemUserBuilder userBuilder = UserBuilderHelper.builder();  
 userBuilder.withUsername(USER).withPassword(USER\_PWD).withName("Joao", "Martins")  
 .withEmail("joaomartins@email.org").withRoles(BaseRoles.CLIENT\_USER);  
 final SystemUser newUser = userBuilder.build();  
  
 SystemUser poweruser;  
 try {  
 poweruser = userRepository.save(newUser);  
 assert poweruser != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", newUser.username());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
  
 private boolean registerAdmin() {  
 final SystemUserBuilder userBuilder = UserBuilderHelper.builder();  
 userBuilder.withUsername(ADMIN).withPassword(ADMIN\_PWD).withName("Developer", "Administrator")  
 .withEmail("admin@email.org").withRoles(BaseRoles.ADMIN);  
 final SystemUser newUser = userBuilder.build();  
  
 SystemUser poweruser;  
 try {  
 poweruser = userRepository.save(newUser);  
 assert poweruser != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", newUser.username());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
  
 private boolean registerManager() {  
 final SystemUserBuilder userBuilder = UserBuilderHelper.builder();  
 userBuilder.withUsername(MANAGER).withPassword(MANAGER\_PWD).withName("Main", "Manager")  
 .withEmail("mainmanager@email.org").withRoles(BaseRoles.MANAGER);  
 final SystemUser newUser = userBuilder.build();  
  
 SystemUser manager;  
 try {  
 manager = userRepository.save(newUser);  
 assert manager != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", newUser.username());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
  
 private boolean registerTeacher() {  
 LocalDate date = LocalDate.of(1990,10,10);  
  
 final SystemUserBuilder userBuilder = UserBuilderHelper.builder();  
 userBuilder.withUsername(TEACHER).withPassword(TEACHER\_PWD).withName("Main", "Teacher")  
 .withEmail("mainteacher@email.org").withRoles(BaseRoles.TEACHER);  
 final SystemUser newUser = userBuilder.build();  
  
  
 teacher=new Teacher(newUser, new Acronym("ABC"), new Tax\_Payer\_Number("123456789"), new Date\_Of\_Birth(date));  
 try {  
 user1 = userRepository.save(newUser);  
 teacherRepository.save(teacher);  
 assert user1 != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", newUser.username());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
  
 private boolean registerStudent() {  
 LocalDate date = LocalDate.of(2002,10,10);  
  
 final SystemUserBuilder userBuilder = UserBuilderHelper.builder();  
 userBuilder.withUsername(STUDENT).withPassword(STUDENT\_PWD).withName("Main", "Student")  
 .withEmail("mainstudent@email.org").withRoles(BaseRoles.STUDENT);  
 final SystemUser newUser = userBuilder.build();  
  
 try {  
 user2 = userRepository.save(newUser);  
 studentRepository.save(new Student(newUser, new MechanographicNumber("1200801"), new Tax\_Payer\_Number("123456789"), new Date\_Of\_Birth(date)));  
 assert user2 != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", newUser.username());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
  
 private boolean registerSharedBoard(){  
 SharedBoard sharedBoard = new SharedBoard(user1, new Shared\_Board\_Title("title"), new NumberofRows(10), new NumberofColumns(10));  
 sharedBoard.shareBoard(user1, user2, SharedBoard.AccessType.WRITE);  
  
 sharedBoardRepository.save(sharedBoard);  
 return false;  
 }  
  
 private boolean registerCourseExams() throws ParseException {  
  
// EntityManagerFactory emf = Persistence.createEntityManagerFactory(Application.settings().getPersistenceUnitName());  
// EntityManager em = emf.createEntityManager();  
 Course course = new Course(new Course\_Name("INF"), new Maximum\_Number\_Of\_Students(200), new Minimum\_Number\_Of\_Students(20), new Small\_Textual\_Description("desc"), teacher);  
// em.getTransaction().begin();  
// em.persist(course);  
// em.getTransaction().commit();  
// em.close();  
// emf.close();  
  
 try {  
 course = courseRepository.save(course);  
 assert course != null;  
 registerExam1(course);  
 registerExam2(course);  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", course.getCourseName());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
  
 private boolean registerExam1(Course course) throws ParseException {  
 SimpleDateFormat simpleDateFormat = new SimpleDateFormat("hh:mm");  
 Date open = simpleDateFormat.parse("09:30");  
 Date close = simpleDateFormat.parse("10:30");  
 SimpleDateFormat simpleDateFormat2 = new SimpleDateFormat("dd-MM-yyyy");  
 Date date = simpleDateFormat2.parse("10-08-2023");  
  
 Exam exam = new Exam(new ExamTitle("exam\_title1"), new ExamDate(date), new ExamTime(open, close), course); //after current time  
  
  
 try {  
 exam = examRepository.save(exam);  
 assert exam != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", exam.identity());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
  
 }  
  
 private boolean registerExam2(Course course) throws ParseException {  
 SimpleDateFormat simpleDateFormat = new SimpleDateFormat("hh:mm");  
 Date open = simpleDateFormat.parse("09:30");  
 Date close = simpleDateFormat.parse("10:30");  
 SimpleDateFormat simpleDateFormat2 = new SimpleDateFormat("dd-MM-yyyy");  
 Date date2 = simpleDateFormat2.parse("10-05-2023");  
  
 Exam exam2 = new Exam(new ExamTitle("exam\_title2"), new ExamDate(date2), new ExamTime(open, close), course); //before current time  
 try {  
 exam2 = examRepository.save(exam2);  
 assert exam2 != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", exam2.identity());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
 private boolean registerClass1() throws ParseException {  
  
 LocalTime startTime = LocalTime.parse("09:00");  
 LocalTime finishTime = LocalTime.parse("10:30");  
  
 LocalDate date = LocalDate.parse("10-06-2023");  
 DayOfWeek dayOfWeek = new DayOfWeek(1);  
  
 Classe classe = new Classe(new Classe\_Title("Class 1"), new Classe\_Start\_Date(date), new Classe\_Finish\_Date(date),  
 new Classe\_Start\_Time(startTime), new Classe\_Finish\_Time(finishTime), dayOfWeek,  
 new Acronym("C1"));  
  
 try {  
 classe = classeRepository.save(classe);  
 assert classe != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
   
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", classe.identity());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
 private boolean registerClass2() throws ParseException {  
  
 LocalTime startTime = LocalTime.parse("11:00");  
 LocalTime finishTime = LocalTime.parse("12:30");  
  
 LocalDate date = LocalDate.parse("11-06-2023");  
 DayOfWeek dayOfWeek = new DayOfWeek(2);  
  
 Classe classe = new Classe(new Classe\_Title("Class 2"), new Classe\_Start\_Date(date), new Classe\_Finish\_Date(date),  
 new Classe\_Start\_Time(startTime), new Classe\_Finish\_Time(finishTime), dayOfWeek,  
 new Acronym("C2"));  
  
 try {  
 classe = classeRepository.save(classe);  
 assert classe != null;  
 return true;  
 } catch (ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated class  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", classe.identity());  
 LOGGER.trace("Assuming existing record", e);  
 return false;  
 }  
 }  
  
  
  
 /\*\*  
 \* authenticate a super user to be able to register new users  
 \*  
 \*/  
 protected void authenticateForBootstrapping() {  
 authenticationService.authenticate(MANAGER, MANAGER\_PWD);  
 Invariants.ensure(authz.hasSession());  
 }  
  
 private String nameOfEntity(final Action boot) {  
 final String name = boot.getClass().getSimpleName();  
 return Strings.left(name, name.length() - "Bootstrapper".length());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\BaseBootstrapper.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\MasterUsersBootstrapper.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.bootstrapers;  
  
import java.util.HashSet;  
import java.util.Set;  
  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.actions.Action;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
  
/\*\*  
 \* @author Paulo Gandra Sousa  
 \* Edited by João Cruz  
 \*  
 \* Classe destinada a criar um utilizador para apresentação do projeto em forma de bootstrapper.  
 \*/  
public class MasterUsersBootstrapper extends UsersBootstrapperBase implements Action {  
  
 @Override  
 public boolean execute() {  
 registerAdmin("Desenvolvedor", TestDataConstants.PASSWORD1, "Desenvolvedor", "LAPR4", "desenvolvedor.lapr4@local"); // Administrador  
 registerManager("JoaoSimoes", TestDataConstants.PASSWORD1, "Joao", "Simoes", "joao.simoes@local"); // Manager 1  
 registerManager("HenriqueMedina", TestDataConstants.PASSWORD1, "Henrique", "Medina", "henrique.medina@local"); // Manager 2  
 registerTeacher("LuisFabiano", TestDataConstants.PASSWORD1, "Luis", "Fabiano", "luis.fabiano@local"); // Teacher 1  
 registerTeacher("AlexandreMatias", TestDataConstants.PASSWORD1, "Alexandre", "Matias", "alexandre.matias@local"); // Teacher 2  
 registerStudent("RicardoFilipe", TestDataConstants.PASSWORD1, "Ricardo", "Filipe", "ricardo.filipe@local"); // Student 1  
 registerStudent("JoseAfonso", TestDataConstants.PASSWORD1, "Jose", "Afonso", "jose.afonso@local"); // Student 2  
 return true;  
 }  
  
 /\*\*  
 \*  
 \*/  
 private void registerAdmin(final String username, final String password, final String firstName, final String lastName, final String email) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.ADMIN);  
  
 registerUser(username, password, firstName, lastName, email, roles);  
 }  
  
 private void registerManager(final String username, final String password, final String firstName, final String lastName, final String email) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.MANAGER);  
  
 registerUser(username, password, firstName, lastName, email, roles);  
 }  
  
 private void registerTeacher(final String username, final String password, final String firstName, final String lastName, final String email) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.TEACHER);  
  
 registerUser(username, password, firstName, lastName, email, roles);  
 }  
  
 private void registerStudent(final String username, final String password, final String firstName, final String lastName, final String email) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.STUDENT);  
  
 registerUser(username, password, firstName, lastName, email, roles);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\MasterUsersBootstrapper.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\TestDataConstants.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.bootstrapers;  
  
import java.util.Calendar;  
  
import eapli.framework.time.util.Calendars;  
  
public final class TestDataConstants {  
  
 public static final String USER\_TEST1 = "user1";  
 public static final String MANAGER\_TEST1 = "manager1";  
 public static final String TEACHER\_TEST1 = "teacher1";  
 public static final String STUDENT\_TEST1 = "student1";  
  
 @SuppressWarnings("squid:S2068")  
 public static final String PASSWORD1 = "Password1";  
  
 @SuppressWarnings("squid:S2885")  
 public static final Calendar DATE\_TO\_BOOK = Calendars.of(2017, 12, 01);  
  
 private TestDataConstants() {  
 // ensure utility  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\TestDataConstants.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\UsersBootstrapperBase.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.bootstrapers;  
  
import java.util.Set;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import eapli.base.usermanagement.application.AddUserController;  
import eapli.base.usermanagement.application.ListUsersController;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
/\*\*  
 \* Classe destinada a listar os utilizadores.  
 \*/  
  
public class UsersBootstrapperBase {  
 private static final Logger LOGGER = LoggerFactory.getLogger(UsersBootstrapperBase.class);  
  
 final AddUserController userController = new AddUserController();  
 final ListUsersController listUserController = new ListUsersController();  
  
 public UsersBootstrapperBase() {  
 super();  
 }  
  
 /\*\*  
 \* @param username  
 \* @param password  
 \* @param firstName  
 \* @param lastName  
 \* @param email  
 \* @param roles  
 \*/  
 protected SystemUser registerUser(final String username, final String password, final String firstName,  
 final String lastName, final String email, final Set<Role> roles) {  
 SystemUser u = null;  
 try {  
 u = userController.addUser(username, password, firstName, lastName, email, roles);  
 LOGGER.debug("»»» %s", username);  
 } catch (final IntegrityViolationException | ConcurrencyException e) {  
 // assuming it is just a primary key violation due to the tentative  
 // of inserting a duplicated user. let's just lookup that user  
 u = listUserController.find(Username.valueOf(username)).orElseThrow(() -> e);  
 }  
 return u;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\UsersBootstrapperBase.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\demo\BackofficeUsersBootstrapper.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.bootstrapers.demo;  
  
import java.util.HashSet;  
import java.util.Set;  
  
import eapli.base.infrastructure.bootstrapers.UsersBootstrapperBase;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.actions.Action;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
  
/\*\*  
 \* @author Paulo Gandra Sousa  
 \*/  
public class BackofficeUsersBootstrapper extends UsersBootstrapperBase implements Action {  
  
 @SuppressWarnings("squid:S2068")  
 private static final String PASSWORD1 = "Password1";  
  
 @Override  
 public boolean execute() {  
 registerStudent("student", PASSWORD1, "Johny", "Cash", "johny.doe@emai.l.com");  
 registerTeacher("teacher", PASSWORD1, "Oven", "Stove", "Oven.and.stove@emai.l.com");  
 registerManager("manager", PASSWORD1, "Master", "Chef", "master.chef@emai.l.com");  
 return true;  
 }  
  
 private void registerStudent(final String username, final String password,  
 final String firstName, final String lastName, final String email) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.STUDENT);  
  
 registerUser(username, password, firstName, lastName, email, roles);  
 }  
  
 private void registerTeacher(final String username, final String password,  
 final String firstName, final String lastName, final String email) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.TEACHER);  
  
 registerUser(username, password, firstName, lastName, email, roles);  
 }  
  
 private void registerManager(final String username, final String password,  
 final String firstName, final String lastName, final String email) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.MANAGER);  
  
 registerUser(username, password, firstName, lastName, email, roles);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\demo\BackofficeUsersBootstrapper.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\demo\BaseDemoBootstrapper.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.infrastructure.bootstrapers.demo;  
  
import eapli.base.infrastructure.bootstrapers.BaseBootstrapper;  
import eapli.framework.actions.Action;  
import eapli.framework.infrastructure.authz.application.AuthenticationService;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.strings.util.Strings;  
import eapli.framework.validations.Invariants;  
  
/\*\*  
 \* Base Bootstrapping data app  
 \*  
 \* @todo avoid duplication with {@link BaseBootstrapper}  
 \* @author Paulo Gandra de Sousa  
 \*  
 \* Classe destinada a criar os utilizadores necessários para o funcionamento da aplicação.  
 \*/  
@SuppressWarnings("squid:S106")  
public class BaseDemoBootstrapper implements Action {  
  
 private static final String POWERUSER\_A1 = "poweruserA1";  
 private static final String POWERUSER = "poweruser";  
 private static final String MANAGER = "manager";  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final AuthenticationService authenticationService = AuthzRegistry.authenticationService();  
  
 @Override  
 public boolean execute() {  
 // declare bootstrap actions  
 final Action[] actions = { new BackofficeUsersBootstrapper(),  
 new ClientUserBootstrapper(), };  
  
 authenticateForBootstrapping();  
  
 // execute all bootstrapping  
 boolean ret = true;  
 for (final Action boot : actions) {  
 System.out.println("Bootstrapping " + nameOfEntity(boot) + "...");  
 ret &= boot.execute();  
 }  
 return ret;  
 }  
  
 /\*\*  
 \* authenticate a super user to be able to register new users  
 \*  
 \*/  
 protected void authenticateForBootstrapping() {  
 authenticationService.authenticate(POWERUSER, MANAGER);  
 Invariants.ensure(authz.hasSession());  
 }  
  
 private String nameOfEntity(final Action boot) {  
 final String name = boot.getClass().getSimpleName();  
 return Strings.left(name, name.length() - "Bootstrapper".length());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\demo\BaseDemoBootstrapper.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\demo\ClientUserBootstrapper.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.bootstrapers.demo;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import eapli.base.clientusermanagement.application.AcceptRefuseSignupFactory;  
import eapli.base.clientusermanagement.application.AcceptRefuseSignupRequestController;  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.base.infrastructure.bootstrapers.TestDataConstants;  
import eapli.base.myclientuser.application.SignupController;  
import eapli.framework.actions.Action;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
  
/\*\*  
 \*  
 \* @author Paulo Sousa  
 \* Edited by João Cruz  
 \*/  
public class ClientUserBootstrapper implements Action {  
 private static final Logger LOGGER = LoggerFactory.getLogger(ClientUserBootstrapper.class);  
  
 private final SignupController signupController = new SignupController();  
 private final AcceptRefuseSignupRequestController acceptController = AcceptRefuseSignupFactory.build();  
  
 @Override  
 public boolean execute() {  
 signupAndApprove(TestDataConstants.USER\_TEST1, "Password1", "John", "Smith", "john@smith.com", TestDataConstants.USER\_TEST1);  
 signupAndApprove(TestDataConstants.MANAGER\_TEST1, "Password1", "Calvin", "Smith", "calvin@smith.com", TestDataConstants.MANAGER\_TEST1);  
 signupAndApprove(TestDataConstants.TEACHER\_TEST1, "Password1", "Elsa", "Smith", "elsa@smith.com", TestDataConstants.TEACHER\_TEST1);  
 signupAndApprove(TestDataConstants.STUDENT\_TEST1, "Password1", "Joseph", "Smith", "joseph@smith.com", TestDataConstants.STUDENT\_TEST1);  
 signupAndApprove("isep959", "Password1", "Mary", "Smith", "mary@smith.com", "isep959");  
 return true;  
 }  
  
 private SignupRequest signupAndApprove(final String username, final String password, final String firstName, final String lastName, final String email, final String mecanographicNumber) {  
 SignupRequest request = null;  
 try {  
 request = signupController.signup(username, password, firstName, lastName, email, mecanographicNumber);  
 acceptController.acceptSignupRequest(request);  
 } catch (final ConcurrencyException | IntegrityViolationException e) {  
 // ignoring exception. assuming it is just a primary key violation  
 // due to the tentative of inserting a duplicated user  
 LOGGER.warn("Assuming {} already exists (activate trace log for details)", username);  
 LOGGER.trace("Assuming existing record", e);  
 }  
 return request;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\bootstrapers\demo\ClientUserBootstrapper.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\smoketests\BaseDemoSmokeTester.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.smoketests;  
  
import eapli.framework.actions.Action;  
  
/\*\*  
 \* execute simple smoke tests on the application layer. we are assuming that the  
 \* bootstrappers mainly test the "register" use cases and some of the "finders"  
 \* to support those "register", so this class will focus mainly on executing the  
 \* other application methods  
 \*  
 \*/  
@SuppressWarnings("squid:S1126")  
public class BaseDemoSmokeTester implements Action {  
  
 @Override  
 public boolean execute() {  
 // TODO: Add your smoke test execute here  
 return true;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\java\eapli\base\infrastructure\smoketests\BaseDemoSmokeTester.java

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\main\resources\application.properties

#####################################################  
##  
## Base  
##  
#####################################################  
  
persistence.repositoryFactory=eapli.base.persistence.impl.jpa.JpaRepositoryFactory  
#persistence.repositoryFactory=eapli.base.persistence.impl.inmemory.InMemoryRepositoryFactory  
ui.menu.layout=horizontal  
persistence.persistenceUnit=eapli.base  
HighCaloriesDishLimit=300

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\main\resources\application.properties

### [File Begins] sem4pi-22-23-61-master\base.bootstrappers\src\test\java\eapli\base\bootstrapapp\AppTest.java

package eapli.base.bootstrapapp;  
  
import junit.framework.Test;  
import junit.framework.TestCase;  
import junit.framework.TestSuite;  
  
/\*\*  
 \* Unit test for simple App.  
 \*/  
public class AppTest extends TestCase {  
  
 /\*\*  
 \* @return the suite of tests being tested  
 \*/  
 public static Test suite() {  
 return new TestSuite(AppTest.class);  
 }  
  
 /\*\*  
 \* Create the test case  
 \*  
 \* @param testName name of the test case  
 \*/  
 public AppTest(String testName) {  
 super(testName);  
 }  
  
 /\*\*  
 \* Rigourous Test :-)  
 \*/  
 public void testApp() {  
 assertTrue(true);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.bootstrappers\src\test\java\eapli\base\bootstrapapp\AppTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\machinet.conf

### Please DO NOT modify the contents of this file. For internal purpose only  
root=7e80baff-8cf9-3f44-88ad-a7f2e412d186  
rootId=c683d734-742d-3f4d-b526-21d8142491be

### [File Ends] sem4pi-22-23-61-master\base.core\machinet.conf

### [File Begins] sem4pi-22-23-61-master\base.core\pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
   
 <artifactId>base.core</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.core</name>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.infrastructure.application</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.core\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\META-INF\application.xml

<?xml version="1.0" encoding="UTF-8"?>  
<application xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/application\_8.xsd"  
 version="8">  
  
</application>

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\META-INF\application.xml

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exam.txt

package eapli.base.ANTLR.ExamValidation;  
  
import java.io.\*;  
  
import eapli.base.ANTLR.ExamValidation.ExamBaseVisitor;  
import eapli.base.ANTLR.ExamValidation.ExamLexer;  
import eapli.base.ANTLR.ExamValidation.ExamParser;  
import eapli.base.ANTLR.ExamValidation.ExamVisitor;  
import org.antlr.v4.runtime.\*;  
import org.antlr.v4.runtime.tree.\*;  
  
public class Exam {  
 public static void main(String[] args) throws IOException {  
 FileInputStream fis = new FileInputStream(new File("teste.txt"));  
 ExamLexer lexer = new ExamLexer(new ANTLRInputStream(fis));  
 CommonTokenStream tokens = new CommonTokenStream(lexer);  
 ExamParser parser = new ExamParser(tokens);  
 ParseTree tree = parser.prog(); // parse  
  
 ExamVisitor<?> visitor = new ExamBaseVisitor<>(); // Replace with your custom visitor  
 visitor.visit(tree);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exam.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExamFile.txt

Teste: {  
 ID: 1,  
 Título: Exame Final de Literatura Inglesa.  
 Secção ID: 1, {  
 Descrição Textual: Analise de Poesia.  
 Dificuldade: 3  
 Limite de Perguntas: 5  
 Perguntas:  
 Pergunta  
 ID: 1,  
 Enunciado: Qual e o tema do poema 'O Corvo'?  
 Tipo de Pergunta: Escolha Múltipla  
 Opções: [  
 Opção 1: Amor;  
 Opção 2: Desespero;  
 Opção 3: Esperanca;  
 Opção 4: Alegria;  
 ]  
 Resposta: 2  
 Cotação: 2  
 Dificuldade: 2  
 Pergunta  
 ID: 2,  
 Enunciado: Identifica o dispositivo poetico usado na frase 'Ser ou nao ser'.  
 Tipo de Pergunta: Resposta Curta  
 Resposta: Metafora.  
 Cotação: 3  
 Dificuldade: 1  
 }  
 Secção  
 ID: 2, {  
 Descrição Textual: Analise de Prosa.  
 Dificuldade: 2  
 Limite de Perguntas: 3  
 Perguntas:  
 Pergunta  
 ID: 3,  
 Enunciado: Autor do romance 'Orgulho e Preconceito' e \_\_\_1\_\_\_ \_\_\_2\_\_\_?  
 Tipo de Pergunta: Palavras em Falta  
 Resposta: {  
 Resposta 1: Jane  
 Resposta 2: Austen  
 }  
 Cotação: 4  
 Dificuldade: 2  
  
 Pergunta  
 ID: 4,  
 Enunciado: Explique o significado da luz verde em 'O Grande Gatsby'.  
 Tipo de Pergunta: Resposta Curta  
 Resposta: Simboliza a esperanca e o Sonho Americano.  
 Cotação: 1  
 Dificuldade: 3  
 Pergunta  
 ID: 5,  
 Enunciado: Quantos campeonatos tem o Benfica?  
 Tipo de Pergunta: Numérica  
 Resposta: 38  
 Cotação: 1  
 Dificuldade: 3  
 Pergunta  
 ID: 6,  
 Enunciado: ceu e azul?  
 Tipo de Pergunta: Verdadeiro ou Falso  
 Resposta: Verdadeiro  
 Cotação: 1  
 Dificuldade: 3  
 Pergunta  
 ID: 7,  
 Enunciado: Conecte a tabela correspondente!  
 Tipo de Pergunta : Correspondência  
 Tabela A: [  
 Enunciado 1: APROG.  
 Enunciado 2: Chocolate.  
 ]  
 Tabela B: [  
 Enunciado 1: bom.  
 Enunciado 2: facil.  
 ]  
 Resposta :  
 Tabela A: 1 - Tabela B: 2  
 Tabela A: 2 - Tabela B: 1  
 Cotação: 1  
 Dificuldade: 3  
  
 }  
 Data Abertura: 30/05/2023 09:00  
 Data Fecho: 06/10/2023 17:00  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExamFile.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exame.g4

grammar Exame;  
  
prog: exame;  
  
exame: TESTE DOISPONTOS ABRECHAVETA corpo\_exame FECHACHAVETA;  
  
corpo\_exame: ID DOISPONTOS id\_exame=INT VIRGULA TITULO DOISPONTOS titulo\_exame=FRASE seccao+ DATAABERTURA DOISPONTOS data\_abertura=DATA DATAFECHO DOISPONTOS data\_fecho=DATA;  
  
seccao: SECCAO ID DOISPONTOS id\_seccao=INT VIRGULA ABRECHAVETA DESCRICAO\_TEXTUAL DOISPONTOS desc\_seccao=FRASE DIFICULDADE DOISPONTOS dificuldade\_seccao=INT LIMITE\_PERGUNTAS DOISPONTOS limite=INT PERGUNTAS DOISPONTOS pergunta+ FECHACHAVETA;  
  
pergunta: PERGUNTA ID DOISPONTOS id\_pergunta=INT VIRGULA ENUNCIADO DOISPONTOS enunciado\_pergunta=FRASE tipo\_pergunta=TIPO\_PERGUNTA DOISPONTOS corpo COTACAO DOISPONTOS cotacao\_pergunta=INT DIFICULDADE DOISPONTOS dificuldade\_pergunta=INT;  
  
corpo: correspondencia  
 | escolha\_multipla  
 | resposta\_curta  
 | numerica  
 | palavras\_em\_falta  
 | verdadeiro\_falso;  
  
verdadeiro\_falso: VERDADEIROOUFALSO RESPOSTA DOISPONTOS resposta\_verdOuFal = (VERDADEIRO | FALSO) ;  
  
palavras\_em\_falta: PALAVRAS\_EM\_FALTA RESPOSTA DOISPONTOS ABRECHAVETA resposta\_falta+ FECHACHAVETA;  
  
resposta\_falta : RESPOSTA INT DOISPONTOS resp\_falta=FRASE;  
  
numerica: NUMERICA RESPOSTA DOISPONTOS resp\_numerica=NUMERO;  
  
resposta\_curta: RESPOSTA\_CURTA RESPOSTA DOISPONTOS resp\_curta=FRASE;  
  
escolha\_multipla: ESCOLHA\_MULTIPLA OPCOES DOISPONTOS ABREPARANTESIS opcao\_escolha\_multipla+ FECHAPARANTESIS RESPOSTA DOISPONTOS resposta\_escolha\_multipla=INT;  
  
opcao\_escolha\_multipla: OPCAO INT DOISPONTOS opcao\_escolha=FRASE;  
  
correspondencia: CORRESPONDENCIA TABELA\_A DOISPONTOS ABREPARANTESIS token\_correspondencia+ FECHAPARANTESIS TABELA\_B DOISPONTOS ABREPARANTESIS token\_correspondencia+ FECHAPARANTESIS RESPOSTA DOISPONTOS respostas\_correspondencia+;  
  
token\_correspondencia: ENUNCIADO INT DOISPONTOS resp\_correspondencia=FRASE;  
  
respostas\_correspondencia: TABELA\_A DOISPONTOS INT TRACO TABELA\_B DOISPONTOS INT;  
  
DOISPONTOS : ':';  
ABRECHAVETA : '{';  
FECHACHAVETA : '}';  
ABREPARANTESIS : '[';  
FECHAPARANTESIS : ']';  
VIRGULA : ',';  
TESTE : 'Teste';  
ID : 'ID';  
TRACO : '-';  
RESPOSTA : 'Resposta';  
TITULO : 'Titulo';  
DATAABERTURA : 'DataAbertura';  
DATAFECHO : 'DataFecho';  
OPCOES : 'Opcoes';  
OPCAO : 'Opcao';  
ENUNCIADO : 'Enunciado';  
TABELA\_A : 'TabelaA';  
TABELA\_B : 'TabelaB';  
COTACAO : 'Cotacao';  
DIFICULDADE : 'Dificuldade';  
DESCRICAO\_TEXTUAL : 'DescricaoTextual';  
PERGUNTAS : 'Perguntas';  
LIMITE\_PERGUNTAS : 'LimitePerguntas';  
SECCAO : 'Seccao';  
PERGUNTA : 'Pergunta';  
TIPO\_PERGUNTA : 'TipoPergunta';  
NUMERICA : 'Numerica';  
ESCOLHA\_MULTIPLA : 'EscolhaMultipla';  
CORRESPONDENCIA : 'Correspondencia';  
VERDADEIROOUFALSO : 'VerdadeiroOuFalso';  
PALAVRAS\_EM\_FALTA : 'PalavrasEmFalta';  
RESPOSTA\_CURTA : 'RespostaCurta';  
VERDADEIRO : 'Verdadeiro';  
FALSO : 'Falso';  
TIPO : 'Tipo';  
INT : [0-9]+;  
DATA : (('0' [0-9]) | ('1' [0-9]) | ('2' [0-9]) | ('3' [0-1])) '/' (('0' [0-9]) | ('1' [0-2])) '/' (([0-9] [0-9] [0-9] [0-9])) ' ' ([0-9][0-9]) DOISPONTOS ([0-9][0-9]);  
WS: [\r\n ]+ -> skip;  
CARACTERE\_ESPECIAL: [\_()+/\-'#"] -> skip;  
NUMERO: INT'.'INT | INT;  
FRASE: STRING (ESPACO STRING)\*;  
STRING : [a-zA-Z]+;  
ESPACO: [ \t] -> skip;

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exame.g4

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exame.interp

token literal names:  
null  
':'  
'{'  
'}'  
'['  
']'  
','  
'Teste'  
'ID'  
'-'  
'Resposta'  
'Titulo'  
'DataAbertura'  
'DataFecho'  
'Opcoes'  
'Opcao'  
'Enunciado'  
'TabelaA'  
'TabelaB'  
'Cotacao'  
'Dificuldade'  
'DescricaoTextual'  
'Perguntas'  
'LimitePerguntas'  
'Seccao'  
'Pergunta'  
'TipoPergunta'  
'Numerica'  
'EscolhaMultipla'  
'Correspondencia'  
'VerdadeiroOuFalso'  
'PalavrasEmFalta'  
'RespostaCurta'  
'Verdadeiro'  
'Falso'  
'Tipo'  
null  
null  
null  
null  
null  
null  
null  
null  
  
token symbolic names:  
null  
DOISPONTOS  
ABRECHAVETA  
FECHACHAVETA  
ABREPARANTESIS  
FECHAPARANTESIS  
VIRGULA  
TESTE  
ID  
TRACO  
RESPOSTA  
TITULO  
DATAABERTURA  
DATAFECHO  
OPCOES  
OPCAO  
ENUNCIADO  
TABELA\_A  
TABELA\_B  
COTACAO  
DIFICULDADE  
DESCRICAO\_TEXTUAL  
PERGUNTAS  
LIMITE\_PERGUNTAS  
SECCAO  
PERGUNTA  
TIPO\_PERGUNTA  
NUMERICA  
ESCOLHA\_MULTIPLA  
CORRESPONDENCIA  
VERDADEIROOUFALSO  
PALAVRAS\_EM\_FALTA  
RESPOSTA\_CURTA  
VERDADEIRO  
FALSO  
TIPO  
INT  
DATA  
WS  
CARACTERE\_ESPECIAL  
NUMERO  
FRASE  
STRING  
ESPACO  
  
rule names:  
prog  
exame  
corpo\_exame  
seccao  
pergunta  
corpo  
verdadeiro\_falso  
palavras\_em\_falta  
resposta\_falta  
numerica  
resposta\_curta  
escolha\_multipla  
opcao\_escolha\_multipla  
correspondencia  
token\_correspondencia  
respostas\_correspondencia  
  
  
atn:  
[4, 1, 43, 199, 2, 0, 7, 0, 2, 1, 7, 1, 2, 2, 7, 2, 2, 3, 7, 3, 2, 4, 7, 4, 2, 5, 7, 5, 2, 6, 7, 6, 2, 7, 7, 7, 2, 8, 7, 8, 2, 9, 7, 9, 2, 10, 7, 10, 2, 11, 7, 11, 2, 12, 7, 12, 2, 13, 7, 13, 2, 14, 7, 14, 2, 15, 7, 15, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 4, 2, 49, 8, 2, 11, 2, 12, 2, 50, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 4, 3, 78, 8, 3, 11, 3, 12, 3, 79, 1, 3, 1, 3, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 4, 1, 5, 1, 5, 1, 5, 1, 5, 1, 5, 1, 5, 3, 5, 108, 8, 5, 1, 6, 1, 6, 1, 6, 1, 6, 1, 6, 1, 7, 1, 7, 1, 7, 1, 7, 1, 7, 4, 7, 120, 8, 7, 11, 7, 12, 7, 121, 1, 7, 1, 7, 1, 8, 1, 8, 1, 8, 1, 8, 1, 8, 1, 9, 1, 9, 1, 9, 1, 9, 1, 9, 1, 10, 1, 10, 1, 10, 1, 10, 1, 10, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 4, 11, 146, 8, 11, 11, 11, 12, 11, 147, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 12, 1, 12, 1, 12, 1, 12, 1, 12, 1, 13, 1, 13, 1, 13, 1, 13, 1, 13, 4, 13, 165, 8, 13, 11, 13, 12, 13, 166, 1, 13, 1, 13, 1, 13, 1, 13, 1, 13, 4, 13, 174, 8, 13, 11, 13, 12, 13, 175, 1, 13, 1, 13, 1, 13, 1, 13, 4, 13, 182, 8, 13, 11, 13, 12, 13, 183, 1, 14, 1, 14, 1, 14, 1, 14, 1, 14, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 0, 0, 16, 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 0, 1, 1, 0, 33, 34, 194, 0, 32, 1, 0, 0, 0, 2, 34, 1, 0, 0, 0, 4, 40, 1, 0, 0, 0, 6, 59, 1, 0, 0, 0, 8, 83, 1, 0, 0, 0, 10, 107, 1, 0, 0, 0, 12, 109, 1, 0, 0, 0, 14, 114, 1, 0, 0, 0, 16, 125, 1, 0, 0, 0, 18, 130, 1, 0, 0, 0, 20, 135, 1, 0, 0, 0, 22, 140, 1, 0, 0, 0, 24, 154, 1, 0, 0, 0, 26, 159, 1, 0, 0, 0, 28, 185, 1, 0, 0, 0, 30, 190, 1, 0, 0, 0, 32, 33, 3, 2, 1, 0, 33, 1, 1, 0, 0, 0, 34, 35, 5, 7, 0, 0, 35, 36, 5, 1, 0, 0, 36, 37, 5, 2, 0, 0, 37, 38, 3, 4, 2, 0, 38, 39, 5, 3, 0, 0, 39, 3, 1, 0, 0, 0, 40, 41, 5, 8, 0, 0, 41, 42, 5, 1, 0, 0, 42, 43, 5, 36, 0, 0, 43, 44, 5, 6, 0, 0, 44, 45, 5, 11, 0, 0, 45, 46, 5, 1, 0, 0, 46, 48, 5, 41, 0, 0, 47, 49, 3, 6, 3, 0, 48, 47, 1, 0, 0, 0, 49, 50, 1, 0, 0, 0, 50, 48, 1, 0, 0, 0, 50, 51, 1, 0, 0, 0, 51, 52, 1, 0, 0, 0, 52, 53, 5, 12, 0, 0, 53, 54, 5, 1, 0, 0, 54, 55, 5, 37, 0, 0, 55, 56, 5, 13, 0, 0, 56, 57, 5, 1, 0, 0, 57, 58, 5, 37, 0, 0, 58, 5, 1, 0, 0, 0, 59, 60, 5, 24, 0, 0, 60, 61, 5, 8, 0, 0, 61, 62, 5, 1, 0, 0, 62, 63, 5, 36, 0, 0, 63, 64, 5, 6, 0, 0, 64, 65, 5, 2, 0, 0, 65, 66, 5, 21, 0, 0, 66, 67, 5, 1, 0, 0, 67, 68, 5, 41, 0, 0, 68, 69, 5, 20, 0, 0, 69, 70, 5, 1, 0, 0, 70, 71, 5, 36, 0, 0, 71, 72, 5, 23, 0, 0, 72, 73, 5, 1, 0, 0, 73, 74, 5, 36, 0, 0, 74, 75, 5, 22, 0, 0, 75, 77, 5, 1, 0, 0, 76, 78, 3, 8, 4, 0, 77, 76, 1, 0, 0, 0, 78, 79, 1, 0, 0, 0, 79, 77, 1, 0, 0, 0, 79, 80, 1, 0, 0, 0, 80, 81, 1, 0, 0, 0, 81, 82, 5, 3, 0, 0, 82, 7, 1, 0, 0, 0, 83, 84, 5, 25, 0, 0, 84, 85, 5, 8, 0, 0, 85, 86, 5, 1, 0, 0, 86, 87, 5, 36, 0, 0, 87, 88, 5, 6, 0, 0, 88, 89, 5, 16, 0, 0, 89, 90, 5, 1, 0, 0, 90, 91, 5, 41, 0, 0, 91, 92, 5, 26, 0, 0, 92, 93, 5, 1, 0, 0, 93, 94, 3, 10, 5, 0, 94, 95, 5, 19, 0, 0, 95, 96, 5, 1, 0, 0, 96, 97, 5, 36, 0, 0, 97, 98, 5, 20, 0, 0, 98, 99, 5, 1, 0, 0, 99, 100, 5, 36, 0, 0, 100, 9, 1, 0, 0, 0, 101, 108, 3, 26, 13, 0, 102, 108, 3, 22, 11, 0, 103, 108, 3, 20, 10, 0, 104, 108, 3, 18, 9, 0, 105, 108, 3, 14, 7, 0, 106, 108, 3, 12, 6, 0, 107, 101, 1, 0, 0, 0, 107, 102, 1, 0, 0, 0, 107, 103, 1, 0, 0, 0, 107, 104, 1, 0, 0, 0, 107, 105, 1, 0, 0, 0, 107, 106, 1, 0, 0, 0, 108, 11, 1, 0, 0, 0, 109, 110, 5, 30, 0, 0, 110, 111, 5, 10, 0, 0, 111, 112, 5, 1, 0, 0, 112, 113, 7, 0, 0, 0, 113, 13, 1, 0, 0, 0, 114, 115, 5, 31, 0, 0, 115, 116, 5, 10, 0, 0, 116, 117, 5, 1, 0, 0, 117, 119, 5, 2, 0, 0, 118, 120, 3, 16, 8, 0, 119, 118, 1, 0, 0, 0, 120, 121, 1, 0, 0, 0, 121, 119, 1, 0, 0, 0, 121, 122, 1, 0, 0, 0, 122, 123, 1, 0, 0, 0, 123, 124, 5, 3, 0, 0, 124, 15, 1, 0, 0, 0, 125, 126, 5, 10, 0, 0, 126, 127, 5, 36, 0, 0, 127, 128, 5, 1, 0, 0, 128, 129, 5, 41, 0, 0, 129, 17, 1, 0, 0, 0, 130, 131, 5, 27, 0, 0, 131, 132, 5, 10, 0, 0, 132, 133, 5, 1, 0, 0, 133, 134, 5, 40, 0, 0, 134, 19, 1, 0, 0, 0, 135, 136, 5, 32, 0, 0, 136, 137, 5, 10, 0, 0, 137, 138, 5, 1, 0, 0, 138, 139, 5, 41, 0, 0, 139, 21, 1, 0, 0, 0, 140, 141, 5, 28, 0, 0, 141, 142, 5, 14, 0, 0, 142, 143, 5, 1, 0, 0, 143, 145, 5, 4, 0, 0, 144, 146, 3, 24, 12, 0, 145, 144, 1, 0, 0, 0, 146, 147, 1, 0, 0, 0, 147, 145, 1, 0, 0, 0, 147, 148, 1, 0, 0, 0, 148, 149, 1, 0, 0, 0, 149, 150, 5, 5, 0, 0, 150, 151, 5, 10, 0, 0, 151, 152, 5, 1, 0, 0, 152, 153, 5, 36, 0, 0, 153, 23, 1, 0, 0, 0, 154, 155, 5, 15, 0, 0, 155, 156, 5, 36, 0, 0, 156, 157, 5, 1, 0, 0, 157, 158, 5, 41, 0, 0, 158, 25, 1, 0, 0, 0, 159, 160, 5, 29, 0, 0, 160, 161, 5, 17, 0, 0, 161, 162, 5, 1, 0, 0, 162, 164, 5, 4, 0, 0, 163, 165, 3, 28, 14, 0, 164, 163, 1, 0, 0, 0, 165, 166, 1, 0, 0, 0, 166, 164, 1, 0, 0, 0, 166, 167, 1, 0, 0, 0, 167, 168, 1, 0, 0, 0, 168, 169, 5, 5, 0, 0, 169, 170, 5, 18, 0, 0, 170, 171, 5, 1, 0, 0, 171, 173, 5, 4, 0, 0, 172, 174, 3, 28, 14, 0, 173, 172, 1, 0, 0, 0, 174, 175, 1, 0, 0, 0, 175, 173, 1, 0, 0, 0, 175, 176, 1, 0, 0, 0, 176, 177, 1, 0, 0, 0, 177, 178, 5, 5, 0, 0, 178, 179, 5, 10, 0, 0, 179, 181, 5, 1, 0, 0, 180, 182, 3, 30, 15, 0, 181, 180, 1, 0, 0, 0, 182, 183, 1, 0, 0, 0, 183, 181, 1, 0, 0, 0, 183, 184, 1, 0, 0, 0, 184, 27, 1, 0, 0, 0, 185, 186, 5, 16, 0, 0, 186, 187, 5, 36, 0, 0, 187, 188, 5, 1, 0, 0, 188, 189, 5, 41, 0, 0, 189, 29, 1, 0, 0, 0, 190, 191, 5, 17, 0, 0, 191, 192, 5, 1, 0, 0, 192, 193, 5, 36, 0, 0, 193, 194, 5, 9, 0, 0, 194, 195, 5, 18, 0, 0, 195, 196, 5, 1, 0, 0, 196, 197, 5, 36, 0, 0, 197, 31, 1, 0, 0, 0, 8, 50, 79, 107, 121, 147, 166, 175, 183]

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exame.interp

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exame.tokens

DOISPONTOS=1  
ABRECHAVETA=2  
FECHACHAVETA=3  
ABREPARANTESIS=4  
FECHAPARANTESIS=5  
VIRGULA=6  
TESTE=7  
ID=8  
TRACO=9  
RESPOSTA=10  
TITULO=11  
DATAABERTURA=12  
DATAFECHO=13  
OPCOES=14  
OPCAO=15  
ENUNCIADO=16  
TABELA\_A=17  
TABELA\_B=18  
COTACAO=19  
DIFICULDADE=20  
DESCRICAO\_TEXTUAL=21  
PERGUNTAS=22  
LIMITE\_PERGUNTAS=23  
SECCAO=24  
PERGUNTA=25  
TIPO\_PERGUNTA=26  
NUMERICA=27  
ESCOLHA\_MULTIPLA=28  
CORRESPONDENCIA=29  
VERDADEIROOUFALSO=30  
PALAVRAS\_EM\_FALTA=31  
RESPOSTA\_CURTA=32  
VERDADEIRO=33  
FALSO=34  
TIPO=35  
INT=36  
DATA=37  
WS=38  
CARACTERE\_ESPECIAL=39  
NUMERO=40  
FRASE=41  
STRING=42  
ESPACO=43  
':'=1  
'{'=2  
'}'=3  
'['=4  
']'=5  
','=6  
'Teste'=7  
'ID'=8  
'-'=9  
'Resposta'=10  
'Titulo'=11  
'DataAbertura'=12  
'DataFecho'=13  
'Opcoes'=14  
'Opcao'=15  
'Enunciado'=16  
'TabelaA'=17  
'TabelaB'=18  
'Cotacao'=19  
'Dificuldade'=20  
'DescricaoTextual'=21  
'Perguntas'=22  
'LimitePerguntas'=23  
'Seccao'=24  
'Pergunta'=25  
'TipoPergunta'=26  
'Numerica'=27  
'EscolhaMultipla'=28  
'Correspondencia'=29  
'VerdadeiroOuFalso'=30  
'PalavrasEmFalta'=31  
'RespostaCurta'=32  
'Verdadeiro'=33  
'Falso'=34  
'Tipo'=35

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\Exame.tokens

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameBaseListener.txt

// Generated from C:/Users/mike\_/OneDrive/Documentos/sem4pi-22-23-61/base.core/src/main/java/eapli/base/ANTLR/ExamValidation\Exame.g4 by ANTLR 4.12.0  
package eapli.base.ANTLR.ExamValidation;  
  
import org.antlr.v4.runtime.ParserRuleContext;  
import org.antlr.v4.runtime.tree.ErrorNode;  
import org.antlr.v4.runtime.tree.TerminalNode;  
  
/\*\*  
 \* This class provides an empty implementation of {@link ExameListener},  
 \* which can be extended to create a listener which only needs to handle a subset  
 \* of the available methods.  
 \*/  
@SuppressWarnings("CheckReturnValue")  
public class ExameBaseListener implements ExameListener {  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterProg(ExameParser.ProgContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitProg(ExameParser.ProgContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterExame(ExameParser.ExameContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitExame(ExameParser.ExameContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterCorpo\_exame(ExameParser.Corpo\_exameContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitCorpo\_exame(ExameParser.Corpo\_exameContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterSeccao(ExameParser.SeccaoContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitSeccao(ExameParser.SeccaoContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterPergunta(ExameParser.PerguntaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitPergunta(ExameParser.PerguntaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterCorpo(ExameParser.CorpoContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitCorpo(ExameParser.CorpoContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterVerdadeiro\_falso(ExameParser.Verdadeiro\_falsoContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitVerdadeiro\_falso(ExameParser.Verdadeiro\_falsoContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterPalavras\_em\_falta(ExameParser.Palavras\_em\_faltaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitPalavras\_em\_falta(ExameParser.Palavras\_em\_faltaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterResposta\_falta(ExameParser.Resposta\_faltaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitResposta\_falta(ExameParser.Resposta\_faltaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterNumerica(ExameParser.NumericaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitNumerica(ExameParser.NumericaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterResposta\_curta(ExameParser.Resposta\_curtaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitResposta\_curta(ExameParser.Resposta\_curtaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterEscolha\_multipla(ExameParser.Escolha\_multiplaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitEscolha\_multipla(ExameParser.Escolha\_multiplaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterOpcao\_escolha\_multipla(ExameParser.Opcao\_escolha\_multiplaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitOpcao\_escolha\_multipla(ExameParser.Opcao\_escolha\_multiplaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterCorrespondencia(ExameParser.CorrespondenciaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitCorrespondencia(ExameParser.CorrespondenciaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterToken\_correspondencia(ExameParser.Token\_correspondenciaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitToken\_correspondencia(ExameParser.Token\_correspondenciaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterRespostas\_correspondencia(ExameParser.Respostas\_correspondenciaContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitRespostas\_correspondencia(ExameParser.Respostas\_correspondenciaContext ctx) { }  
  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void enterEveryRule(ParserRuleContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void exitEveryRule(ParserRuleContext ctx) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void visitTerminal(TerminalNode node) { }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation does nothing.</p>  
 \*/  
 @Override public void visitErrorNode(ErrorNode node) { }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameBaseListener.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameBaseVisitor.txt

// Generated from C:/Users/mike\_/OneDrive/Documentos/sem4pi-22-23-61/base.core/src/main/java/eapli/base/ANTLR/ExamValidation\Exame.g4 by ANTLR 4.12.0  
package eapli.base.ANTLR.ExamValidation;  
import org.antlr.v4.runtime.tree.AbstractParseTreeVisitor;  
  
/\*\*  
 \* This class provides an empty implementation of {@link ExameVisitor},  
 \* which can be extended to create a visitor which only needs to handle a subset  
 \* of the available methods.  
 \*  
 \* @param <T> The return type of the visit operation. Use {@link Void} for  
 \* operations with no return type.  
 \*/  
@SuppressWarnings("CheckReturnValue")  
public class ExameBaseVisitor<T> extends AbstractParseTreeVisitor<T> implements ExameVisitor<T> {  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitProg(ExameParser.ProgContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitExame(ExameParser.ExameContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitCorpo\_exame(ExameParser.Corpo\_exameContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitSeccao(ExameParser.SeccaoContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitPergunta(ExameParser.PerguntaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitCorpo(ExameParser.CorpoContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitVerdadeiro\_falso(ExameParser.Verdadeiro\_falsoContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitPalavras\_em\_falta(ExameParser.Palavras\_em\_faltaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitResposta\_falta(ExameParser.Resposta\_faltaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitNumerica(ExameParser.NumericaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitResposta\_curta(ExameParser.Resposta\_curtaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitEscolha\_multipla(ExameParser.Escolha\_multiplaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitOpcao\_escolha\_multipla(ExameParser.Opcao\_escolha\_multiplaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitCorrespondencia(ExameParser.CorrespondenciaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitToken\_correspondencia(ExameParser.Token\_correspondenciaContext ctx) { return visitChildren(ctx); }  
 /\*\*  
 \* {@inheritDoc}  
 \*  
 \* <p>The default implementation returns the result of calling  
 \* {@link #visitChildren} on {@code ctx}.</p>  
 \*/  
 @Override public T visitRespostas\_correspondencia(ExameParser.Respostas\_correspondenciaContext ctx) { return visitChildren(ctx); }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameBaseVisitor.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameLexer.interp

token literal names:  
null  
':'  
'{'  
'}'  
'['  
']'  
','  
'Teste'  
'ID'  
'-'  
'Resposta'  
'Titulo'  
'DataAbertura'  
'DataFecho'  
'Opcoes'  
'Opcao'  
'Enunciado'  
'TabelaA'  
'TabelaB'  
'Cotacao'  
'Dificuldade'  
'DescricaoTextual'  
'Perguntas'  
'LimitePerguntas'  
'Seccao'  
'Pergunta'  
'TipoPergunta'  
'Numerica'  
'EscolhaMultipla'  
'Correspondencia'  
'VerdadeiroOuFalso'  
'PalavrasEmFalta'  
'RespostaCurta'  
'Verdadeiro'  
'Falso'  
'Tipo'  
null  
null  
null  
null  
null  
null  
null  
null  
  
token symbolic names:  
null  
DOISPONTOS  
ABRECHAVETA  
FECHACHAVETA  
ABREPARANTESIS  
FECHAPARANTESIS  
VIRGULA  
TESTE  
ID  
TRACO  
RESPOSTA  
TITULO  
DATAABERTURA  
DATAFECHO  
OPCOES  
OPCAO  
ENUNCIADO  
TABELA\_A  
TABELA\_B  
COTACAO  
DIFICULDADE  
DESCRICAO\_TEXTUAL  
PERGUNTAS  
LIMITE\_PERGUNTAS  
SECCAO  
PERGUNTA  
TIPO\_PERGUNTA  
NUMERICA  
ESCOLHA\_MULTIPLA  
CORRESPONDENCIA  
VERDADEIROOUFALSO  
PALAVRAS\_EM\_FALTA  
RESPOSTA\_CURTA  
VERDADEIRO  
FALSO  
TIPO  
INT  
DATA  
WS  
CARACTERE\_ESPECIAL  
NUMERO  
FRASE  
STRING  
ESPACO  
  
rule names:  
DOISPONTOS  
ABRECHAVETA  
FECHACHAVETA  
ABREPARANTESIS  
FECHAPARANTESIS  
VIRGULA  
TESTE  
ID  
TRACO  
RESPOSTA  
TITULO  
DATAABERTURA  
DATAFECHO  
OPCOES  
OPCAO  
ENUNCIADO  
TABELA\_A  
TABELA\_B  
COTACAO  
DIFICULDADE  
DESCRICAO\_TEXTUAL  
PERGUNTAS  
LIMITE\_PERGUNTAS  
SECCAO  
PERGUNTA  
TIPO\_PERGUNTA  
NUMERICA  
ESCOLHA\_MULTIPLA  
CORRESPONDENCIA  
VERDADEIROOUFALSO  
PALAVRAS\_EM\_FALTA  
RESPOSTA\_CURTA  
VERDADEIRO  
FALSO  
TIPO  
INT  
DATA  
WS  
CARACTERE\_ESPECIAL  
NUMERO  
FRASE  
STRING  
ESPACO  
  
channel names:  
DEFAULT\_TOKEN\_CHANNEL  
HIDDEN  
  
mode names:  
DEFAULT\_MODE  
  
atn:  
[4, 0, 43, 463, 6, -1, 2, 0, 7, 0, 2, 1, 7, 1, 2, 2, 7, 2, 2, 3, 7, 3, 2, 4, 7, 4, 2, 5, 7, 5, 2, 6, 7, 6, 2, 7, 7, 7, 2, 8, 7, 8, 2, 9, 7, 9, 2, 10, 7, 10, 2, 11, 7, 11, 2, 12, 7, 12, 2, 13, 7, 13, 2, 14, 7, 14, 2, 15, 7, 15, 2, 16, 7, 16, 2, 17, 7, 17, 2, 18, 7, 18, 2, 19, 7, 19, 2, 20, 7, 20, 2, 21, 7, 21, 2, 22, 7, 22, 2, 23, 7, 23, 2, 24, 7, 24, 2, 25, 7, 25, 2, 26, 7, 26, 2, 27, 7, 27, 2, 28, 7, 28, 2, 29, 7, 29, 2, 30, 7, 30, 2, 31, 7, 31, 2, 32, 7, 32, 2, 33, 7, 33, 2, 34, 7, 34, 2, 35, 7, 35, 2, 36, 7, 36, 2, 37, 7, 37, 2, 38, 7, 38, 2, 39, 7, 39, 2, 40, 7, 40, 2, 41, 7, 41, 2, 42, 7, 42, 1, 0, 1, 0, 1, 1, 1, 1, 1, 2, 1, 2, 1, 3, 1, 3, 1, 4, 1, 4, 1, 5, 1, 5, 1, 6, 1, 6, 1, 6, 1, 6, 1, 6, 1, 6, 1, 7, 1, 7, 1, 7, 1, 8, 1, 8, 1, 9, 1, 9, 1, 9, 1, 9, 1, 9, 1, 9, 1, 9, 1, 9, 1, 9, 1, 10, 1, 10, 1, 10, 1, 10, 1, 10, 1, 10, 1, 10, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 11, 1, 12, 1, 12, 1, 12, 1, 12, 1, 12, 1, 12, 1, 12, 1, 12, 1, 12, 1, 12, 1, 13, 1, 13, 1, 13, 1, 13, 1, 13, 1, 13, 1, 13, 1, 14, 1, 14, 1, 14, 1, 14, 1, 14, 1, 14, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 15, 1, 16, 1, 16, 1, 16, 1, 16, 1, 16, 1, 16, 1, 16, 1, 16, 1, 17, 1, 17, 1, 17, 1, 17, 1, 17, 1, 17, 1, 17, 1, 17, 1, 18, 1, 18, 1, 18, 1, 18, 1, 18, 1, 18, 1, 18, 1, 18, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 19, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 20, 1, 21, 1, 21, 1, 21, 1, 21, 1, 21, 1, 21, 1, 21, 1, 21, 1, 21, 1, 21, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 22, 1, 23, 1, 23, 1, 23, 1, 23, 1, 23, 1, 23, 1, 23, 1, 24, 1, 24, 1, 24, 1, 24, 1, 24, 1, 24, 1, 24, 1, 24, 1, 24, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 25, 1, 26, 1, 26, 1, 26, 1, 26, 1, 26, 1, 26, 1, 26, 1, 26, 1, 26, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 27, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 28, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 30, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 31, 1, 32, 1, 32, 1, 32, 1, 32, 1, 32, 1, 32, 1, 32, 1, 32, 1, 32, 1, 32, 1, 32, 1, 33, 1, 33, 1, 33, 1, 33, 1, 33, 1, 33, 1, 34, 1, 34, 1, 34, 1, 34, 1, 34, 1, 35, 4, 35, 393, 8, 35, 11, 35, 12, 35, 394, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 3, 36, 405, 8, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 3, 36, 412, 8, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 36, 1, 37, 4, 37, 429, 8, 37, 11, 37, 12, 37, 430, 1, 37, 1, 37, 1, 38, 1, 38, 1, 38, 1, 38, 1, 39, 1, 39, 1, 39, 1, 39, 1, 39, 3, 39, 444, 8, 39, 1, 40, 1, 40, 1, 40, 1, 40, 5, 40, 450, 8, 40, 10, 40, 12, 40, 453, 9, 40, 1, 41, 4, 41, 456, 8, 41, 11, 41, 12, 41, 457, 1, 42, 1, 42, 1, 42, 1, 42, 0, 0, 43, 1, 1, 3, 2, 5, 3, 7, 4, 9, 5, 11, 6, 13, 7, 15, 8, 17, 9, 19, 10, 21, 11, 23, 12, 25, 13, 27, 14, 29, 15, 31, 16, 33, 17, 35, 18, 37, 19, 39, 20, 41, 21, 43, 22, 45, 23, 47, 24, 49, 25, 51, 26, 53, 27, 55, 28, 57, 29, 59, 30, 61, 31, 63, 32, 65, 33, 67, 34, 69, 35, 71, 36, 73, 37, 75, 38, 77, 39, 79, 40, 81, 41, 83, 42, 85, 43, 1, 0, 7, 1, 0, 48, 57, 1, 0, 48, 49, 1, 0, 48, 50, 3, 0, 10, 10, 13, 13, 32, 32, 6, 0, 34, 35, 39, 41, 43, 43, 45, 45, 47, 47, 95, 95, 2, 0, 65, 90, 97, 122, 2, 0, 9, 9, 32, 32, 471, 0, 1, 1, 0, 0, 0, 0, 3, 1, 0, 0, 0, 0, 5, 1, 0, 0, 0, 0, 7, 1, 0, 0, 0, 0, 9, 1, 0, 0, 0, 0, 11, 1, 0, 0, 0, 0, 13, 1, 0, 0, 0, 0, 15, 1, 0, 0, 0, 0, 17, 1, 0, 0, 0, 0, 19, 1, 0, 0, 0, 0, 21, 1, 0, 0, 0, 0, 23, 1, 0, 0, 0, 0, 25, 1, 0, 0, 0, 0, 27, 1, 0, 0, 0, 0, 29, 1, 0, 0, 0, 0, 31, 1, 0, 0, 0, 0, 33, 1, 0, 0, 0, 0, 35, 1, 0, 0, 0, 0, 37, 1, 0, 0, 0, 0, 39, 1, 0, 0, 0, 0, 41, 1, 0, 0, 0, 0, 43, 1, 0, 0, 0, 0, 45, 1, 0, 0, 0, 0, 47, 1, 0, 0, 0, 0, 49, 1, 0, 0, 0, 0, 51, 1, 0, 0, 0, 0, 53, 1, 0, 0, 0, 0, 55, 1, 0, 0, 0, 0, 57, 1, 0, 0, 0, 0, 59, 1, 0, 0, 0, 0, 61, 1, 0, 0, 0, 0, 63, 1, 0, 0, 0, 0, 65, 1, 0, 0, 0, 0, 67, 1, 0, 0, 0, 0, 69, 1, 0, 0, 0, 0, 71, 1, 0, 0, 0, 0, 73, 1, 0, 0, 0, 0, 75, 1, 0, 0, 0, 0, 77, 1, 0, 0, 0, 0, 79, 1, 0, 0, 0, 0, 81, 1, 0, 0, 0, 0, 83, 1, 0, 0, 0, 0, 85, 1, 0, 0, 0, 1, 87, 1, 0, 0, 0, 3, 89, 1, 0, 0, 0, 5, 91, 1, 0, 0, 0, 7, 93, 1, 0, 0, 0, 9, 95, 1, 0, 0, 0, 11, 97, 1, 0, 0, 0, 13, 99, 1, 0, 0, 0, 15, 105, 1, 0, 0, 0, 17, 108, 1, 0, 0, 0, 19, 110, 1, 0, 0, 0, 21, 119, 1, 0, 0, 0, 23, 126, 1, 0, 0, 0, 25, 139, 1, 0, 0, 0, 27, 149, 1, 0, 0, 0, 29, 156, 1, 0, 0, 0, 31, 162, 1, 0, 0, 0, 33, 172, 1, 0, 0, 0, 35, 180, 1, 0, 0, 0, 37, 188, 1, 0, 0, 0, 39, 196, 1, 0, 0, 0, 41, 208, 1, 0, 0, 0, 43, 225, 1, 0, 0, 0, 45, 235, 1, 0, 0, 0, 47, 251, 1, 0, 0, 0, 49, 258, 1, 0, 0, 0, 51, 267, 1, 0, 0, 0, 53, 280, 1, 0, 0, 0, 55, 289, 1, 0, 0, 0, 57, 305, 1, 0, 0, 0, 59, 321, 1, 0, 0, 0, 61, 339, 1, 0, 0, 0, 63, 355, 1, 0, 0, 0, 65, 369, 1, 0, 0, 0, 67, 380, 1, 0, 0, 0, 69, 386, 1, 0, 0, 0, 71, 392, 1, 0, 0, 0, 73, 404, 1, 0, 0, 0, 75, 428, 1, 0, 0, 0, 77, 434, 1, 0, 0, 0, 79, 443, 1, 0, 0, 0, 81, 445, 1, 0, 0, 0, 83, 455, 1, 0, 0, 0, 85, 459, 1, 0, 0, 0, 87, 88, 5, 58, 0, 0, 88, 2, 1, 0, 0, 0, 89, 90, 5, 123, 0, 0, 90, 4, 1, 0, 0, 0, 91, 92, 5, 125, 0, 0, 92, 6, 1, 0, 0, 0, 93, 94, 5, 91, 0, 0, 94, 8, 1, 0, 0, 0, 95, 96, 5, 93, 0, 0, 96, 10, 1, 0, 0, 0, 97, 98, 5, 44, 0, 0, 98, 12, 1, 0, 0, 0, 99, 100, 5, 84, 0, 0, 100, 101, 5, 101, 0, 0, 101, 102, 5, 115, 0, 0, 102, 103, 5, 116, 0, 0, 103, 104, 5, 101, 0, 0, 104, 14, 1, 0, 0, 0, 105, 106, 5, 73, 0, 0, 106, 107, 5, 68, 0, 0, 107, 16, 1, 0, 0, 0, 108, 109, 5, 45, 0, 0, 109, 18, 1, 0, 0, 0, 110, 111, 5, 82, 0, 0, 111, 112, 5, 101, 0, 0, 112, 113, 5, 115, 0, 0, 113, 114, 5, 112, 0, 0, 114, 115, 5, 111, 0, 0, 115, 116, 5, 115, 0, 0, 116, 117, 5, 116, 0, 0, 117, 118, 5, 97, 0, 0, 118, 20, 1, 0, 0, 0, 119, 120, 5, 84, 0, 0, 120, 121, 5, 105, 0, 0, 121, 122, 5, 116, 0, 0, 122, 123, 5, 117, 0, 0, 123, 124, 5, 108, 0, 0, 124, 125, 5, 111, 0, 0, 125, 22, 1, 0, 0, 0, 126, 127, 5, 68, 0, 0, 127, 128, 5, 97, 0, 0, 128, 129, 5, 116, 0, 0, 129, 130, 5, 97, 0, 0, 130, 131, 5, 65, 0, 0, 131, 132, 5, 98, 0, 0, 132, 133, 5, 101, 0, 0, 133, 134, 5, 114, 0, 0, 134, 135, 5, 116, 0, 0, 135, 136, 5, 117, 0, 0, 136, 137, 5, 114, 0, 0, 137, 138, 5, 97, 0, 0, 138, 24, 1, 0, 0, 0, 139, 140, 5, 68, 0, 0, 140, 141, 5, 97, 0, 0, 141, 142, 5, 116, 0, 0, 142, 143, 5, 97, 0, 0, 143, 144, 5, 70, 0, 0, 144, 145, 5, 101, 0, 0, 145, 146, 5, 99, 0, 0, 146, 147, 5, 104, 0, 0, 147, 148, 5, 111, 0, 0, 148, 26, 1, 0, 0, 0, 149, 150, 5, 79, 0, 0, 150, 151, 5, 112, 0, 0, 151, 152, 5, 99, 0, 0, 152, 153, 5, 111, 0, 0, 153, 154, 5, 101, 0, 0, 154, 155, 5, 115, 0, 0, 155, 28, 1, 0, 0, 0, 156, 157, 5, 79, 0, 0, 157, 158, 5, 112, 0, 0, 158, 159, 5, 99, 0, 0, 159, 160, 5, 97, 0, 0, 160, 161, 5, 111, 0, 0, 161, 30, 1, 0, 0, 0, 162, 163, 5, 69, 0, 0, 163, 164, 5, 110, 0, 0, 164, 165, 5, 117, 0, 0, 165, 166, 5, 110, 0, 0, 166, 167, 5, 99, 0, 0, 167, 168, 5, 105, 0, 0, 168, 169, 5, 97, 0, 0, 169, 170, 5, 100, 0, 0, 170, 171, 5, 111, 0, 0, 171, 32, 1, 0, 0, 0, 172, 173, 5, 84, 0, 0, 173, 174, 5, 97, 0, 0, 174, 175, 5, 98, 0, 0, 175, 176, 5, 101, 0, 0, 176, 177, 5, 108, 0, 0, 177, 178, 5, 97, 0, 0, 178, 179, 5, 65, 0, 0, 179, 34, 1, 0, 0, 0, 180, 181, 5, 84, 0, 0, 181, 182, 5, 97, 0, 0, 182, 183, 5, 98, 0, 0, 183, 184, 5, 101, 0, 0, 184, 185, 5, 108, 0, 0, 185, 186, 5, 97, 0, 0, 186, 187, 5, 66, 0, 0, 187, 36, 1, 0, 0, 0, 188, 189, 5, 67, 0, 0, 189, 190, 5, 111, 0, 0, 190, 191, 5, 116, 0, 0, 191, 192, 5, 97, 0, 0, 192, 193, 5, 99, 0, 0, 193, 194, 5, 97, 0, 0, 194, 195, 5, 111, 0, 0, 195, 38, 1, 0, 0, 0, 196, 197, 5, 68, 0, 0, 197, 198, 5, 105, 0, 0, 198, 199, 5, 102, 0, 0, 199, 200, 5, 105, 0, 0, 200, 201, 5, 99, 0, 0, 201, 202, 5, 117, 0, 0, 202, 203, 5, 108, 0, 0, 203, 204, 5, 100, 0, 0, 204, 205, 5, 97, 0, 0, 205, 206, 5, 100, 0, 0, 206, 207, 5, 101, 0, 0, 207, 40, 1, 0, 0, 0, 208, 209, 5, 68, 0, 0, 209, 210, 5, 101, 0, 0, 210, 211, 5, 115, 0, 0, 211, 212, 5, 99, 0, 0, 212, 213, 5, 114, 0, 0, 213, 214, 5, 105, 0, 0, 214, 215, 5, 99, 0, 0, 215, 216, 5, 97, 0, 0, 216, 217, 5, 111, 0, 0, 217, 218, 5, 84, 0, 0, 218, 219, 5, 101, 0, 0, 219, 220, 5, 120, 0, 0, 220, 221, 5, 116, 0, 0, 221, 222, 5, 117, 0, 0, 222, 223, 5, 97, 0, 0, 223, 224, 5, 108, 0, 0, 224, 42, 1, 0, 0, 0, 225, 226, 5, 80, 0, 0, 226, 227, 5, 101, 0, 0, 227, 228, 5, 114, 0, 0, 228, 229, 5, 103, 0, 0, 229, 230, 5, 117, 0, 0, 230, 231, 5, 110, 0, 0, 231, 232, 5, 116, 0, 0, 232, 233, 5, 97, 0, 0, 233, 234, 5, 115, 0, 0, 234, 44, 1, 0, 0, 0, 235, 236, 5, 76, 0, 0, 236, 237, 5, 105, 0, 0, 237, 238, 5, 109, 0, 0, 238, 239, 5, 105, 0, 0, 239, 240, 5, 116, 0, 0, 240, 241, 5, 101, 0, 0, 241, 242, 5, 80, 0, 0, 242, 243, 5, 101, 0, 0, 243, 244, 5, 114, 0, 0, 244, 245, 5, 103, 0, 0, 245, 246, 5, 117, 0, 0, 246, 247, 5, 110, 0, 0, 247, 248, 5, 116, 0, 0, 248, 249, 5, 97, 0, 0, 249, 250, 5, 115, 0, 0, 250, 46, 1, 0, 0, 0, 251, 252, 5, 83, 0, 0, 252, 253, 5, 101, 0, 0, 253, 254, 5, 99, 0, 0, 254, 255, 5, 99, 0, 0, 255, 256, 5, 97, 0, 0, 256, 257, 5, 111, 0, 0, 257, 48, 1, 0, 0, 0, 258, 259, 5, 80, 0, 0, 259, 260, 5, 101, 0, 0, 260, 261, 5, 114, 0, 0, 261, 262, 5, 103, 0, 0, 262, 263, 5, 117, 0, 0, 263, 264, 5, 110, 0, 0, 264, 265, 5, 116, 0, 0, 265, 266, 5, 97, 0, 0, 266, 50, 1, 0, 0, 0, 267, 268, 5, 84, 0, 0, 268, 269, 5, 105, 0, 0, 269, 270, 5, 112, 0, 0, 270, 271, 5, 111, 0, 0, 271, 272, 5, 80, 0, 0, 272, 273, 5, 101, 0, 0, 273, 274, 5, 114, 0, 0, 274, 275, 5, 103, 0, 0, 275, 276, 5, 117, 0, 0, 276, 277, 5, 110, 0, 0, 277, 278, 5, 116, 0, 0, 278, 279, 5, 97, 0, 0, 279, 52, 1, 0, 0, 0, 280, 281, 5, 78, 0, 0, 281, 282, 5, 117, 0, 0, 282, 283, 5, 109, 0, 0, 283, 284, 5, 101, 0, 0, 284, 285, 5, 114, 0, 0, 285, 286, 5, 105, 0, 0, 286, 287, 5, 99, 0, 0, 287, 288, 5, 97, 0, 0, 288, 54, 1, 0, 0, 0, 289, 290, 5, 69, 0, 0, 290, 291, 5, 115, 0, 0, 291, 292, 5, 99, 0, 0, 292, 293, 5, 111, 0, 0, 293, 294, 5, 108, 0, 0, 294, 295, 5, 104, 0, 0, 295, 296, 5, 97, 0, 0, 296, 297, 5, 77, 0, 0, 297, 298, 5, 117, 0, 0, 298, 299, 5, 108, 0, 0, 299, 300, 5, 116, 0, 0, 300, 301, 5, 105, 0, 0, 301, 302, 5, 112, 0, 0, 302, 303, 5, 108, 0, 0, 303, 304, 5, 97, 0, 0, 304, 56, 1, 0, 0, 0, 305, 306, 5, 67, 0, 0, 306, 307, 5, 111, 0, 0, 307, 308, 5, 114, 0, 0, 308, 309, 5, 114, 0, 0, 309, 310, 5, 101, 0, 0, 310, 311, 5, 115, 0, 0, 311, 312, 5, 112, 0, 0, 312, 313, 5, 111, 0, 0, 313, 314, 5, 110, 0, 0, 314, 315, 5, 100, 0, 0, 315, 316, 5, 101, 0, 0, 316, 317, 5, 110, 0, 0, 317, 318, 5, 99, 0, 0, 318, 319, 5, 105, 0, 0, 319, 320, 5, 97, 0, 0, 320, 58, 1, 0, 0, 0, 321, 322, 5, 86, 0, 0, 322, 323, 5, 101, 0, 0, 323, 324, 5, 114, 0, 0, 324, 325, 5, 100, 0, 0, 325, 326, 5, 97, 0, 0, 326, 327, 5, 100, 0, 0, 327, 328, 5, 101, 0, 0, 328, 329, 5, 105, 0, 0, 329, 330, 5, 114, 0, 0, 330, 331, 5, 111, 0, 0, 331, 332, 5, 79, 0, 0, 332, 333, 5, 117, 0, 0, 333, 334, 5, 70, 0, 0, 334, 335, 5, 97, 0, 0, 335, 336, 5, 108, 0, 0, 336, 337, 5, 115, 0, 0, 337, 338, 5, 111, 0, 0, 338, 60, 1, 0, 0, 0, 339, 340, 5, 80, 0, 0, 340, 341, 5, 97, 0, 0, 341, 342, 5, 108, 0, 0, 342, 343, 5, 97, 0, 0, 343, 344, 5, 118, 0, 0, 344, 345, 5, 114, 0, 0, 345, 346, 5, 97, 0, 0, 346, 347, 5, 115, 0, 0, 347, 348, 5, 69, 0, 0, 348, 349, 5, 109, 0, 0, 349, 350, 5, 70, 0, 0, 350, 351, 5, 97, 0, 0, 351, 352, 5, 108, 0, 0, 352, 353, 5, 116, 0, 0, 353, 354, 5, 97, 0, 0, 354, 62, 1, 0, 0, 0, 355, 356, 5, 82, 0, 0, 356, 357, 5, 101, 0, 0, 357, 358, 5, 115, 0, 0, 358, 359, 5, 112, 0, 0, 359, 360, 5, 111, 0, 0, 360, 361, 5, 115, 0, 0, 361, 362, 5, 116, 0, 0, 362, 363, 5, 97, 0, 0, 363, 364, 5, 67, 0, 0, 364, 365, 5, 117, 0, 0, 365, 366, 5, 114, 0, 0, 366, 367, 5, 116, 0, 0, 367, 368, 5, 97, 0, 0, 368, 64, 1, 0, 0, 0, 369, 370, 5, 86, 0, 0, 370, 371, 5, 101, 0, 0, 371, 372, 5, 114, 0, 0, 372, 373, 5, 100, 0, 0, 373, 374, 5, 97, 0, 0, 374, 375, 5, 100, 0, 0, 375, 376, 5, 101, 0, 0, 376, 377, 5, 105, 0, 0, 377, 378, 5, 114, 0, 0, 378, 379, 5, 111, 0, 0, 379, 66, 1, 0, 0, 0, 380, 381, 5, 70, 0, 0, 381, 382, 5, 97, 0, 0, 382, 383, 5, 108, 0, 0, 383, 384, 5, 115, 0, 0, 384, 385, 5, 111, 0, 0, 385, 68, 1, 0, 0, 0, 386, 387, 5, 84, 0, 0, 387, 388, 5, 105, 0, 0, 388, 389, 5, 112, 0, 0, 389, 390, 5, 111, 0, 0, 390, 70, 1, 0, 0, 0, 391, 393, 7, 0, 0, 0, 392, 391, 1, 0, 0, 0, 393, 394, 1, 0, 0, 0, 394, 392, 1, 0, 0, 0, 394, 395, 1, 0, 0, 0, 395, 72, 1, 0, 0, 0, 396, 397, 5, 48, 0, 0, 397, 405, 7, 0, 0, 0, 398, 399, 5, 49, 0, 0, 399, 405, 7, 0, 0, 0, 400, 401, 5, 50, 0, 0, 401, 405, 7, 0, 0, 0, 402, 403, 5, 51, 0, 0, 403, 405, 7, 1, 0, 0, 404, 396, 1, 0, 0, 0, 404, 398, 1, 0, 0, 0, 404, 400, 1, 0, 0, 0, 404, 402, 1, 0, 0, 0, 405, 406, 1, 0, 0, 0, 406, 411, 5, 47, 0, 0, 407, 408, 5, 48, 0, 0, 408, 412, 7, 0, 0, 0, 409, 410, 5, 49, 0, 0, 410, 412, 7, 2, 0, 0, 411, 407, 1, 0, 0, 0, 411, 409, 1, 0, 0, 0, 412, 413, 1, 0, 0, 0, 413, 414, 5, 47, 0, 0, 414, 415, 7, 0, 0, 0, 415, 416, 7, 0, 0, 0, 416, 417, 7, 0, 0, 0, 417, 418, 7, 0, 0, 0, 418, 419, 1, 0, 0, 0, 419, 420, 5, 32, 0, 0, 420, 421, 7, 0, 0, 0, 421, 422, 7, 0, 0, 0, 422, 423, 1, 0, 0, 0, 423, 424, 3, 1, 0, 0, 424, 425, 7, 0, 0, 0, 425, 426, 7, 0, 0, 0, 426, 74, 1, 0, 0, 0, 427, 429, 7, 3, 0, 0, 428, 427, 1, 0, 0, 0, 429, 430, 1, 0, 0, 0, 430, 428, 1, 0, 0, 0, 430, 431, 1, 0, 0, 0, 431, 432, 1, 0, 0, 0, 432, 433, 6, 37, 0, 0, 433, 76, 1, 0, 0, 0, 434, 435, 7, 4, 0, 0, 435, 436, 1, 0, 0, 0, 436, 437, 6, 38, 0, 0, 437, 78, 1, 0, 0, 0, 438, 439, 3, 71, 35, 0, 439, 440, 5, 46, 0, 0, 440, 441, 3, 71, 35, 0, 441, 444, 1, 0, 0, 0, 442, 444, 3, 71, 35, 0, 443, 438, 1, 0, 0, 0, 443, 442, 1, 0, 0, 0, 444, 80, 1, 0, 0, 0, 445, 451, 3, 83, 41, 0, 446, 447, 3, 85, 42, 0, 447, 448, 3, 83, 41, 0, 448, 450, 1, 0, 0, 0, 449, 446, 1, 0, 0, 0, 450, 453, 1, 0, 0, 0, 451, 449, 1, 0, 0, 0, 451, 452, 1, 0, 0, 0, 452, 82, 1, 0, 0, 0, 453, 451, 1, 0, 0, 0, 454, 456, 7, 5, 0, 0, 455, 454, 1, 0, 0, 0, 456, 457, 1, 0, 0, 0, 457, 455, 1, 0, 0, 0, 457, 458, 1, 0, 0, 0, 458, 84, 1, 0, 0, 0, 459, 460, 7, 6, 0, 0, 460, 461, 1, 0, 0, 0, 461, 462, 6, 42, 0, 0, 462, 86, 1, 0, 0, 0, 8, 0, 394, 404, 411, 430, 443, 451, 457, 1, 6, 0, 0]

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameLexer.interp

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameLexer.txt

// Generated from C:/Users/mike\_/OneDrive/Documentos/sem4pi-22-23-61/base.core/src/main/java/eapli/base/ANTLR/ExamValidation\Exame.g4 by ANTLR 4.12.0  
package eapli.base.ANTLR.ExamValidation;  
import org.antlr.v4.runtime.Lexer;  
import org.antlr.v4.runtime.CharStream;  
import org.antlr.v4.runtime.Token;  
import org.antlr.v4.runtime.TokenStream;  
import org.antlr.v4.runtime.\*;  
import org.antlr.v4.runtime.atn.\*;  
import org.antlr.v4.runtime.dfa.DFA;  
import org.antlr.v4.runtime.misc.\*;  
  
@SuppressWarnings({"all", "warnings", "unchecked", "unused", "cast", "CheckReturnValue"})  
public class ExameLexer extends Lexer {  
 static { RuntimeMetaData.checkVersion("4.12.0", RuntimeMetaData.VERSION); }  
  
 protected static final DFA[] \_decisionToDFA;  
 protected static final PredictionContextCache \_sharedContextCache =  
 new PredictionContextCache();  
 public static final int  
 DOISPONTOS=1, ABRECHAVETA=2, FECHACHAVETA=3, ABREPARANTESIS=4, FECHAPARANTESIS=5,   
 VIRGULA=6, TESTE=7, ID=8, TRACO=9, RESPOSTA=10, TITULO=11, DATAABERTURA=12,   
 DATAFECHO=13, OPCOES=14, OPCAO=15, ENUNCIADO=16, TABELA\_A=17, TABELA\_B=18,   
 COTACAO=19, DIFICULDADE=20, DESCRICAO\_TEXTUAL=21, PERGUNTAS=22, LIMITE\_PERGUNTAS=23,   
 SECCAO=24, PERGUNTA=25, TIPO\_PERGUNTA=26, NUMERICA=27, ESCOLHA\_MULTIPLA=28,   
 CORRESPONDENCIA=29, VERDADEIROOUFALSO=30, PALAVRAS\_EM\_FALTA=31, RESPOSTA\_CURTA=32,   
 VERDADEIRO=33, FALSO=34, TIPO=35, INT=36, DATA=37, WS=38, CARACTERE\_ESPECIAL=39,   
 NUMERO=40, FRASE=41, STRING=42, ESPACO=43;  
 public static String[] channelNames = {  
 "DEFAULT\_TOKEN\_CHANNEL", "HIDDEN"  
 };  
  
 public static String[] modeNames = {  
 "DEFAULT\_MODE"  
 };  
  
 private static String[] makeRuleNames() {  
 return new String[] {  
 "DOISPONTOS", "ABRECHAVETA", "FECHACHAVETA", "ABREPARANTESIS", "FECHAPARANTESIS",   
 "VIRGULA", "TESTE", "ID", "TRACO", "RESPOSTA", "TITULO", "DATAABERTURA",   
 "DATAFECHO", "OPCOES", "OPCAO", "ENUNCIADO", "TABELA\_A", "TABELA\_B",   
 "COTACAO", "DIFICULDADE", "DESCRICAO\_TEXTUAL", "PERGUNTAS", "LIMITE\_PERGUNTAS",   
 "SECCAO", "PERGUNTA", "TIPO\_PERGUNTA", "NUMERICA", "ESCOLHA\_MULTIPLA",   
 "CORRESPONDENCIA", "VERDADEIROOUFALSO", "PALAVRAS\_EM\_FALTA", "RESPOSTA\_CURTA",   
 "VERDADEIRO", "FALSO", "TIPO", "INT", "DATA", "WS", "CARACTERE\_ESPECIAL",   
 "NUMERO", "FRASE", "STRING", "ESPACO"  
 };  
 }  
 public static final String[] ruleNames = makeRuleNames();  
  
 private static String[] makeLiteralNames() {  
 return new String[] {  
 null, "':'", "'{'", "'}'", "'['", "']'", "','", "'Teste'", "'ID'", "'-'",   
 "'Resposta'", "'Titulo'", "'DataAbertura'", "'DataFecho'", "'Opcoes'",   
 "'Opcao'", "'Enunciado'", "'TabelaA'", "'TabelaB'", "'Cotacao'", "'Dificuldade'",   
 "'DescricaoTextual'", "'Perguntas'", "'LimitePerguntas'", "'Seccao'",   
 "'Pergunta'", "'TipoPergunta'", "'Numerica'", "'EscolhaMultipla'", "'Correspondencia'",   
 "'VerdadeiroOuFalso'", "'PalavrasEmFalta'", "'RespostaCurta'", "'Verdadeiro'",   
 "'Falso'", "'Tipo'"  
 };  
 }  
 private static final String[] \_LITERAL\_NAMES = makeLiteralNames();  
 private static String[] makeSymbolicNames() {  
 return new String[] {  
 null, "DOISPONTOS", "ABRECHAVETA", "FECHACHAVETA", "ABREPARANTESIS",   
 "FECHAPARANTESIS", "VIRGULA", "TESTE", "ID", "TRACO", "RESPOSTA", "TITULO",   
 "DATAABERTURA", "DATAFECHO", "OPCOES", "OPCAO", "ENUNCIADO", "TABELA\_A",   
 "TABELA\_B", "COTACAO", "DIFICULDADE", "DESCRICAO\_TEXTUAL", "PERGUNTAS",   
 "LIMITE\_PERGUNTAS", "SECCAO", "PERGUNTA", "TIPO\_PERGUNTA", "NUMERICA",   
 "ESCOLHA\_MULTIPLA", "CORRESPONDENCIA", "VERDADEIROOUFALSO", "PALAVRAS\_EM\_FALTA",   
 "RESPOSTA\_CURTA", "VERDADEIRO", "FALSO", "TIPO", "INT", "DATA", "WS",   
 "CARACTERE\_ESPECIAL", "NUMERO", "FRASE", "STRING", "ESPACO"  
 };  
 }  
 private static final String[] \_SYMBOLIC\_NAMES = makeSymbolicNames();  
 public static final Vocabulary VOCABULARY = new VocabularyImpl(\_LITERAL\_NAMES, \_SYMBOLIC\_NAMES);  
  
 /\*\*  
 \* @deprecated Use {@link #VOCABULARY} instead.  
 \*/  
 @Deprecated  
 public static final String[] tokenNames;  
 static {  
 tokenNames = new String[\_SYMBOLIC\_NAMES.length];  
 for (int i = 0; i < tokenNames.length; i++) {  
 tokenNames[i] = VOCABULARY.getLiteralName(i);  
 if (tokenNames[i] == null) {  
 tokenNames[i] = VOCABULARY.getSymbolicName(i);  
 }  
  
 if (tokenNames[i] == null) {  
 tokenNames[i] = "<INVALID>";  
 }  
 }  
 }  
  
 @Override  
 @Deprecated  
 public String[] getTokenNames() {  
 return tokenNames;  
 }  
  
 @Override  
  
 public Vocabulary getVocabulary() {  
 return VOCABULARY;  
 }  
  
  
 public ExameLexer(CharStream input) {  
 super(input);  
 \_interp = new LexerATNSimulator(this,\_ATN,\_decisionToDFA,\_sharedContextCache);  
 }  
  
 @Override  
 public String getGrammarFileName() { return "Exame.g4"; }  
  
 @Override  
 public String[] getRuleNames() { return ruleNames; }  
  
 @Override  
 public String getSerializedATN() { return \_serializedATN; }  
  
 @Override  
 public String[] getChannelNames() { return channelNames; }  
  
 @Override  
 public String[] getModeNames() { return modeNames; }  
  
 @Override  
 public ATN getATN() { return \_ATN; }  
  
 public static final String \_serializedATN =  
 "\u0004\u0000+\u01cf\u0006\uffff\uffff\u0002\u0000\u0007\u0000\u0002\u0001"+  
 "\u0007\u0001\u0002\u0002\u0007\u0002\u0002\u0003\u0007\u0003\u0002\u0004"+  
 "\u0007\u0004\u0002\u0005\u0007\u0005\u0002\u0006\u0007\u0006\u0002\u0007"+  
 "\u0007\u0007\u0002\b\u0007\b\u0002\t\u0007\t\u0002\n\u0007\n\u0002\u000b"+  
 "\u0007\u000b\u0002\f\u0007\f\u0002\r\u0007\r\u0002\u000e\u0007\u000e\u0002"+  
 "\u000f\u0007\u000f\u0002\u0010\u0007\u0010\u0002\u0011\u0007\u0011\u0002"+  
 "\u0012\u0007\u0012\u0002\u0013\u0007\u0013\u0002\u0014\u0007\u0014\u0002"+  
 "\u0015\u0007\u0015\u0002\u0016\u0007\u0016\u0002\u0017\u0007\u0017\u0002"+  
 "\u0018\u0007\u0018\u0002\u0019\u0007\u0019\u0002\u001a\u0007\u001a\u0002"+  
 "\u001b\u0007\u001b\u0002\u001c\u0007\u001c\u0002\u001d\u0007\u001d\u0002"+  
 "\u001e\u0007\u001e\u0002\u001f\u0007\u001f\u0002 \u0007 \u0002!\u0007"+  
 "!\u0002\"\u0007\"\u0002#\u0007#\u0002$\u0007$\u0002%\u0007%\u0002&\u0007"+  
 "&\u0002\'\u0007\'\u0002(\u0007(\u0002)\u0007)\u0002\*\u0007\*\u0001\u0000"+  
 "\u0001\u0000\u0001\u0001\u0001\u0001\u0001\u0002\u0001\u0002\u0001\u0003"+  
 "\u0001\u0003\u0001\u0004\u0001\u0004\u0001\u0005\u0001\u0005\u0001\u0006"+  
 "\u0001\u0006\u0001\u0006\u0001\u0006\u0001\u0006\u0001\u0006\u0001\u0007"+  
 "\u0001\u0007\u0001\u0007\u0001\b\u0001\b\u0001\t\u0001\t\u0001\t\u0001"+  
 "\t\u0001\t\u0001\t\u0001\t\u0001\t\u0001\t\u0001\n\u0001\n\u0001\n\u0001"+  
 "\n\u0001\n\u0001\n\u0001\n\u0001\u000b\u0001\u000b\u0001\u000b\u0001\u000b"+  
 "\u0001\u000b\u0001\u000b\u0001\u000b\u0001\u000b\u0001\u000b\u0001\u000b"+  
 "\u0001\u000b\u0001\u000b\u0001\u000b\u0001\f\u0001\f\u0001\f\u0001\f\u0001"+  
 "\f\u0001\f\u0001\f\u0001\f\u0001\f\u0001\f\u0001\r\u0001\r\u0001\r\u0001"+  
 "\r\u0001\r\u0001\r\u0001\r\u0001\u000e\u0001\u000e\u0001\u000e\u0001\u000e"+  
 "\u0001\u000e\u0001\u000e\u0001\u000f\u0001\u000f\u0001\u000f\u0001\u000f"+  
 "\u0001\u000f\u0001\u000f\u0001\u000f\u0001\u000f\u0001\u000f\u0001\u000f"+  
 "\u0001\u0010\u0001\u0010\u0001\u0010\u0001\u0010\u0001\u0010\u0001\u0010"+  
 "\u0001\u0010\u0001\u0010\u0001\u0011\u0001\u0011\u0001\u0011\u0001\u0011"+  
 "\u0001\u0011\u0001\u0011\u0001\u0011\u0001\u0011\u0001\u0012\u0001\u0012"+  
 "\u0001\u0012\u0001\u0012\u0001\u0012\u0001\u0012\u0001\u0012\u0001\u0012"+  
 "\u0001\u0013\u0001\u0013\u0001\u0013\u0001\u0013\u0001\u0013\u0001\u0013"+  
 "\u0001\u0013\u0001\u0013\u0001\u0013\u0001\u0013\u0001\u0013\u0001\u0013"+  
 "\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014"+  
 "\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014"+  
 "\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0014\u0001\u0015"+  
 "\u0001\u0015\u0001\u0015\u0001\u0015\u0001\u0015\u0001\u0015\u0001\u0015"+  
 "\u0001\u0015\u0001\u0015\u0001\u0015\u0001\u0016\u0001\u0016\u0001\u0016"+  
 "\u0001\u0016\u0001\u0016\u0001\u0016\u0001\u0016\u0001\u0016\u0001\u0016"+  
 "\u0001\u0016\u0001\u0016\u0001\u0016\u0001\u0016\u0001\u0016\u0001\u0016"+  
 "\u0001\u0016\u0001\u0017\u0001\u0017\u0001\u0017\u0001\u0017\u0001\u0017"+  
 "\u0001\u0017\u0001\u0017\u0001\u0018\u0001\u0018\u0001\u0018\u0001\u0018"+  
 "\u0001\u0018\u0001\u0018\u0001\u0018\u0001\u0018\u0001\u0018\u0001\u0019"+  
 "\u0001\u0019\u0001\u0019\u0001\u0019\u0001\u0019\u0001\u0019\u0001\u0019"+  
 "\u0001\u0019\u0001\u0019\u0001\u0019\u0001\u0019\u0001\u0019\u0001\u0019"+  
 "\u0001\u001a\u0001\u001a\u0001\u001a\u0001\u001a\u0001\u001a\u0001\u001a"+  
 "\u0001\u001a\u0001\u001a\u0001\u001a\u0001\u001b\u0001\u001b\u0001\u001b"+  
 "\u0001\u001b\u0001\u001b\u0001\u001b\u0001\u001b\u0001\u001b\u0001\u001b"+  
 "\u0001\u001b\u0001\u001b\u0001\u001b\u0001\u001b\u0001\u001b\u0001\u001b"+  
 "\u0001\u001b\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001c"+  
 "\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001c"+  
 "\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001c\u0001\u001d"+  
 "\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d"+  
 "\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d"+  
 "\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001d\u0001\u001e"+  
 "\u0001\u001e\u0001\u001e\u0001\u001e\u0001\u001e\u0001\u001e\u0001\u001e"+  
 "\u0001\u001e\u0001\u001e\u0001\u001e\u0001\u001e\u0001\u001e\u0001\u001e"+  
 "\u0001\u001e\u0001\u001e\u0001\u001e\u0001\u001f\u0001\u001f\u0001\u001f"+  
 "\u0001\u001f\u0001\u001f\u0001\u001f\u0001\u001f\u0001\u001f\u0001\u001f"+  
 "\u0001\u001f\u0001\u001f\u0001\u001f\u0001\u001f\u0001\u001f\u0001 \u0001"+  
 " \u0001 \u0001 \u0001 \u0001 \u0001 \u0001 \u0001 \u0001 \u0001 \u0001"+  
 "!\u0001!\u0001!\u0001!\u0001!\u0001!\u0001\"\u0001\"\u0001\"\u0001\"\u0001"+  
 "\"\u0001#\u0004#\u0189\b#\u000b#\f#\u018a\u0001$\u0001$\u0001$\u0001$"+  
 "\u0001$\u0001$\u0001$\u0001$\u0003$\u0195\b$\u0001$\u0001$\u0001$\u0001"+  
 "$\u0001$\u0003$\u019c\b$\u0001$\u0001$\u0001$\u0001$\u0001$\u0001$\u0001"+  
 "$\u0001$\u0001$\u0001$\u0001$\u0001$\u0001$\u0001$\u0001%\u0004%\u01ad"+  
 "\b%\u000b%\f%\u01ae\u0001%\u0001%\u0001&\u0001&\u0001&\u0001&\u0001\'"+  
 "\u0001\'\u0001\'\u0001\'\u0001\'\u0003\'\u01bc\b\'\u0001(\u0001(\u0001"+  
 "(\u0001(\u0005(\u01c2\b(\n(\f(\u01c5\t(\u0001)\u0004)\u01c8\b)\u000b)"+  
 "\f)\u01c9\u0001\*\u0001\*\u0001\*\u0001\*\u0000\u0000+\u0001\u0001\u0003\u0002"+  
 "\u0005\u0003\u0007\u0004\t\u0005\u000b\u0006\r\u0007\u000f\b\u0011\t\u0013"+  
 "\n\u0015\u000b\u0017\f\u0019\r\u001b\u000e\u001d\u000f\u001f\u0010!\u0011"+  
 "#\u0012%\u0013\'\u0014)\u0015+\u0016-\u0017/\u00181\u00193\u001a5\u001b"+  
 "7\u001c9\u001d;\u001e=\u001f? A!C\"E#G$I%K&M\'O(Q)S\*U+\u0001\u0000\u0007"+  
 "\u0001\u000009\u0001\u000001\u0001\u000002\u0003\u0000\n\n\r\r \u0006"+  
 "\u0000\"#\')++--//\_\_\u0002\u0000AZaz\u0002\u0000\t\t \u01d7\u0000\u0001"+  
 "\u0001\u0000\u0000\u0000\u0000\u0003\u0001\u0000\u0000\u0000\u0000\u0005"+  
 "\u0001\u0000\u0000\u0000\u0000\u0007\u0001\u0000\u0000\u0000\u0000\t\u0001"+  
 "\u0000\u0000\u0000\u0000\u000b\u0001\u0000\u0000\u0000\u0000\r\u0001\u0000"+  
 "\u0000\u0000\u0000\u000f\u0001\u0000\u0000\u0000\u0000\u0011\u0001\u0000"+  
 "\u0000\u0000\u0000\u0013\u0001\u0000\u0000\u0000\u0000\u0015\u0001\u0000"+  
 "\u0000\u0000\u0000\u0017\u0001\u0000\u0000\u0000\u0000\u0019\u0001\u0000"+  
 "\u0000\u0000\u0000\u001b\u0001\u0000\u0000\u0000\u0000\u001d\u0001\u0000"+  
 "\u0000\u0000\u0000\u001f\u0001\u0000\u0000\u0000\u0000!\u0001\u0000\u0000"+  
 "\u0000\u0000#\u0001\u0000\u0000\u0000\u0000%\u0001\u0000\u0000\u0000\u0000"+  
 "\'\u0001\u0000\u0000\u0000\u0000)\u0001\u0000\u0000\u0000\u0000+\u0001"+  
 "\u0000\u0000\u0000\u0000-\u0001\u0000\u0000\u0000\u0000/\u0001\u0000\u0000"+  
 "\u0000\u00001\u0001\u0000\u0000\u0000\u00003\u0001\u0000\u0000\u0000\u0000"+  
 "5\u0001\u0000\u0000\u0000\u00007\u0001\u0000\u0000\u0000\u00009\u0001"+  
 "\u0000\u0000\u0000\u0000;\u0001\u0000\u0000\u0000\u0000=\u0001\u0000\u0000"+  
 "\u0000\u0000?\u0001\u0000\u0000\u0000\u0000A\u0001\u0000\u0000\u0000\u0000"+  
 "C\u0001\u0000\u0000\u0000\u0000E\u0001\u0000\u0000\u0000\u0000G\u0001"+  
 "\u0000\u0000\u0000\u0000I\u0001\u0000\u0000\u0000\u0000K\u0001\u0000\u0000"+  
 "\u0000\u0000M\u0001\u0000\u0000\u0000\u0000O\u0001\u0000\u0000\u0000\u0000"+  
 "Q\u0001\u0000\u0000\u0000\u0000S\u0001\u0000\u0000\u0000\u0000U\u0001"+  
 "\u0000\u0000\u0000\u0001W\u0001\u0000\u0000\u0000\u0003Y\u0001\u0000\u0000"+  
 "\u0000\u0005[\u0001\u0000\u0000\u0000\u0007]\u0001\u0000\u0000\u0000\t"+  
 "\_\u0001\u0000\u0000\u0000\u000ba\u0001\u0000\u0000\u0000\rc\u0001\u0000"+  
 "\u0000\u0000\u000fi\u0001\u0000\u0000\u0000\u0011l\u0001\u0000\u0000\u0000"+  
 "\u0013n\u0001\u0000\u0000\u0000\u0015w\u0001\u0000\u0000\u0000\u0017~"+  
 "\u0001\u0000\u0000\u0000\u0019\u008b\u0001\u0000\u0000\u0000\u001b\u0095"+  
 "\u0001\u0000\u0000\u0000\u001d\u009c\u0001\u0000\u0000\u0000\u001f\u00a2"+  
 "\u0001\u0000\u0000\u0000!\u00ac\u0001\u0000\u0000\u0000#\u00b4\u0001\u0000"+  
 "\u0000\u0000%\u00bc\u0001\u0000\u0000\u0000\'\u00c4\u0001\u0000\u0000"+  
 "\u0000)\u00d0\u0001\u0000\u0000\u0000+\u00e1\u0001\u0000\u0000\u0000-"+  
 "\u00eb\u0001\u0000\u0000\u0000/\u00fb\u0001\u0000\u0000\u00001\u0102\u0001"+  
 "\u0000\u0000\u00003\u010b\u0001\u0000\u0000\u00005\u0118\u0001\u0000\u0000"+  
 "\u00007\u0121\u0001\u0000\u0000\u00009\u0131\u0001\u0000\u0000\u0000;"+  
 "\u0141\u0001\u0000\u0000\u0000=\u0153\u0001\u0000\u0000\u0000?\u0163\u0001"+  
 "\u0000\u0000\u0000A\u0171\u0001\u0000\u0000\u0000C\u017c\u0001\u0000\u0000"+  
 "\u0000E\u0182\u0001\u0000\u0000\u0000G\u0188\u0001\u0000\u0000\u0000I"+  
 "\u0194\u0001\u0000\u0000\u0000K\u01ac\u0001\u0000\u0000\u0000M\u01b2\u0001"+  
 "\u0000\u0000\u0000O\u01bb\u0001\u0000\u0000\u0000Q\u01bd\u0001\u0000\u0000"+  
 "\u0000S\u01c7\u0001\u0000\u0000\u0000U\u01cb\u0001\u0000\u0000\u0000W"+  
 "X\u0005:\u0000\u0000X\u0002\u0001\u0000\u0000\u0000YZ\u0005{\u0000\u0000"+  
 "Z\u0004\u0001\u0000\u0000\u0000[\\\u0005}\u0000\u0000\\\u0006\u0001\u0000"+  
 "\u0000\u0000]^\u0005[\u0000\u0000^\b\u0001\u0000\u0000\u0000\_`\u0005]"+  
 "\u0000\u0000`\n\u0001\u0000\u0000\u0000ab\u0005,\u0000\u0000b\f\u0001"+  
 "\u0000\u0000\u0000cd\u0005T\u0000\u0000de\u0005e\u0000\u0000ef\u0005s"+  
 "\u0000\u0000fg\u0005t\u0000\u0000gh\u0005e\u0000\u0000h\u000e\u0001\u0000"+  
 "\u0000\u0000ij\u0005I\u0000\u0000jk\u0005D\u0000\u0000k\u0010\u0001\u0000"+  
 "\u0000\u0000lm\u0005-\u0000\u0000m\u0012\u0001\u0000\u0000\u0000no\u0005"+  
 "R\u0000\u0000op\u0005e\u0000\u0000pq\u0005s\u0000\u0000qr\u0005p\u0000"+  
 "\u0000rs\u0005o\u0000\u0000st\u0005s\u0000\u0000tu\u0005t\u0000\u0000"+  
 "uv\u0005a\u0000\u0000v\u0014\u0001\u0000\u0000\u0000wx\u0005T\u0000\u0000"+  
 "xy\u0005i\u0000\u0000yz\u0005t\u0000\u0000z{\u0005u\u0000\u0000{|\u0005"+  
 "l\u0000\u0000|}\u0005o\u0000\u0000}\u0016\u0001\u0000\u0000\u0000~\u007f"+  
 "\u0005D\u0000\u0000\u007f\u0080\u0005a\u0000\u0000\u0080\u0081\u0005t"+  
 "\u0000\u0000\u0081\u0082\u0005a\u0000\u0000\u0082\u0083\u0005A\u0000\u0000"+  
 "\u0083\u0084\u0005b\u0000\u0000\u0084\u0085\u0005e\u0000\u0000\u0085\u0086"+  
 "\u0005r\u0000\u0000\u0086\u0087\u0005t\u0000\u0000\u0087\u0088\u0005u"+  
 "\u0000\u0000\u0088\u0089\u0005r\u0000\u0000\u0089\u008a\u0005a\u0000\u0000"+  
 "\u008a\u0018\u0001\u0000\u0000\u0000\u008b\u008c\u0005D\u0000\u0000\u008c"+  
 "\u008d\u0005a\u0000\u0000\u008d\u008e\u0005t\u0000\u0000\u008e\u008f\u0005"+  
 "a\u0000\u0000\u008f\u0090\u0005F\u0000\u0000\u0090\u0091\u0005e\u0000"+  
 "\u0000\u0091\u0092\u0005c\u0000\u0000\u0092\u0093\u0005h\u0000\u0000\u0093"+  
 "\u0094\u0005o\u0000\u0000\u0094\u001a\u0001\u0000\u0000\u0000\u0095\u0096"+  
 "\u0005O\u0000\u0000\u0096\u0097\u0005p\u0000\u0000\u0097\u0098\u0005c"+  
 "\u0000\u0000\u0098\u0099\u0005o\u0000\u0000\u0099\u009a\u0005e\u0000\u0000"+  
 "\u009a\u009b\u0005s\u0000\u0000\u009b\u001c\u0001\u0000\u0000\u0000\u009c"+  
 "\u009d\u0005O\u0000\u0000\u009d\u009e\u0005p\u0000\u0000\u009e\u009f\u0005"+  
 "c\u0000\u0000\u009f\u00a0\u0005a\u0000\u0000\u00a0\u00a1\u0005o\u0000"+  
 "\u0000\u00a1\u001e\u0001\u0000\u0000\u0000\u00a2\u00a3\u0005E\u0000\u0000"+  
 "\u00a3\u00a4\u0005n\u0000\u0000\u00a4\u00a5\u0005u\u0000\u0000\u00a5\u00a6"+  
 "\u0005n\u0000\u0000\u00a6\u00a7\u0005c\u0000\u0000\u00a7\u00a8\u0005i"+  
 "\u0000\u0000\u00a8\u00a9\u0005a\u0000\u0000\u00a9\u00aa\u0005d\u0000\u0000"+  
 "\u00aa\u00ab\u0005o\u0000\u0000\u00ab \u0001\u0000\u0000\u0000\u00ac\u00ad"+  
 "\u0005T\u0000\u0000\u00ad\u00ae\u0005a\u0000\u0000\u00ae\u00af\u0005b"+  
 "\u0000\u0000\u00af\u00b0\u0005e\u0000\u0000\u00b0\u00b1\u0005l\u0000\u0000"+  
 "\u00b1\u00b2\u0005a\u0000\u0000\u00b2\u00b3\u0005A\u0000\u0000\u00b3\""+  
 "\u0001\u0000\u0000\u0000\u00b4\u00b5\u0005T\u0000\u0000\u00b5\u00b6\u0005"+  
 "a\u0000\u0000\u00b6\u00b7\u0005b\u0000\u0000\u00b7\u00b8\u0005e\u0000"+  
 "\u0000\u00b8\u00b9\u0005l\u0000\u0000\u00b9\u00ba\u0005a\u0000\u0000\u00ba"+  
 "\u00bb\u0005B\u0000\u0000\u00bb$\u0001\u0000\u0000\u0000\u00bc\u00bd\u0005"+  
 "C\u0000\u0000\u00bd\u00be\u0005o\u0000\u0000\u00be\u00bf\u0005t\u0000"+  
 "\u0000\u00bf\u00c0\u0005a\u0000\u0000\u00c0\u00c1\u0005c\u0000\u0000\u00c1"+  
 "\u00c2\u0005a\u0000\u0000\u00c2\u00c3\u0005o\u0000\u0000\u00c3&\u0001"+  
 "\u0000\u0000\u0000\u00c4\u00c5\u0005D\u0000\u0000\u00c5\u00c6\u0005i\u0000"+  
 "\u0000\u00c6\u00c7\u0005f\u0000\u0000\u00c7\u00c8\u0005i\u0000\u0000\u00c8"+  
 "\u00c9\u0005c\u0000\u0000\u00c9\u00ca\u0005u\u0000\u0000\u00ca\u00cb\u0005"+  
 "l\u0000\u0000\u00cb\u00cc\u0005d\u0000\u0000\u00cc\u00cd\u0005a\u0000"+  
 "\u0000\u00cd\u00ce\u0005d\u0000\u0000\u00ce\u00cf\u0005e\u0000\u0000\u00cf"+  
 "(\u0001\u0000\u0000\u0000\u00d0\u00d1\u0005D\u0000\u0000\u00d1\u00d2\u0005"+  
 "e\u0000\u0000\u00d2\u00d3\u0005s\u0000\u0000\u00d3\u00d4\u0005c\u0000"+  
 "\u0000\u00d4\u00d5\u0005r\u0000\u0000\u00d5\u00d6\u0005i\u0000\u0000\u00d6"+  
 "\u00d7\u0005c\u0000\u0000\u00d7\u00d8\u0005a\u0000\u0000\u00d8\u00d9\u0005"+  
 "o\u0000\u0000\u00d9\u00da\u0005T\u0000\u0000\u00da\u00db\u0005e\u0000"+  
 "\u0000\u00db\u00dc\u0005x\u0000\u0000\u00dc\u00dd\u0005t\u0000\u0000\u00dd"+  
 "\u00de\u0005u\u0000\u0000\u00de\u00df\u0005a\u0000\u0000\u00df\u00e0\u0005"+  
 "l\u0000\u0000\u00e0\*\u0001\u0000\u0000\u0000\u00e1\u00e2\u0005P\u0000"+  
 "\u0000\u00e2\u00e3\u0005e\u0000\u0000\u00e3\u00e4\u0005r\u0000\u0000\u00e4"+  
 "\u00e5\u0005g\u0000\u0000\u00e5\u00e6\u0005u\u0000\u0000\u00e6\u00e7\u0005"+  
 "n\u0000\u0000\u00e7\u00e8\u0005t\u0000\u0000\u00e8\u00e9\u0005a\u0000"+  
 "\u0000\u00e9\u00ea\u0005s\u0000\u0000\u00ea,\u0001\u0000\u0000\u0000\u00eb"+  
 "\u00ec\u0005L\u0000\u0000\u00ec\u00ed\u0005i\u0000\u0000\u00ed\u00ee\u0005"+  
 "m\u0000\u0000\u00ee\u00ef\u0005i\u0000\u0000\u00ef\u00f0\u0005t\u0000"+  
 "\u0000\u00f0\u00f1\u0005e\u0000\u0000\u00f1\u00f2\u0005P\u0000\u0000\u00f2"+  
 "\u00f3\u0005e\u0000\u0000\u00f3\u00f4\u0005r\u0000\u0000\u00f4\u00f5\u0005"+  
 "g\u0000\u0000\u00f5\u00f6\u0005u\u0000\u0000\u00f6\u00f7\u0005n\u0000"+  
 "\u0000\u00f7\u00f8\u0005t\u0000\u0000\u00f8\u00f9\u0005a\u0000\u0000\u00f9"+  
 "\u00fa\u0005s\u0000\u0000\u00fa.\u0001\u0000\u0000\u0000\u00fb\u00fc\u0005"+  
 "S\u0000\u0000\u00fc\u00fd\u0005e\u0000\u0000\u00fd\u00fe\u0005c\u0000"+  
 "\u0000\u00fe\u00ff\u0005c\u0000\u0000\u00ff\u0100\u0005a\u0000\u0000\u0100"+  
 "\u0101\u0005o\u0000\u0000\u01010\u0001\u0000\u0000\u0000\u0102\u0103\u0005"+  
 "P\u0000\u0000\u0103\u0104\u0005e\u0000\u0000\u0104\u0105\u0005r\u0000"+  
 "\u0000\u0105\u0106\u0005g\u0000\u0000\u0106\u0107\u0005u\u0000\u0000\u0107"+  
 "\u0108\u0005n\u0000\u0000\u0108\u0109\u0005t\u0000\u0000\u0109\u010a\u0005"+  
 "a\u0000\u0000\u010a2\u0001\u0000\u0000\u0000\u010b\u010c\u0005T\u0000"+  
 "\u0000\u010c\u010d\u0005i\u0000\u0000\u010d\u010e\u0005p\u0000\u0000\u010e"+  
 "\u010f\u0005o\u0000\u0000\u010f\u0110\u0005P\u0000\u0000\u0110\u0111\u0005"+  
 "e\u0000\u0000\u0111\u0112\u0005r\u0000\u0000\u0112\u0113\u0005g\u0000"+  
 "\u0000\u0113\u0114\u0005u\u0000\u0000\u0114\u0115\u0005n\u0000\u0000\u0115"+  
 "\u0116\u0005t\u0000\u0000\u0116\u0117\u0005a\u0000\u0000\u01174\u0001"+  
 "\u0000\u0000\u0000\u0118\u0119\u0005N\u0000\u0000\u0119\u011a\u0005u\u0000"+  
 "\u0000\u011a\u011b\u0005m\u0000\u0000\u011b\u011c\u0005e\u0000\u0000\u011c"+  
 "\u011d\u0005r\u0000\u0000\u011d\u011e\u0005i\u0000\u0000\u011e\u011f\u0005"+  
 "c\u0000\u0000\u011f\u0120\u0005a\u0000\u0000\u01206\u0001\u0000\u0000"+  
 "\u0000\u0121\u0122\u0005E\u0000\u0000\u0122\u0123\u0005s\u0000\u0000\u0123"+  
 "\u0124\u0005c\u0000\u0000\u0124\u0125\u0005o\u0000\u0000\u0125\u0126\u0005"+  
 "l\u0000\u0000\u0126\u0127\u0005h\u0000\u0000\u0127\u0128\u0005a\u0000"+  
 "\u0000\u0128\u0129\u0005M\u0000\u0000\u0129\u012a\u0005u\u0000\u0000\u012a"+  
 "\u012b\u0005l\u0000\u0000\u012b\u012c\u0005t\u0000\u0000\u012c\u012d\u0005"+  
 "i\u0000\u0000\u012d\u012e\u0005p\u0000\u0000\u012e\u012f\u0005l\u0000"+  
 "\u0000\u012f\u0130\u0005a\u0000\u0000\u01308\u0001\u0000\u0000\u0000\u0131"+  
 "\u0132\u0005C\u0000\u0000\u0132\u0133\u0005o\u0000\u0000\u0133\u0134\u0005"+  
 "r\u0000\u0000\u0134\u0135\u0005r\u0000\u0000\u0135\u0136\u0005e\u0000"+  
 "\u0000\u0136\u0137\u0005s\u0000\u0000\u0137\u0138\u0005p\u0000\u0000\u0138"+  
 "\u0139\u0005o\u0000\u0000\u0139\u013a\u0005n\u0000\u0000\u013a\u013b\u0005"+  
 "d\u0000\u0000\u013b\u013c\u0005e\u0000\u0000\u013c\u013d\u0005n\u0000"+  
 "\u0000\u013d\u013e\u0005c\u0000\u0000\u013e\u013f\u0005i\u0000\u0000\u013f"+  
 "\u0140\u0005a\u0000\u0000\u0140:\u0001\u0000\u0000\u0000\u0141\u0142\u0005"+  
 "V\u0000\u0000\u0142\u0143\u0005e\u0000\u0000\u0143\u0144\u0005r\u0000"+  
 "\u0000\u0144\u0145\u0005d\u0000\u0000\u0145\u0146\u0005a\u0000\u0000\u0146"+  
 "\u0147\u0005d\u0000\u0000\u0147\u0148\u0005e\u0000\u0000\u0148\u0149\u0005"+  
 "i\u0000\u0000\u0149\u014a\u0005r\u0000\u0000\u014a\u014b\u0005o\u0000"+  
 "\u0000\u014b\u014c\u0005O\u0000\u0000\u014c\u014d\u0005u\u0000\u0000\u014d"+  
 "\u014e\u0005F\u0000\u0000\u014e\u014f\u0005a\u0000\u0000\u014f\u0150\u0005"+  
 "l\u0000\u0000\u0150\u0151\u0005s\u0000\u0000\u0151\u0152\u0005o\u0000"+  
 "\u0000\u0152<\u0001\u0000\u0000\u0000\u0153\u0154\u0005P\u0000\u0000\u0154"+  
 "\u0155\u0005a\u0000\u0000\u0155\u0156\u0005l\u0000\u0000\u0156\u0157\u0005"+  
 "a\u0000\u0000\u0157\u0158\u0005v\u0000\u0000\u0158\u0159\u0005r\u0000"+  
 "\u0000\u0159\u015a\u0005a\u0000\u0000\u015a\u015b\u0005s\u0000\u0000\u015b"+  
 "\u015c\u0005E\u0000\u0000\u015c\u015d\u0005m\u0000\u0000\u015d\u015e\u0005"+  
 "F\u0000\u0000\u015e\u015f\u0005a\u0000\u0000\u015f\u0160\u0005l\u0000"+  
 "\u0000\u0160\u0161\u0005t\u0000\u0000\u0161\u0162\u0005a\u0000\u0000\u0162"+  
 ">\u0001\u0000\u0000\u0000\u0163\u0164\u0005R\u0000\u0000\u0164\u0165\u0005"+  
 "e\u0000\u0000\u0165\u0166\u0005s\u0000\u0000\u0166\u0167\u0005p\u0000"+  
 "\u0000\u0167\u0168\u0005o\u0000\u0000\u0168\u0169\u0005s\u0000\u0000\u0169"+  
 "\u016a\u0005t\u0000\u0000\u016a\u016b\u0005a\u0000\u0000\u016b\u016c\u0005"+  
 "C\u0000\u0000\u016c\u016d\u0005u\u0000\u0000\u016d\u016e\u0005r\u0000"+  
 "\u0000\u016e\u016f\u0005t\u0000\u0000\u016f\u0170\u0005a\u0000\u0000\u0170"+  
 "@\u0001\u0000\u0000\u0000\u0171\u0172\u0005V\u0000\u0000\u0172\u0173\u0005"+  
 "e\u0000\u0000\u0173\u0174\u0005r\u0000\u0000\u0174\u0175\u0005d\u0000"+  
 "\u0000\u0175\u0176\u0005a\u0000\u0000\u0176\u0177\u0005d\u0000\u0000\u0177"+  
 "\u0178\u0005e\u0000\u0000\u0178\u0179\u0005i\u0000\u0000\u0179\u017a\u0005"+  
 "r\u0000\u0000\u017a\u017b\u0005o\u0000\u0000\u017bB\u0001\u0000\u0000"+  
 "\u0000\u017c\u017d\u0005F\u0000\u0000\u017d\u017e\u0005a\u0000\u0000\u017e"+  
 "\u017f\u0005l\u0000\u0000\u017f\u0180\u0005s\u0000\u0000\u0180\u0181\u0005"+  
 "o\u0000\u0000\u0181D\u0001\u0000\u0000\u0000\u0182\u0183\u0005T\u0000"+  
 "\u0000\u0183\u0184\u0005i\u0000\u0000\u0184\u0185\u0005p\u0000\u0000\u0185"+  
 "\u0186\u0005o\u0000\u0000\u0186F\u0001\u0000\u0000\u0000\u0187\u0189\u0007"+  
 "\u0000\u0000\u0000\u0188\u0187\u0001\u0000\u0000\u0000\u0189\u018a\u0001"+  
 "\u0000\u0000\u0000\u018a\u0188\u0001\u0000\u0000\u0000\u018a\u018b\u0001"+  
 "\u0000\u0000\u0000\u018bH\u0001\u0000\u0000\u0000\u018c\u018d\u00050\u0000"+  
 "\u0000\u018d\u0195\u0007\u0000\u0000\u0000\u018e\u018f\u00051\u0000\u0000"+  
 "\u018f\u0195\u0007\u0000\u0000\u0000\u0190\u0191\u00052\u0000\u0000\u0191"+  
 "\u0195\u0007\u0000\u0000\u0000\u0192\u0193\u00053\u0000\u0000\u0193\u0195"+  
 "\u0007\u0001\u0000\u0000\u0194\u018c\u0001\u0000\u0000\u0000\u0194\u018e"+  
 "\u0001\u0000\u0000\u0000\u0194\u0190\u0001\u0000\u0000\u0000\u0194\u0192"+  
 "\u0001\u0000\u0000\u0000\u0195\u0196\u0001\u0000\u0000\u0000\u0196\u019b"+  
 "\u0005/\u0000\u0000\u0197\u0198\u00050\u0000\u0000\u0198\u019c\u0007\u0000"+  
 "\u0000\u0000\u0199\u019a\u00051\u0000\u0000\u019a\u019c\u0007\u0002\u0000"+  
 "\u0000\u019b\u0197\u0001\u0000\u0000\u0000\u019b\u0199\u0001\u0000\u0000"+  
 "\u0000\u019c\u019d\u0001\u0000\u0000\u0000\u019d\u019e\u0005/\u0000\u0000"+  
 "\u019e\u019f\u0007\u0000\u0000\u0000\u019f\u01a0\u0007\u0000\u0000\u0000"+  
 "\u01a0\u01a1\u0007\u0000\u0000\u0000\u01a1\u01a2\u0007\u0000\u0000\u0000"+  
 "\u01a2\u01a3\u0001\u0000\u0000\u0000\u01a3\u01a4\u0005 \u0000\u0000\u01a4"+  
 "\u01a5\u0007\u0000\u0000\u0000\u01a5\u01a6\u0007\u0000\u0000\u0000\u01a6"+  
 "\u01a7\u0001\u0000\u0000\u0000\u01a7\u01a8\u0003\u0001\u0000\u0000\u01a8"+  
 "\u01a9\u0007\u0000\u0000\u0000\u01a9\u01aa\u0007\u0000\u0000\u0000\u01aa"+  
 "J\u0001\u0000\u0000\u0000\u01ab\u01ad\u0007\u0003\u0000\u0000\u01ac\u01ab"+  
 "\u0001\u0000\u0000\u0000\u01ad\u01ae\u0001\u0000\u0000\u0000\u01ae\u01ac"+  
 "\u0001\u0000\u0000\u0000\u01ae\u01af\u0001\u0000\u0000\u0000\u01af\u01b0"+  
 "\u0001\u0000\u0000\u0000\u01b0\u01b1\u0006%\u0000\u0000\u01b1L\u0001\u0000"+  
 "\u0000\u0000\u01b2\u01b3\u0007\u0004\u0000\u0000\u01b3\u01b4\u0001\u0000"+  
 "\u0000\u0000\u01b4\u01b5\u0006&\u0000\u0000\u01b5N\u0001\u0000\u0000\u0000"+  
 "\u01b6\u01b7\u0003G#\u0000\u01b7\u01b8\u0005.\u0000\u0000\u01b8\u01b9"+  
 "\u0003G#\u0000\u01b9\u01bc\u0001\u0000\u0000\u0000\u01ba\u01bc\u0003G"+  
 "#\u0000\u01bb\u01b6\u0001\u0000\u0000\u0000\u01bb\u01ba\u0001\u0000\u0000"+  
 "\u0000\u01bcP\u0001\u0000\u0000\u0000\u01bd\u01c3\u0003S)\u0000\u01be"+  
 "\u01bf\u0003U\*\u0000\u01bf\u01c0\u0003S)\u0000\u01c0\u01c2\u0001\u0000"+  
 "\u0000\u0000\u01c1\u01be\u0001\u0000\u0000\u0000\u01c2\u01c5\u0001\u0000"+  
 "\u0000\u0000\u01c3\u01c1\u0001\u0000\u0000\u0000\u01c3\u01c4\u0001\u0000"+  
 "\u0000\u0000\u01c4R\u0001\u0000\u0000\u0000\u01c5\u01c3\u0001\u0000\u0000"+  
 "\u0000\u01c6\u01c8\u0007\u0005\u0000\u0000\u01c7\u01c6\u0001\u0000\u0000"+  
 "\u0000\u01c8\u01c9\u0001\u0000\u0000\u0000\u01c9\u01c7\u0001\u0000\u0000"+  
 "\u0000\u01c9\u01ca\u0001\u0000\u0000\u0000\u01caT\u0001\u0000\u0000\u0000"+  
 "\u01cb\u01cc\u0007\u0006\u0000\u0000\u01cc\u01cd\u0001\u0000\u0000\u0000"+  
 "\u01cd\u01ce\u0006\*\u0000\u0000\u01ceV\u0001\u0000\u0000\u0000\b\u0000"+  
 "\u018a\u0194\u019b\u01ae\u01bb\u01c3\u01c9\u0001\u0006\u0000\u0000";  
 public static final ATN \_ATN =  
 new ATNDeserializer().deserialize(\_serializedATN.toCharArray());  
 static {  
 \_decisionToDFA = new DFA[\_ATN.getNumberOfDecisions()];  
 for (int i = 0; i < \_ATN.getNumberOfDecisions(); i++) {  
 \_decisionToDFA[i] = new DFA(\_ATN.getDecisionState(i), i);  
 }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameLexer.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameListener.txt

// Generated from C:/Users/mike\_/OneDrive/Documentos/sem4pi-22-23-61/base.core/src/main/java/eapli/base/ANTLR/ExamValidation\Exame.g4 by ANTLR 4.12.0  
package eapli.base.ANTLR.ExamValidation;  
import org.antlr.v4.runtime.tree.ParseTreeListener;  
  
/\*\*  
 \* This interface defines a complete listener for a parse tree produced by  
 \* {@link ExameParser}.  
 \*/  
public interface ExameListener extends ParseTreeListener {  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#prog}.  
 \* @param ctx the parse tree  
 \*/  
 void enterProg(ExameParser.ProgContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#prog}.  
 \* @param ctx the parse tree  
 \*/  
 void exitProg(ExameParser.ProgContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#exame}.  
 \* @param ctx the parse tree  
 \*/  
 void enterExame(ExameParser.ExameContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#exame}.  
 \* @param ctx the parse tree  
 \*/  
 void exitExame(ExameParser.ExameContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#corpo\_exame}.  
 \* @param ctx the parse tree  
 \*/  
 void enterCorpo\_exame(ExameParser.Corpo\_exameContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#corpo\_exame}.  
 \* @param ctx the parse tree  
 \*/  
 void exitCorpo\_exame(ExameParser.Corpo\_exameContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#seccao}.  
 \* @param ctx the parse tree  
 \*/  
 void enterSeccao(ExameParser.SeccaoContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#seccao}.  
 \* @param ctx the parse tree  
 \*/  
 void exitSeccao(ExameParser.SeccaoContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#pergunta}.  
 \* @param ctx the parse tree  
 \*/  
 void enterPergunta(ExameParser.PerguntaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#pergunta}.  
 \* @param ctx the parse tree  
 \*/  
 void exitPergunta(ExameParser.PerguntaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#corpo}.  
 \* @param ctx the parse tree  
 \*/  
 void enterCorpo(ExameParser.CorpoContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#corpo}.  
 \* @param ctx the parse tree  
 \*/  
 void exitCorpo(ExameParser.CorpoContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#verdadeiro\_falso}.  
 \* @param ctx the parse tree  
 \*/  
 void enterVerdadeiro\_falso(ExameParser.Verdadeiro\_falsoContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#verdadeiro\_falso}.  
 \* @param ctx the parse tree  
 \*/  
 void exitVerdadeiro\_falso(ExameParser.Verdadeiro\_falsoContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#palavras\_em\_falta}.  
 \* @param ctx the parse tree  
 \*/  
 void enterPalavras\_em\_falta(ExameParser.Palavras\_em\_faltaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#palavras\_em\_falta}.  
 \* @param ctx the parse tree  
 \*/  
 void exitPalavras\_em\_falta(ExameParser.Palavras\_em\_faltaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#resposta\_falta}.  
 \* @param ctx the parse tree  
 \*/  
 void enterResposta\_falta(ExameParser.Resposta\_faltaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#resposta\_falta}.  
 \* @param ctx the parse tree  
 \*/  
 void exitResposta\_falta(ExameParser.Resposta\_faltaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#numerica}.  
 \* @param ctx the parse tree  
 \*/  
 void enterNumerica(ExameParser.NumericaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#numerica}.  
 \* @param ctx the parse tree  
 \*/  
 void exitNumerica(ExameParser.NumericaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#resposta\_curta}.  
 \* @param ctx the parse tree  
 \*/  
 void enterResposta\_curta(ExameParser.Resposta\_curtaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#resposta\_curta}.  
 \* @param ctx the parse tree  
 \*/  
 void exitResposta\_curta(ExameParser.Resposta\_curtaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#escolha\_multipla}.  
 \* @param ctx the parse tree  
 \*/  
 void enterEscolha\_multipla(ExameParser.Escolha\_multiplaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#escolha\_multipla}.  
 \* @param ctx the parse tree  
 \*/  
 void exitEscolha\_multipla(ExameParser.Escolha\_multiplaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#opcao\_escolha\_multipla}.  
 \* @param ctx the parse tree  
 \*/  
 void enterOpcao\_escolha\_multipla(ExameParser.Opcao\_escolha\_multiplaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#opcao\_escolha\_multipla}.  
 \* @param ctx the parse tree  
 \*/  
 void exitOpcao\_escolha\_multipla(ExameParser.Opcao\_escolha\_multiplaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#correspondencia}.  
 \* @param ctx the parse tree  
 \*/  
 void enterCorrespondencia(ExameParser.CorrespondenciaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#correspondencia}.  
 \* @param ctx the parse tree  
 \*/  
 void exitCorrespondencia(ExameParser.CorrespondenciaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#token\_correspondencia}.  
 \* @param ctx the parse tree  
 \*/  
 void enterToken\_correspondencia(ExameParser.Token\_correspondenciaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#token\_correspondencia}.  
 \* @param ctx the parse tree  
 \*/  
 void exitToken\_correspondencia(ExameParser.Token\_correspondenciaContext ctx);  
 /\*\*  
 \* Enter a parse tree produced by {@link ExameParser#respostas\_correspondencia}.  
 \* @param ctx the parse tree  
 \*/  
 void enterRespostas\_correspondencia(ExameParser.Respostas\_correspondenciaContext ctx);  
 /\*\*  
 \* Exit a parse tree produced by {@link ExameParser#respostas\_correspondencia}.  
 \* @param ctx the parse tree  
 \*/  
 void exitRespostas\_correspondencia(ExameParser.Respostas\_correspondenciaContext ctx);  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameListener.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameParser.txt

// Generated from C:/Users/mike\_/OneDrive/Documentos/sem4pi-22-23-61/base.core/src/main/java/eapli/base/ANTLR/ExamValidation\Exame.g4 by ANTLR 4.12.0  
package eapli.base.ANTLR.ExamValidation;  
import org.antlr.v4.runtime.atn.\*;  
import org.antlr.v4.runtime.dfa.DFA;  
import org.antlr.v4.runtime.\*;  
import org.antlr.v4.runtime.misc.\*;  
import org.antlr.v4.runtime.tree.\*;  
import java.util.List;  
import java.util.Iterator;  
import java.util.ArrayList;  
  
  
public class ExameParser extends Parser {  
 static { RuntimeMetaData.checkVersion("4.12.0", RuntimeMetaData.VERSION); }  
  
 protected static final DFA[] \_decisionToDFA;  
 protected static final PredictionContextCache \_sharedContextCache =  
 new PredictionContextCache();  
 public static final int  
 DOISPONTOS=1, ABRECHAVETA=2, FECHACHAVETA=3, ABREPARANTESIS=4, FECHAPARANTESIS=5,   
 VIRGULA=6, TESTE=7, ID=8, TRACO=9, RESPOSTA=10, TITULO=11, DATAABERTURA=12,   
 DATAFECHO=13, OPCOES=14, OPCAO=15, ENUNCIADO=16, TABELA\_A=17, TABELA\_B=18,   
 COTACAO=19, DIFICULDADE=20, DESCRICAO\_TEXTUAL=21, PERGUNTAS=22, LIMITE\_PERGUNTAS=23,   
 SECCAO=24, PERGUNTA=25, TIPO\_PERGUNTA=26, NUMERICA=27, ESCOLHA\_MULTIPLA=28,   
 CORRESPONDENCIA=29, VERDADEIROOUFALSO=30, PALAVRAS\_EM\_FALTA=31, RESPOSTA\_CURTA=32,   
 VERDADEIRO=33, FALSO=34, TIPO=35, INT=36, DATA=37, WS=38, CARACTERE\_ESPECIAL=39,   
 NUMERO=40, FRASE=41, STRING=42, ESPACO=43;  
 public static final int  
 RULE\_prog = 0, RULE\_exame = 1, RULE\_corpo\_exame = 2, RULE\_seccao = 3,   
 RULE\_pergunta = 4, RULE\_corpo = 5, RULE\_verdadeiro\_falso = 6, RULE\_palavras\_em\_falta = 7,   
 RULE\_resposta\_falta = 8, RULE\_numerica = 9, RULE\_resposta\_curta = 10,   
 RULE\_escolha\_multipla = 11, RULE\_opcao\_escolha\_multipla = 12, RULE\_correspondencia = 13,   
 RULE\_token\_correspondencia = 14, RULE\_respostas\_correspondencia = 15;  
 private static String[] makeRuleNames() {  
 return new String[] {  
 "prog", "exame", "corpo\_exame", "seccao", "pergunta", "corpo", "verdadeiro\_falso",   
 "palavras\_em\_falta", "resposta\_falta", "numerica", "resposta\_curta",   
 "escolha\_multipla", "opcao\_escolha\_multipla", "correspondencia", "token\_correspondencia",   
 "respostas\_correspondencia"  
 };  
 }  
 public static final String[] ruleNames = makeRuleNames();  
  
 private static String[] makeLiteralNames() {  
 return new String[] {  
 null, "':'", "'{'", "'}'", "'['", "']'", "','", "'Teste'", "'ID'", "'-'",   
 "'Resposta'", "'Titulo'", "'DataAbertura'", "'DataFecho'", "'Opcoes'",   
 "'Opcao'", "'Enunciado'", "'TabelaA'", "'TabelaB'", "'Cotacao'", "'Dificuldade'",   
 "'DescricaoTextual'", "'Perguntas'", "'LimitePerguntas'", "'Seccao'",   
 "'Pergunta'", "'TipoPergunta'", "'Numerica'", "'EscolhaMultipla'", "'Correspondencia'",   
 "'VerdadeiroOuFalso'", "'PalavrasEmFalta'", "'RespostaCurta'", "'Verdadeiro'",   
 "'Falso'", "'Tipo'"  
 };  
 }  
 private static final String[] \_LITERAL\_NAMES = makeLiteralNames();  
 private static String[] makeSymbolicNames() {  
 return new String[] {  
 null, "DOISPONTOS", "ABRECHAVETA", "FECHACHAVETA", "ABREPARANTESIS",   
 "FECHAPARANTESIS", "VIRGULA", "TESTE", "ID", "TRACO", "RESPOSTA", "TITULO",   
 "DATAABERTURA", "DATAFECHO", "OPCOES", "OPCAO", "ENUNCIADO", "TABELA\_A",   
 "TABELA\_B", "COTACAO", "DIFICULDADE", "DESCRICAO\_TEXTUAL", "PERGUNTAS",   
 "LIMITE\_PERGUNTAS", "SECCAO", "PERGUNTA", "TIPO\_PERGUNTA", "NUMERICA",   
 "ESCOLHA\_MULTIPLA", "CORRESPONDENCIA", "VERDADEIROOUFALSO", "PALAVRAS\_EM\_FALTA",   
 "RESPOSTA\_CURTA", "VERDADEIRO", "FALSO", "TIPO", "INT", "DATA", "WS",   
 "CARACTERE\_ESPECIAL", "NUMERO", "FRASE", "STRING", "ESPACO"  
 };  
 }  
 private static final String[] \_SYMBOLIC\_NAMES = makeSymbolicNames();  
 public static final Vocabulary VOCABULARY = new VocabularyImpl(\_LITERAL\_NAMES, \_SYMBOLIC\_NAMES);  
  
 /\*\*  
 \* @deprecated Use {@link #VOCABULARY} instead.  
 \*/  
 @Deprecated  
 public static final String[] tokenNames;  
 static {  
 tokenNames = new String[\_SYMBOLIC\_NAMES.length];  
 for (int i = 0; i < tokenNames.length; i++) {  
 tokenNames[i] = VOCABULARY.getLiteralName(i);  
 if (tokenNames[i] == null) {  
 tokenNames[i] = VOCABULARY.getSymbolicName(i);  
 }  
  
 if (tokenNames[i] == null) {  
 tokenNames[i] = "<INVALID>";  
 }  
 }  
 }  
  
 @Override  
 @Deprecated  
 public String[] getTokenNames() {  
 return tokenNames;  
 }  
  
 @Override  
  
 public Vocabulary getVocabulary() {  
 return VOCABULARY;  
 }  
  
 @Override  
 public String getGrammarFileName() { return "Exame.g4"; }  
  
 @Override  
 public String[] getRuleNames() { return ruleNames; }  
  
 @Override  
 public String getSerializedATN() { return \_serializedATN; }  
  
 @Override  
 public ATN getATN() { return \_ATN; }  
  
 public ExameParser(TokenStream input) {  
 super(input);  
 \_interp = new ParserATNSimulator(this,\_ATN,\_decisionToDFA,\_sharedContextCache);  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class ProgContext extends ParserRuleContext {  
 public ExameContext exame() {  
 return getRuleContext(ExameContext.class,0);  
 }  
 public ProgContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_prog; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterProg(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitProg(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitProg(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final ProgContext prog() throws RecognitionException {  
 ProgContext \_localctx = new ProgContext(\_ctx, getState());  
 enterRule(\_localctx, 0, RULE\_prog);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(32);  
 exame();  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class ExameContext extends ParserRuleContext {  
 public TerminalNode TESTE() { return getToken(ExameParser.TESTE, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode ABRECHAVETA() { return getToken(ExameParser.ABRECHAVETA, 0); }  
 public Corpo\_exameContext corpo\_exame() {  
 return getRuleContext(Corpo\_exameContext.class,0);  
 }  
 public TerminalNode FECHACHAVETA() { return getToken(ExameParser.FECHACHAVETA, 0); }  
 public ExameContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_exame; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterExame(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitExame(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitExame(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final ExameContext exame() throws RecognitionException {  
 ExameContext \_localctx = new ExameContext(\_ctx, getState());  
 enterRule(\_localctx, 2, RULE\_exame);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(34);  
 match(TESTE);  
 setState(35);  
 match(DOISPONTOS);  
 setState(36);  
 match(ABRECHAVETA);  
 setState(37);  
 corpo\_exame();  
 setState(38);  
 match(FECHACHAVETA);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Corpo\_exameContext extends ParserRuleContext {  
 public Token id\_exame;  
 public Token titulo\_exame;  
 public Token data\_abertura;  
 public Token data\_fecho;  
 public TerminalNode ID() { return getToken(ExameParser.ID, 0); }  
 public List<TerminalNode> DOISPONTOS() { return getTokens(ExameParser.DOISPONTOS); }  
 public TerminalNode DOISPONTOS(int i) {  
 return getToken(ExameParser.DOISPONTOS, i);  
 }  
 public TerminalNode VIRGULA() { return getToken(ExameParser.VIRGULA, 0); }  
 public TerminalNode TITULO() { return getToken(ExameParser.TITULO, 0); }  
 public TerminalNode DATAABERTURA() { return getToken(ExameParser.DATAABERTURA, 0); }  
 public TerminalNode DATAFECHO() { return getToken(ExameParser.DATAFECHO, 0); }  
 public TerminalNode INT() { return getToken(ExameParser.INT, 0); }  
 public TerminalNode FRASE() { return getToken(ExameParser.FRASE, 0); }  
 public List<TerminalNode> DATA() { return getTokens(ExameParser.DATA); }  
 public TerminalNode DATA(int i) {  
 return getToken(ExameParser.DATA, i);  
 }  
 public List<SeccaoContext> seccao() {  
 return getRuleContexts(SeccaoContext.class);  
 }  
 public SeccaoContext seccao(int i) {  
 return getRuleContext(SeccaoContext.class,i);  
 }  
 public Corpo\_exameContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_corpo\_exame; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterCorpo\_exame(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitCorpo\_exame(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitCorpo\_exame(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Corpo\_exameContext corpo\_exame() throws RecognitionException {  
 Corpo\_exameContext \_localctx = new Corpo\_exameContext(\_ctx, getState());  
 enterRule(\_localctx, 4, RULE\_corpo\_exame);  
 int \_la;  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(40);  
 match(ID);  
 setState(41);  
 match(DOISPONTOS);  
 setState(42);  
 ((Corpo\_exameContext)\_localctx).id\_exame = match(INT);  
 setState(43);  
 match(VIRGULA);  
 setState(44);  
 match(TITULO);  
 setState(45);  
 match(DOISPONTOS);  
 setState(46);  
 ((Corpo\_exameContext)\_localctx).titulo\_exame = match(FRASE);  
 setState(48);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 do {  
 {  
 {  
 setState(47);  
 seccao();  
 }  
 }  
 setState(50);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 } while ( \_la==SECCAO );  
 setState(52);  
 match(DATAABERTURA);  
 setState(53);  
 match(DOISPONTOS);  
 setState(54);  
 ((Corpo\_exameContext)\_localctx).data\_abertura = match(DATA);  
 setState(55);  
 match(DATAFECHO);  
 setState(56);  
 match(DOISPONTOS);  
 setState(57);  
 ((Corpo\_exameContext)\_localctx).data\_fecho = match(DATA);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class SeccaoContext extends ParserRuleContext {  
 public Token id\_seccao;  
 public Token desc\_seccao;  
 public Token dificuldade\_seccao;  
 public Token limite;  
 public TerminalNode SECCAO() { return getToken(ExameParser.SECCAO, 0); }  
 public TerminalNode ID() { return getToken(ExameParser.ID, 0); }  
 public List<TerminalNode> DOISPONTOS() { return getTokens(ExameParser.DOISPONTOS); }  
 public TerminalNode DOISPONTOS(int i) {  
 return getToken(ExameParser.DOISPONTOS, i);  
 }  
 public TerminalNode VIRGULA() { return getToken(ExameParser.VIRGULA, 0); }  
 public TerminalNode ABRECHAVETA() { return getToken(ExameParser.ABRECHAVETA, 0); }  
 public TerminalNode DESCRICAO\_TEXTUAL() { return getToken(ExameParser.DESCRICAO\_TEXTUAL, 0); }  
 public TerminalNode DIFICULDADE() { return getToken(ExameParser.DIFICULDADE, 0); }  
 public TerminalNode LIMITE\_PERGUNTAS() { return getToken(ExameParser.LIMITE\_PERGUNTAS, 0); }  
 public TerminalNode PERGUNTAS() { return getToken(ExameParser.PERGUNTAS, 0); }  
 public TerminalNode FECHACHAVETA() { return getToken(ExameParser.FECHACHAVETA, 0); }  
 public List<TerminalNode> INT() { return getTokens(ExameParser.INT); }  
 public TerminalNode INT(int i) {  
 return getToken(ExameParser.INT, i);  
 }  
 public TerminalNode FRASE() { return getToken(ExameParser.FRASE, 0); }  
 public List<PerguntaContext> pergunta() {  
 return getRuleContexts(PerguntaContext.class);  
 }  
 public PerguntaContext pergunta(int i) {  
 return getRuleContext(PerguntaContext.class,i);  
 }  
 public SeccaoContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_seccao; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterSeccao(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitSeccao(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitSeccao(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final SeccaoContext seccao() throws RecognitionException {  
 SeccaoContext \_localctx = new SeccaoContext(\_ctx, getState());  
 enterRule(\_localctx, 6, RULE\_seccao);  
 int \_la;  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(59);  
 match(SECCAO);  
 setState(60);  
 match(ID);  
 setState(61);  
 match(DOISPONTOS);  
 setState(62);  
 ((SeccaoContext)\_localctx).id\_seccao = match(INT);  
 setState(63);  
 match(VIRGULA);  
 setState(64);  
 match(ABRECHAVETA);  
 setState(65);  
 match(DESCRICAO\_TEXTUAL);  
 setState(66);  
 match(DOISPONTOS);  
 setState(67);  
 ((SeccaoContext)\_localctx).desc\_seccao = match(FRASE);  
 setState(68);  
 match(DIFICULDADE);  
 setState(69);  
 match(DOISPONTOS);  
 setState(70);  
 ((SeccaoContext)\_localctx).dificuldade\_seccao = match(INT);  
 setState(71);  
 match(LIMITE\_PERGUNTAS);  
 setState(72);  
 match(DOISPONTOS);  
 setState(73);  
 ((SeccaoContext)\_localctx).limite = match(INT);  
 setState(74);  
 match(PERGUNTAS);  
 setState(75);  
 match(DOISPONTOS);  
 setState(77);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 do {  
 {  
 {  
 setState(76);  
 pergunta();  
 }  
 }  
 setState(79);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 } while ( \_la==PERGUNTA );  
 setState(81);  
 match(FECHACHAVETA);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class PerguntaContext extends ParserRuleContext {  
 public Token id\_pergunta;  
 public Token enunciado\_pergunta;  
 public Token tipo\_pergunta;  
 public Token cotacao\_pergunta;  
 public Token dificuldade\_pergunta;  
 public TerminalNode PERGUNTA() { return getToken(ExameParser.PERGUNTA, 0); }  
 public TerminalNode ID() { return getToken(ExameParser.ID, 0); }  
 public List<TerminalNode> DOISPONTOS() { return getTokens(ExameParser.DOISPONTOS); }  
 public TerminalNode DOISPONTOS(int i) {  
 return getToken(ExameParser.DOISPONTOS, i);  
 }  
 public TerminalNode VIRGULA() { return getToken(ExameParser.VIRGULA, 0); }  
 public TerminalNode ENUNCIADO() { return getToken(ExameParser.ENUNCIADO, 0); }  
 public CorpoContext corpo() {  
 return getRuleContext(CorpoContext.class,0);  
 }  
 public TerminalNode COTACAO() { return getToken(ExameParser.COTACAO, 0); }  
 public TerminalNode DIFICULDADE() { return getToken(ExameParser.DIFICULDADE, 0); }  
 public List<TerminalNode> INT() { return getTokens(ExameParser.INT); }  
 public TerminalNode INT(int i) {  
 return getToken(ExameParser.INT, i);  
 }  
 public TerminalNode FRASE() { return getToken(ExameParser.FRASE, 0); }  
 public TerminalNode TIPO\_PERGUNTA() { return getToken(ExameParser.TIPO\_PERGUNTA, 0); }  
 public PerguntaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_pergunta; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterPergunta(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitPergunta(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitPergunta(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final PerguntaContext pergunta() throws RecognitionException {  
 PerguntaContext \_localctx = new PerguntaContext(\_ctx, getState());  
 enterRule(\_localctx, 8, RULE\_pergunta);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(83);  
 match(PERGUNTA);  
 setState(84);  
 match(ID);  
 setState(85);  
 match(DOISPONTOS);  
 setState(86);  
 ((PerguntaContext)\_localctx).id\_pergunta = match(INT);  
 setState(87);  
 match(VIRGULA);  
 setState(88);  
 match(ENUNCIADO);  
 setState(89);  
 match(DOISPONTOS);  
 setState(90);  
 ((PerguntaContext)\_localctx).enunciado\_pergunta = match(FRASE);  
 setState(91);  
 ((PerguntaContext)\_localctx).tipo\_pergunta = match(TIPO\_PERGUNTA);  
 setState(92);  
 match(DOISPONTOS);  
 setState(93);  
 corpo();  
 setState(94);  
 match(COTACAO);  
 setState(95);  
 match(DOISPONTOS);  
 setState(96);  
 ((PerguntaContext)\_localctx).cotacao\_pergunta = match(INT);  
 setState(97);  
 match(DIFICULDADE);  
 setState(98);  
 match(DOISPONTOS);  
 setState(99);  
 ((PerguntaContext)\_localctx).dificuldade\_pergunta = match(INT);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class CorpoContext extends ParserRuleContext {  
 public CorrespondenciaContext correspondencia() {  
 return getRuleContext(CorrespondenciaContext.class,0);  
 }  
 public Escolha\_multiplaContext escolha\_multipla() {  
 return getRuleContext(Escolha\_multiplaContext.class,0);  
 }  
 public Resposta\_curtaContext resposta\_curta() {  
 return getRuleContext(Resposta\_curtaContext.class,0);  
 }  
 public NumericaContext numerica() {  
 return getRuleContext(NumericaContext.class,0);  
 }  
 public Palavras\_em\_faltaContext palavras\_em\_falta() {  
 return getRuleContext(Palavras\_em\_faltaContext.class,0);  
 }  
 public Verdadeiro\_falsoContext verdadeiro\_falso() {  
 return getRuleContext(Verdadeiro\_falsoContext.class,0);  
 }  
 public CorpoContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_corpo; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterCorpo(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitCorpo(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitCorpo(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final CorpoContext corpo() throws RecognitionException {  
 CorpoContext \_localctx = new CorpoContext(\_ctx, getState());  
 enterRule(\_localctx, 10, RULE\_corpo);  
 try {  
 setState(107);  
 \_errHandler.sync(this);  
 switch (\_input.LA(1)) {  
 case CORRESPONDENCIA:  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(101);  
 correspondencia();  
 }  
 break;  
 case ESCOLHA\_MULTIPLA:  
 enterOuterAlt(\_localctx, 2);  
 {  
 setState(102);  
 escolha\_multipla();  
 }  
 break;  
 case RESPOSTA\_CURTA:  
 enterOuterAlt(\_localctx, 3);  
 {  
 setState(103);  
 resposta\_curta();  
 }  
 break;  
 case NUMERICA:  
 enterOuterAlt(\_localctx, 4);  
 {  
 setState(104);  
 numerica();  
 }  
 break;  
 case PALAVRAS\_EM\_FALTA:  
 enterOuterAlt(\_localctx, 5);  
 {  
 setState(105);  
 palavras\_em\_falta();  
 }  
 break;  
 case VERDADEIROOUFALSO:  
 enterOuterAlt(\_localctx, 6);  
 {  
 setState(106);  
 verdadeiro\_falso();  
 }  
 break;  
 default:  
 throw new NoViableAltException(this);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Verdadeiro\_falsoContext extends ParserRuleContext {  
 public Token resposta\_verdOuFal;  
 public TerminalNode VERDADEIROOUFALSO() { return getToken(ExameParser.VERDADEIROOUFALSO, 0); }  
 public TerminalNode RESPOSTA() { return getToken(ExameParser.RESPOSTA, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode VERDADEIRO() { return getToken(ExameParser.VERDADEIRO, 0); }  
 public TerminalNode FALSO() { return getToken(ExameParser.FALSO, 0); }  
 public Verdadeiro\_falsoContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_verdadeiro\_falso; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterVerdadeiro\_falso(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitVerdadeiro\_falso(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitVerdadeiro\_falso(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Verdadeiro\_falsoContext verdadeiro\_falso() throws RecognitionException {  
 Verdadeiro\_falsoContext \_localctx = new Verdadeiro\_falsoContext(\_ctx, getState());  
 enterRule(\_localctx, 12, RULE\_verdadeiro\_falso);  
 int \_la;  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(109);  
 match(VERDADEIROOUFALSO);  
 setState(110);  
 match(RESPOSTA);  
 setState(111);  
 match(DOISPONTOS);  
 setState(112);  
 ((Verdadeiro\_falsoContext)\_localctx).resposta\_verdOuFal = \_input.LT(1);  
 \_la = \_input.LA(1);  
 if ( !(\_la==VERDADEIRO || \_la==FALSO) ) {  
 ((Verdadeiro\_falsoContext)\_localctx).resposta\_verdOuFal = (Token)\_errHandler.recoverInline(this);  
 }  
 else {  
 if ( \_input.LA(1)==Token.EOF ) matchedEOF = true;  
 \_errHandler.reportMatch(this);  
 consume();  
 }  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Palavras\_em\_faltaContext extends ParserRuleContext {  
 public TerminalNode PALAVRAS\_EM\_FALTA() { return getToken(ExameParser.PALAVRAS\_EM\_FALTA, 0); }  
 public TerminalNode RESPOSTA() { return getToken(ExameParser.RESPOSTA, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode ABRECHAVETA() { return getToken(ExameParser.ABRECHAVETA, 0); }  
 public TerminalNode FECHACHAVETA() { return getToken(ExameParser.FECHACHAVETA, 0); }  
 public List<Resposta\_faltaContext> resposta\_falta() {  
 return getRuleContexts(Resposta\_faltaContext.class);  
 }  
 public Resposta\_faltaContext resposta\_falta(int i) {  
 return getRuleContext(Resposta\_faltaContext.class,i);  
 }  
 public Palavras\_em\_faltaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_palavras\_em\_falta; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterPalavras\_em\_falta(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitPalavras\_em\_falta(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitPalavras\_em\_falta(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Palavras\_em\_faltaContext palavras\_em\_falta() throws RecognitionException {  
 Palavras\_em\_faltaContext \_localctx = new Palavras\_em\_faltaContext(\_ctx, getState());  
 enterRule(\_localctx, 14, RULE\_palavras\_em\_falta);  
 int \_la;  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(114);  
 match(PALAVRAS\_EM\_FALTA);  
 setState(115);  
 match(RESPOSTA);  
 setState(116);  
 match(DOISPONTOS);  
 setState(117);  
 match(ABRECHAVETA);  
 setState(119);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 do {  
 {  
 {  
 setState(118);  
 resposta\_falta();  
 }  
 }  
 setState(121);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 } while ( \_la==RESPOSTA );  
 setState(123);  
 match(FECHACHAVETA);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Resposta\_faltaContext extends ParserRuleContext {  
 public Token resp\_falta;  
 public TerminalNode RESPOSTA() { return getToken(ExameParser.RESPOSTA, 0); }  
 public TerminalNode INT() { return getToken(ExameParser.INT, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode FRASE() { return getToken(ExameParser.FRASE, 0); }  
 public Resposta\_faltaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_resposta\_falta; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterResposta\_falta(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitResposta\_falta(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitResposta\_falta(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Resposta\_faltaContext resposta\_falta() throws RecognitionException {  
 Resposta\_faltaContext \_localctx = new Resposta\_faltaContext(\_ctx, getState());  
 enterRule(\_localctx, 16, RULE\_resposta\_falta);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(125);  
 match(RESPOSTA);  
 setState(126);  
 match(INT);  
 setState(127);  
 match(DOISPONTOS);  
 setState(128);  
 ((Resposta\_faltaContext)\_localctx).resp\_falta = match(FRASE);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class NumericaContext extends ParserRuleContext {  
 public Token resp\_numerica;  
 public TerminalNode NUMERICA() { return getToken(ExameParser.NUMERICA, 0); }  
 public TerminalNode RESPOSTA() { return getToken(ExameParser.RESPOSTA, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode NUMERO() { return getToken(ExameParser.NUMERO, 0); }  
 public NumericaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_numerica; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterNumerica(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitNumerica(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitNumerica(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final NumericaContext numerica() throws RecognitionException {  
 NumericaContext \_localctx = new NumericaContext(\_ctx, getState());  
 enterRule(\_localctx, 18, RULE\_numerica);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(130);  
 match(NUMERICA);  
 setState(131);  
 match(RESPOSTA);  
 setState(132);  
 match(DOISPONTOS);  
 setState(133);  
 ((NumericaContext)\_localctx).resp\_numerica = match(NUMERO);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Resposta\_curtaContext extends ParserRuleContext {  
 public Token resp\_curta;  
 public TerminalNode RESPOSTA\_CURTA() { return getToken(ExameParser.RESPOSTA\_CURTA, 0); }  
 public TerminalNode RESPOSTA() { return getToken(ExameParser.RESPOSTA, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode FRASE() { return getToken(ExameParser.FRASE, 0); }  
 public Resposta\_curtaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_resposta\_curta; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterResposta\_curta(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitResposta\_curta(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitResposta\_curta(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Resposta\_curtaContext resposta\_curta() throws RecognitionException {  
 Resposta\_curtaContext \_localctx = new Resposta\_curtaContext(\_ctx, getState());  
 enterRule(\_localctx, 20, RULE\_resposta\_curta);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(135);  
 match(RESPOSTA\_CURTA);  
 setState(136);  
 match(RESPOSTA);  
 setState(137);  
 match(DOISPONTOS);  
 setState(138);  
 ((Resposta\_curtaContext)\_localctx).resp\_curta = match(FRASE);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Escolha\_multiplaContext extends ParserRuleContext {  
 public Token resposta\_escolha\_multipla;  
 public TerminalNode ESCOLHA\_MULTIPLA() { return getToken(ExameParser.ESCOLHA\_MULTIPLA, 0); }  
 public TerminalNode OPCOES() { return getToken(ExameParser.OPCOES, 0); }  
 public List<TerminalNode> DOISPONTOS() { return getTokens(ExameParser.DOISPONTOS); }  
 public TerminalNode DOISPONTOS(int i) {  
 return getToken(ExameParser.DOISPONTOS, i);  
 }  
 public TerminalNode ABREPARANTESIS() { return getToken(ExameParser.ABREPARANTESIS, 0); }  
 public TerminalNode FECHAPARANTESIS() { return getToken(ExameParser.FECHAPARANTESIS, 0); }  
 public TerminalNode RESPOSTA() { return getToken(ExameParser.RESPOSTA, 0); }  
 public TerminalNode INT() { return getToken(ExameParser.INT, 0); }  
 public List<Opcao\_escolha\_multiplaContext> opcao\_escolha\_multipla() {  
 return getRuleContexts(Opcao\_escolha\_multiplaContext.class);  
 }  
 public Opcao\_escolha\_multiplaContext opcao\_escolha\_multipla(int i) {  
 return getRuleContext(Opcao\_escolha\_multiplaContext.class,i);  
 }  
 public Escolha\_multiplaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_escolha\_multipla; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterEscolha\_multipla(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitEscolha\_multipla(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitEscolha\_multipla(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Escolha\_multiplaContext escolha\_multipla() throws RecognitionException {  
 Escolha\_multiplaContext \_localctx = new Escolha\_multiplaContext(\_ctx, getState());  
 enterRule(\_localctx, 22, RULE\_escolha\_multipla);  
 int \_la;  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(140);  
 match(ESCOLHA\_MULTIPLA);  
 setState(141);  
 match(OPCOES);  
 setState(142);  
 match(DOISPONTOS);  
 setState(143);  
 match(ABREPARANTESIS);  
 setState(145);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 do {  
 {  
 {  
 setState(144);  
 opcao\_escolha\_multipla();  
 }  
 }  
 setState(147);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 } while ( \_la==OPCAO );  
 setState(149);  
 match(FECHAPARANTESIS);  
 setState(150);  
 match(RESPOSTA);  
 setState(151);  
 match(DOISPONTOS);  
 setState(152);  
 ((Escolha\_multiplaContext)\_localctx).resposta\_escolha\_multipla = match(INT);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Opcao\_escolha\_multiplaContext extends ParserRuleContext {  
 public Token opcao\_escolha;  
 public TerminalNode OPCAO() { return getToken(ExameParser.OPCAO, 0); }  
 public TerminalNode INT() { return getToken(ExameParser.INT, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode FRASE() { return getToken(ExameParser.FRASE, 0); }  
 public Opcao\_escolha\_multiplaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_opcao\_escolha\_multipla; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterOpcao\_escolha\_multipla(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitOpcao\_escolha\_multipla(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitOpcao\_escolha\_multipla(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Opcao\_escolha\_multiplaContext opcao\_escolha\_multipla() throws RecognitionException {  
 Opcao\_escolha\_multiplaContext \_localctx = new Opcao\_escolha\_multiplaContext(\_ctx, getState());  
 enterRule(\_localctx, 24, RULE\_opcao\_escolha\_multipla);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(154);  
 match(OPCAO);  
 setState(155);  
 match(INT);  
 setState(156);  
 match(DOISPONTOS);  
 setState(157);  
 ((Opcao\_escolha\_multiplaContext)\_localctx).opcao\_escolha = match(FRASE);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class CorrespondenciaContext extends ParserRuleContext {  
 public TerminalNode CORRESPONDENCIA() { return getToken(ExameParser.CORRESPONDENCIA, 0); }  
 public TerminalNode TABELA\_A() { return getToken(ExameParser.TABELA\_A, 0); }  
 public List<TerminalNode> DOISPONTOS() { return getTokens(ExameParser.DOISPONTOS); }  
 public TerminalNode DOISPONTOS(int i) {  
 return getToken(ExameParser.DOISPONTOS, i);  
 }  
 public List<TerminalNode> ABREPARANTESIS() { return getTokens(ExameParser.ABREPARANTESIS); }  
 public TerminalNode ABREPARANTESIS(int i) {  
 return getToken(ExameParser.ABREPARANTESIS, i);  
 }  
 public List<TerminalNode> FECHAPARANTESIS() { return getTokens(ExameParser.FECHAPARANTESIS); }  
 public TerminalNode FECHAPARANTESIS(int i) {  
 return getToken(ExameParser.FECHAPARANTESIS, i);  
 }  
 public TerminalNode TABELA\_B() { return getToken(ExameParser.TABELA\_B, 0); }  
 public TerminalNode RESPOSTA() { return getToken(ExameParser.RESPOSTA, 0); }  
 public List<Token\_correspondenciaContext> token\_correspondencia() {  
 return getRuleContexts(Token\_correspondenciaContext.class);  
 }  
 public Token\_correspondenciaContext token\_correspondencia(int i) {  
 return getRuleContext(Token\_correspondenciaContext.class,i);  
 }  
 public List<Respostas\_correspondenciaContext> respostas\_correspondencia() {  
 return getRuleContexts(Respostas\_correspondenciaContext.class);  
 }  
 public Respostas\_correspondenciaContext respostas\_correspondencia(int i) {  
 return getRuleContext(Respostas\_correspondenciaContext.class,i);  
 }  
 public CorrespondenciaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_correspondencia; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterCorrespondencia(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitCorrespondencia(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitCorrespondencia(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final CorrespondenciaContext correspondencia() throws RecognitionException {  
 CorrespondenciaContext \_localctx = new CorrespondenciaContext(\_ctx, getState());  
 enterRule(\_localctx, 26, RULE\_correspondencia);  
 int \_la;  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(159);  
 match(CORRESPONDENCIA);  
 setState(160);  
 match(TABELA\_A);  
 setState(161);  
 match(DOISPONTOS);  
 setState(162);  
 match(ABREPARANTESIS);  
 setState(164);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 do {  
 {  
 {  
 setState(163);  
 token\_correspondencia();  
 }  
 }  
 setState(166);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 } while ( \_la==ENUNCIADO );  
 setState(168);  
 match(FECHAPARANTESIS);  
 setState(169);  
 match(TABELA\_B);  
 setState(170);  
 match(DOISPONTOS);  
 setState(171);  
 match(ABREPARANTESIS);  
 setState(173);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 do {  
 {  
 {  
 setState(172);  
 token\_correspondencia();  
 }  
 }  
 setState(175);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 } while ( \_la==ENUNCIADO );  
 setState(177);  
 match(FECHAPARANTESIS);  
 setState(178);  
 match(RESPOSTA);  
 setState(179);  
 match(DOISPONTOS);  
 setState(181);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 do {  
 {  
 {  
 setState(180);  
 respostas\_correspondencia();  
 }  
 }  
 setState(183);   
 \_errHandler.sync(this);  
 \_la = \_input.LA(1);  
 } while ( \_la==TABELA\_A );  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Token\_correspondenciaContext extends ParserRuleContext {  
 public Token resp\_correspondencia;  
 public TerminalNode ENUNCIADO() { return getToken(ExameParser.ENUNCIADO, 0); }  
 public TerminalNode INT() { return getToken(ExameParser.INT, 0); }  
 public TerminalNode DOISPONTOS() { return getToken(ExameParser.DOISPONTOS, 0); }  
 public TerminalNode FRASE() { return getToken(ExameParser.FRASE, 0); }  
 public Token\_correspondenciaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_token\_correspondencia; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterToken\_correspondencia(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitToken\_correspondencia(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitToken\_correspondencia(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Token\_correspondenciaContext token\_correspondencia() throws RecognitionException {  
 Token\_correspondenciaContext \_localctx = new Token\_correspondenciaContext(\_ctx, getState());  
 enterRule(\_localctx, 28, RULE\_token\_correspondencia);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(185);  
 match(ENUNCIADO);  
 setState(186);  
 match(INT);  
 setState(187);  
 match(DOISPONTOS);  
 setState(188);  
 ((Token\_correspondenciaContext)\_localctx).resp\_correspondencia = match(FRASE);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 @SuppressWarnings("CheckReturnValue")  
 public static class Respostas\_correspondenciaContext extends ParserRuleContext {  
 public TerminalNode TABELA\_A() { return getToken(ExameParser.TABELA\_A, 0); }  
 public List<TerminalNode> DOISPONTOS() { return getTokens(ExameParser.DOISPONTOS); }  
 public TerminalNode DOISPONTOS(int i) {  
 return getToken(ExameParser.DOISPONTOS, i);  
 }  
 public List<TerminalNode> INT() { return getTokens(ExameParser.INT); }  
 public TerminalNode INT(int i) {  
 return getToken(ExameParser.INT, i);  
 }  
 public TerminalNode TRACO() { return getToken(ExameParser.TRACO, 0); }  
 public TerminalNode TABELA\_B() { return getToken(ExameParser.TABELA\_B, 0); }  
 public Respostas\_correspondenciaContext(ParserRuleContext parent, int invokingState) {  
 super(parent, invokingState);  
 }  
 @Override public int getRuleIndex() { return RULE\_respostas\_correspondencia; }  
 @Override  
 public void enterRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).enterRespostas\_correspondencia(this);  
 }  
 @Override  
 public void exitRule(ParseTreeListener listener) {  
 if ( listener instanceof ExameListener ) ((ExameListener)listener).exitRespostas\_correspondencia(this);  
 }  
 @Override  
 public <T> T accept(ParseTreeVisitor<? extends T> visitor) {  
 if ( visitor instanceof ExameVisitor ) return ((ExameVisitor<? extends T>)visitor).visitRespostas\_correspondencia(this);  
 else return visitor.visitChildren(this);  
 }  
 }  
  
 public final Respostas\_correspondenciaContext respostas\_correspondencia() throws RecognitionException {  
 Respostas\_correspondenciaContext \_localctx = new Respostas\_correspondenciaContext(\_ctx, getState());  
 enterRule(\_localctx, 30, RULE\_respostas\_correspondencia);  
 try {  
 enterOuterAlt(\_localctx, 1);  
 {  
 setState(190);  
 match(TABELA\_A);  
 setState(191);  
 match(DOISPONTOS);  
 setState(192);  
 match(INT);  
 setState(193);  
 match(TRACO);  
 setState(194);  
 match(TABELA\_B);  
 setState(195);  
 match(DOISPONTOS);  
 setState(196);  
 match(INT);  
 }  
 }  
 catch (RecognitionException re) {  
 \_localctx.exception = re;  
 \_errHandler.reportError(this, re);  
 \_errHandler.recover(this, re);  
 }  
 finally {  
 exitRule();  
 }  
 return \_localctx;  
 }  
  
 public static final String \_serializedATN =  
 "\u0004\u0001+\u00c7\u0002\u0000\u0007\u0000\u0002\u0001\u0007\u0001\u0002"+  
 "\u0002\u0007\u0002\u0002\u0003\u0007\u0003\u0002\u0004\u0007\u0004\u0002"+  
 "\u0005\u0007\u0005\u0002\u0006\u0007\u0006\u0002\u0007\u0007\u0007\u0002"+  
 "\b\u0007\b\u0002\t\u0007\t\u0002\n\u0007\n\u0002\u000b\u0007\u000b\u0002"+  
 "\f\u0007\f\u0002\r\u0007\r\u0002\u000e\u0007\u000e\u0002\u000f\u0007\u000f"+  
 "\u0001\u0000\u0001\u0000\u0001\u0001\u0001\u0001\u0001\u0001\u0001\u0001"+  
 "\u0001\u0001\u0001\u0001\u0001\u0002\u0001\u0002\u0001\u0002\u0001\u0002"+  
 "\u0001\u0002\u0001\u0002\u0001\u0002\u0001\u0002\u0004\u00021\b\u0002"+  
 "\u000b\u0002\f\u00022\u0001\u0002\u0001\u0002\u0001\u0002\u0001\u0002"+  
 "\u0001\u0002\u0001\u0002\u0001\u0002\u0001\u0003\u0001\u0003\u0001\u0003"+  
 "\u0001\u0003\u0001\u0003\u0001\u0003\u0001\u0003\u0001\u0003\u0001\u0003"+  
 "\u0001\u0003\u0001\u0003\u0001\u0003\u0001\u0003\u0001\u0003\u0001\u0003"+  
 "\u0001\u0003\u0001\u0003\u0001\u0003\u0004\u0003N\b\u0003\u000b\u0003"+  
 "\f\u0003O\u0001\u0003\u0001\u0003\u0001\u0004\u0001\u0004\u0001\u0004"+  
 "\u0001\u0004\u0001\u0004\u0001\u0004\u0001\u0004\u0001\u0004\u0001\u0004"+  
 "\u0001\u0004\u0001\u0004\u0001\u0004\u0001\u0004\u0001\u0004\u0001\u0004"+  
 "\u0001\u0004\u0001\u0004\u0001\u0004\u0001\u0005\u0001\u0005\u0001\u0005"+  
 "\u0001\u0005\u0001\u0005\u0001\u0005\u0003\u0005l\b\u0005\u0001\u0006"+  
 "\u0001\u0006\u0001\u0006\u0001\u0006\u0001\u0006\u0001\u0007\u0001\u0007"+  
 "\u0001\u0007\u0001\u0007\u0001\u0007\u0004\u0007x\b\u0007\u000b\u0007"+  
 "\f\u0007y\u0001\u0007\u0001\u0007\u0001\b\u0001\b\u0001\b\u0001\b\u0001"+  
 "\b\u0001\t\u0001\t\u0001\t\u0001\t\u0001\t\u0001\n\u0001\n\u0001\n\u0001"+  
 "\n\u0001\n\u0001\u000b\u0001\u000b\u0001\u000b\u0001\u000b\u0001\u000b"+  
 "\u0004\u000b\u0092\b\u000b\u000b\u000b\f\u000b\u0093\u0001\u000b\u0001"+  
 "\u000b\u0001\u000b\u0001\u000b\u0001\u000b\u0001\f\u0001\f\u0001\f\u0001"+  
 "\f\u0001\f\u0001\r\u0001\r\u0001\r\u0001\r\u0001\r\u0004\r\u00a5\b\r\u000b"+  
 "\r\f\r\u00a6\u0001\r\u0001\r\u0001\r\u0001\r\u0001\r\u0004\r\u00ae\b\r"+  
 "\u000b\r\f\r\u00af\u0001\r\u0001\r\u0001\r\u0001\r\u0004\r\u00b6\b\r\u000b"+  
 "\r\f\r\u00b7\u0001\u000e\u0001\u000e\u0001\u000e\u0001\u000e\u0001\u000e"+  
 "\u0001\u000f\u0001\u000f\u0001\u000f\u0001\u000f\u0001\u000f\u0001\u000f"+  
 "\u0001\u000f\u0001\u000f\u0001\u000f\u0000\u0000\u0010\u0000\u0002\u0004"+  
 "\u0006\b\n\f\u000e\u0010\u0012\u0014\u0016\u0018\u001a\u001c\u001e\u0000"+  
 "\u0001\u0001\u0000!\"\u00c2\u0000 \u0001\u0000\u0000\u0000\u0002\"\u0001"+  
 "\u0000\u0000\u0000\u0004(\u0001\u0000\u0000\u0000\u0006;\u0001\u0000\u0000"+  
 "\u0000\bS\u0001\u0000\u0000\u0000\nk\u0001\u0000\u0000\u0000\fm\u0001"+  
 "\u0000\u0000\u0000\u000er\u0001\u0000\u0000\u0000\u0010}\u0001\u0000\u0000"+  
 "\u0000\u0012\u0082\u0001\u0000\u0000\u0000\u0014\u0087\u0001\u0000\u0000"+  
 "\u0000\u0016\u008c\u0001\u0000\u0000\u0000\u0018\u009a\u0001\u0000\u0000"+  
 "\u0000\u001a\u009f\u0001\u0000\u0000\u0000\u001c\u00b9\u0001\u0000\u0000"+  
 "\u0000\u001e\u00be\u0001\u0000\u0000\u0000 !\u0003\u0002\u0001\u0000!"+  
 "\u0001\u0001\u0000\u0000\u0000\"#\u0005\u0007\u0000\u0000#$\u0005\u0001"+  
 "\u0000\u0000$%\u0005\u0002\u0000\u0000%&\u0003\u0004\u0002\u0000&\'\u0005"+  
 "\u0003\u0000\u0000\'\u0003\u0001\u0000\u0000\u0000()\u0005\b\u0000\u0000"+  
 ")\*\u0005\u0001\u0000\u0000\*+\u0005$\u0000\u0000+,\u0005\u0006\u0000\u0000"+  
 ",-\u0005\u000b\u0000\u0000-.\u0005\u0001\u0000\u0000.0\u0005)\u0000\u0000"+  
 "/1\u0003\u0006\u0003\u00000/\u0001\u0000\u0000\u000012\u0001\u0000\u0000"+  
 "\u000020\u0001\u0000\u0000\u000023\u0001\u0000\u0000\u000034\u0001\u0000"+  
 "\u0000\u000045\u0005\f\u0000\u000056\u0005\u0001\u0000\u000067\u0005%"+  
 "\u0000\u000078\u0005\r\u0000\u000089\u0005\u0001\u0000\u00009:\u0005%"+  
 "\u0000\u0000:\u0005\u0001\u0000\u0000\u0000;<\u0005\u0018\u0000\u0000"+  
 "<=\u0005\b\u0000\u0000=>\u0005\u0001\u0000\u0000>?\u0005$\u0000\u0000"+  
 "?@\u0005\u0006\u0000\u0000@A\u0005\u0002\u0000\u0000AB\u0005\u0015\u0000"+  
 "\u0000BC\u0005\u0001\u0000\u0000CD\u0005)\u0000\u0000DE\u0005\u0014\u0000"+  
 "\u0000EF\u0005\u0001\u0000\u0000FG\u0005$\u0000\u0000GH\u0005\u0017\u0000"+  
 "\u0000HI\u0005\u0001\u0000\u0000IJ\u0005$\u0000\u0000JK\u0005\u0016\u0000"+  
 "\u0000KM\u0005\u0001\u0000\u0000LN\u0003\b\u0004\u0000ML\u0001\u0000\u0000"+  
 "\u0000NO\u0001\u0000\u0000\u0000OM\u0001\u0000\u0000\u0000OP\u0001\u0000"+  
 "\u0000\u0000PQ\u0001\u0000\u0000\u0000QR\u0005\u0003\u0000\u0000R\u0007"+  
 "\u0001\u0000\u0000\u0000ST\u0005\u0019\u0000\u0000TU\u0005\b\u0000\u0000"+  
 "UV\u0005\u0001\u0000\u0000VW\u0005$\u0000\u0000WX\u0005\u0006\u0000\u0000"+  
 "XY\u0005\u0010\u0000\u0000YZ\u0005\u0001\u0000\u0000Z[\u0005)\u0000\u0000"+  
 "[\\\u0005\u001a\u0000\u0000\\]\u0005\u0001\u0000\u0000]^\u0003\n\u0005"+  
 "\u0000^\_\u0005\u0013\u0000\u0000\_`\u0005\u0001\u0000\u0000`a\u0005$\u0000"+  
 "\u0000ab\u0005\u0014\u0000\u0000bc\u0005\u0001\u0000\u0000cd\u0005$\u0000"+  
 "\u0000d\t\u0001\u0000\u0000\u0000el\u0003\u001a\r\u0000fl\u0003\u0016"+  
 "\u000b\u0000gl\u0003\u0014\n\u0000hl\u0003\u0012\t\u0000il\u0003\u000e"+  
 "\u0007\u0000jl\u0003\f\u0006\u0000ke\u0001\u0000\u0000\u0000kf\u0001\u0000"+  
 "\u0000\u0000kg\u0001\u0000\u0000\u0000kh\u0001\u0000\u0000\u0000ki\u0001"+  
 "\u0000\u0000\u0000kj\u0001\u0000\u0000\u0000l\u000b\u0001\u0000\u0000"+  
 "\u0000mn\u0005\u001e\u0000\u0000no\u0005\n\u0000\u0000op\u0005\u0001\u0000"+  
 "\u0000pq\u0007\u0000\u0000\u0000q\r\u0001\u0000\u0000\u0000rs\u0005\u001f"+  
 "\u0000\u0000st\u0005\n\u0000\u0000tu\u0005\u0001\u0000\u0000uw\u0005\u0002"+  
 "\u0000\u0000vx\u0003\u0010\b\u0000wv\u0001\u0000\u0000\u0000xy\u0001\u0000"+  
 "\u0000\u0000yw\u0001\u0000\u0000\u0000yz\u0001\u0000\u0000\u0000z{\u0001"+  
 "\u0000\u0000\u0000{|\u0005\u0003\u0000\u0000|\u000f\u0001\u0000\u0000"+  
 "\u0000}~\u0005\n\u0000\u0000~\u007f\u0005$\u0000\u0000\u007f\u0080\u0005"+  
 "\u0001\u0000\u0000\u0080\u0081\u0005)\u0000\u0000\u0081\u0011\u0001\u0000"+  
 "\u0000\u0000\u0082\u0083\u0005\u001b\u0000\u0000\u0083\u0084\u0005\n\u0000"+  
 "\u0000\u0084\u0085\u0005\u0001\u0000\u0000\u0085\u0086\u0005(\u0000\u0000"+  
 "\u0086\u0013\u0001\u0000\u0000\u0000\u0087\u0088\u0005 \u0000\u0000\u0088"+  
 "\u0089\u0005\n\u0000\u0000\u0089\u008a\u0005\u0001\u0000\u0000\u008a\u008b"+  
 "\u0005)\u0000\u0000\u008b\u0015\u0001\u0000\u0000\u0000\u008c\u008d\u0005"+  
 "\u001c\u0000\u0000\u008d\u008e\u0005\u000e\u0000\u0000\u008e\u008f\u0005"+  
 "\u0001\u0000\u0000\u008f\u0091\u0005\u0004\u0000\u0000\u0090\u0092\u0003"+  
 "\u0018\f\u0000\u0091\u0090\u0001\u0000\u0000\u0000\u0092\u0093\u0001\u0000"+  
 "\u0000\u0000\u0093\u0091\u0001\u0000\u0000\u0000\u0093\u0094\u0001\u0000"+  
 "\u0000\u0000\u0094\u0095\u0001\u0000\u0000\u0000\u0095\u0096\u0005\u0005"+  
 "\u0000\u0000\u0096\u0097\u0005\n\u0000\u0000\u0097\u0098\u0005\u0001\u0000"+  
 "\u0000\u0098\u0099\u0005$\u0000\u0000\u0099\u0017\u0001\u0000\u0000\u0000"+  
 "\u009a\u009b\u0005\u000f\u0000\u0000\u009b\u009c\u0005$\u0000\u0000\u009c"+  
 "\u009d\u0005\u0001\u0000\u0000\u009d\u009e\u0005)\u0000\u0000\u009e\u0019"+  
 "\u0001\u0000\u0000\u0000\u009f\u00a0\u0005\u001d\u0000\u0000\u00a0\u00a1"+  
 "\u0005\u0011\u0000\u0000\u00a1\u00a2\u0005\u0001\u0000\u0000\u00a2\u00a4"+  
 "\u0005\u0004\u0000\u0000\u00a3\u00a5\u0003\u001c\u000e\u0000\u00a4\u00a3"+  
 "\u0001\u0000\u0000\u0000\u00a5\u00a6\u0001\u0000\u0000\u0000\u00a6\u00a4"+  
 "\u0001\u0000\u0000\u0000\u00a6\u00a7\u0001\u0000\u0000\u0000\u00a7\u00a8"+  
 "\u0001\u0000\u0000\u0000\u00a8\u00a9\u0005\u0005\u0000\u0000\u00a9\u00aa"+  
 "\u0005\u0012\u0000\u0000\u00aa\u00ab\u0005\u0001\u0000\u0000\u00ab\u00ad"+  
 "\u0005\u0004\u0000\u0000\u00ac\u00ae\u0003\u001c\u000e\u0000\u00ad\u00ac"+  
 "\u0001\u0000\u0000\u0000\u00ae\u00af\u0001\u0000\u0000\u0000\u00af\u00ad"+  
 "\u0001\u0000\u0000\u0000\u00af\u00b0\u0001\u0000\u0000\u0000\u00b0\u00b1"+  
 "\u0001\u0000\u0000\u0000\u00b1\u00b2\u0005\u0005\u0000\u0000\u00b2\u00b3"+  
 "\u0005\n\u0000\u0000\u00b3\u00b5\u0005\u0001\u0000\u0000\u00b4\u00b6\u0003"+  
 "\u001e\u000f\u0000\u00b5\u00b4\u0001\u0000\u0000\u0000\u00b6\u00b7\u0001"+  
 "\u0000\u0000\u0000\u00b7\u00b5\u0001\u0000\u0000\u0000\u00b7\u00b8\u0001"+  
 "\u0000\u0000\u0000\u00b8\u001b\u0001\u0000\u0000\u0000\u00b9\u00ba\u0005"+  
 "\u0010\u0000\u0000\u00ba\u00bb\u0005$\u0000\u0000\u00bb\u00bc\u0005\u0001"+  
 "\u0000\u0000\u00bc\u00bd\u0005)\u0000\u0000\u00bd\u001d\u0001\u0000\u0000"+  
 "\u0000\u00be\u00bf\u0005\u0011\u0000\u0000\u00bf\u00c0\u0005\u0001\u0000"+  
 "\u0000\u00c0\u00c1\u0005$\u0000\u0000\u00c1\u00c2\u0005\t\u0000\u0000"+  
 "\u00c2\u00c3\u0005\u0012\u0000\u0000\u00c3\u00c4\u0005\u0001\u0000\u0000"+  
 "\u00c4\u00c5\u0005$\u0000\u0000\u00c5\u001f\u0001\u0000\u0000\u0000\b"+  
 "2Oky\u0093\u00a6\u00af\u00b7";  
 public static final ATN \_ATN =  
 new ATNDeserializer().deserialize(\_serializedATN.toCharArray());  
 static {  
 \_decisionToDFA = new DFA[\_ATN.getNumberOfDecisions()];  
 for (int i = 0; i < \_ATN.getNumberOfDecisions(); i++) {  
 \_decisionToDFA[i] = new DFA(\_ATN.getDecisionState(i), i);  
 }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameParser.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameVisitor.txt

// Generated from C:/Users/mike\_/OneDrive/Documentos/sem4pi-22-23-61/base.core/src/main/java/eapli/base/ANTLR/ExamValidation\Exame.g4 by ANTLR 4.12.0  
package eapli.base.ANTLR.ExamValidation;  
import org.antlr.v4.runtime.tree.ParseTreeVisitor;  
  
/\*\*  
 \* This interface defines a complete generic visitor for a parse tree produced  
 \* by {@link ExameParser}.  
 \*  
 \* @param <T> The return type of the visit operation. Use {@link Void} for  
 \* operations with no return type.  
 \*/  
public interface ExameVisitor<T> extends ParseTreeVisitor<T> {  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#prog}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitProg(ExameParser.ProgContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#exame}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitExame(ExameParser.ExameContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#corpo\_exame}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitCorpo\_exame(ExameParser.Corpo\_exameContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#seccao}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitSeccao(ExameParser.SeccaoContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#pergunta}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitPergunta(ExameParser.PerguntaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#corpo}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitCorpo(ExameParser.CorpoContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#verdadeiro\_falso}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitVerdadeiro\_falso(ExameParser.Verdadeiro\_falsoContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#palavras\_em\_falta}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitPalavras\_em\_falta(ExameParser.Palavras\_em\_faltaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#resposta\_falta}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitResposta\_falta(ExameParser.Resposta\_faltaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#numerica}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitNumerica(ExameParser.NumericaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#resposta\_curta}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitResposta\_curta(ExameParser.Resposta\_curtaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#escolha\_multipla}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitEscolha\_multipla(ExameParser.Escolha\_multiplaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#opcao\_escolha\_multipla}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitOpcao\_escolha\_multipla(ExameParser.Opcao\_escolha\_multiplaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#correspondencia}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitCorrespondencia(ExameParser.CorrespondenciaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#token\_correspondencia}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitToken\_correspondencia(ExameParser.Token\_correspondenciaContext ctx);  
 /\*\*  
 \* Visit a parse tree produced by {@link ExameParser#respostas\_correspondencia}.  
 \* @param ctx the parse tree  
 \* @return the visitor result  
 \*/  
 T visitRespostas\_correspondencia(ExameParser.Respostas\_correspondenciaContext ctx);  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\ExamValidation\ExameVisitor.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\FormativeExamValidation\ExameFormativo.g4

grammar ExameFormativo;  
  
prog: exam;  
  
exam: TESTE DOISPONTOS ABRECHAVETA formative\_exam\_body FECHACHAVETA;  
  
formative\_exam\_body: ID DOISPONTOS id\_exam=INT VIRGULA TITULO DOISPONTOS title\_exam=FRASE section+;  
  
section: SECÇÃO ID DOISPONTOS id\_section=INT VIRGULA ABRECHAVETA DESCRICAO\_TEXTUAL DOISPONTOS desc\_section=FRASE DIFICULDADE DOISPONTOS difficulty\_section=INT LIMITE\_PERGUNTAS DOISPONTOS limit=INT TIPO\_PERGUNTA DOISPONTOS question\_type=QUESTION\_TYPE FECHACHAVETA;  
  
QUESTION\_TYPE: 'Numérica' | 'Escolha Múltipla' | 'Correspondência' | 'Verdadeiro ou Falso' | 'Palavras em Falta' | 'Resposta Curta';  
  
DOISPONTOS : ':';  
ABRECHAVETA : '{';  
FECHACHAVETA : '}';  
VIRGULA : ',';  
TESTE : 'Teste';  
ID : 'ID';  
TITULO : 'Título';  
SECÇÕES : 'Secções';  
ENUNCIADO : 'Enunciado';  
COTACAO : 'Cotação';  
DIFICULDADE : 'Dificuldade';  
DESCRICAO\_TEXTUAL : 'Descrição Textual';  
LIMITE\_PERGUNTAS : 'Limite de Perguntas';  
SECÇÃO : 'Secção';  
TIPO\_PERGUNTA : 'Tipo de Pergunta';  
INT : [0-9]+;  
WS: [\r\n ]+ -> skip;  
CARACTERE\_ESPECIAL: [\_()+/\-'#"] -> skip;  
STRING : [a-zA-Z]+[0-9]+ | '\_' | '\'' ;  
ESPACO: [ \t] -> skip;  
FRASE: CARACTERE\_ESPECIAL? (PALAVRA|INT) PALAVRA (PALAVRA|INT|ESPACO|CARACTERE\_ESPECIAL|VIRGULA)\* FIM CARACTERE\_ESPECIAL? ;  
PALAVRA: [A-Za-z]+;  
FIM: [.!?;];

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\FormativeExamValidation\ExameFormativo.g4

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\FormativeExamValidation\ExameFormativoFile.txt

Teste: {  
 ID: 1,  
 Título: "Exame Formativo de Literatura Inglesa",  
 Secção ID: 1, {  
 Descrição Textual: "Analise de Poesia",  
 Dificuldade: 3,  
 Limite de Perguntas: 5,  
 Tipo de Pergunta: "Escolha Múltipla"  
 },  
 Secção ID: 2, {  
 Descrição Textual: "Analise de Prosa",  
 Dificuldade: 2,  
 Limite de Perguntas: 3,  
 Tipo de Pergunta: "Resposta Curta"  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\FormativeExamValidation\ExameFormativoFile.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\QuestionValidation\Question.g4

grammar Question;  
  
pprog: (add\_question | update\_question)+;  
  
add\_question: ADD\_QUESTION COLON OPENBRACE question\_body CLOSEBRACE SEMI;  
  
update\_question: UPDATE\_QUESTION COLON OPENBRACE question\_id COMMA question\_body CLOSEBRACE SEMI;  
  
question\_id: ID COLON INT;  
  
question\_body: (title | question\_type | options | answer ) (COMMA (title | question\_type | options | answer))\* ;  
  
title: TITLE COLON PHRASE;  
question\_type: QUESTION\_TYPE COLON PHRASE;  
options: OPTIONS COLON OPENBRACKETS option+ CLOSEBRACKETS;  
answer: ANSWER COLON INT;  
  
option: OPTION INT COLON PHRASE;  
  
ADD\_QUESTION: 'Add Question';  
UPDATE\_QUESTION: 'Update Question';  
ID: 'ID';  
TITLE: 'Title';  
OPTIONS: 'Options';  
OPTION: 'Option';  
ANSWER: 'Answer';  
QUESTION\_TYPE: 'Question Type';  
INT: [0-9]+;  
PHRASE: '"' ~["]\* '"';  
COLON: ':';  
COMMA: ',';  
OPENBRACE: '{';  
CLOSEBRACE: '}';  
OPENBRACKETS: '[';  
CLOSEBRACKETS: ']';  
SEMI: ';';  
WS: [\r\n\t ]+ -> skip;

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\QuestionValidation\Question.g4

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\QuestionValidation\QuestionFile.txt

Add Question:  
 {  
 ID: 1,  
 Title: "Multiple Choice Question",  
 Question Type: "Multiple Choice",  
 Options: [  
 Option 1: "Option A",  
 Option 2: "Option B",  
 Option 3: "Option C"  
 ],  
 Answer: 2  
 }  
  
 Update Question:  
 {  
 ID: 1,  
 Title: "Multiple Choice Question (Updated)",  
 Question Type: "Multiple Choice",  
 Options: [  
 Option 1: "Option A",  
 Option 2: "Option B",  
 Option 3: "Option C",  
 Option 4: "Option D"  
 ],  
 Answer: 3  
 }

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ANTLR\QuestionValidation\QuestionFile.txt

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\AppSettings.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base;  
  
import java.io.FileNotFoundException;  
import java.io.IOException;  
import java.io.InputStream;  
import java.util.HashMap;  
import java.util.Map;  
import java.util.Properties;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
/\*\*  
 \* the application settings.  
 \*  
 \* @author Paulo Gandra Sousa  
 \*/  
public class AppSettings {  
 private static final Logger LOGGER = LoggerFactory.getLogger(AppSettings.class);  
  
 private static final String PROPERTIES\_RESOURCE = "application.properties";  
 private static final String REPOSITORY\_FACTORY\_KEY = "persistence.repositoryFactory";  
 private static final String UI\_MENU\_LAYOUT\_KEY = "ui.menu.layout";  
 private static final String PERSISTENCE\_UNIT\_KEY = "persistence.persistenceUnit";  
 private static final String SCHEMA\_GENERATION\_KEY = "javax.persistence.schema-generation.database.action";  
 private static final String HIGH\_CALORIES\_DISH\_LIMIT = "HighCaloriesDishLimit";  
  
 private final Properties applicationProperties = new Properties();  
  
 public AppSettings() {  
 loadProperties();  
 }  
  
 private void loadProperties() {  
 try (InputStream propertiesStream = this.getClass().getClassLoader()  
 .getResourceAsStream(PROPERTIES\_RESOURCE)) {  
 if (propertiesStream != null) {  
 this.applicationProperties.load(propertiesStream);  
 } else {  
 throw new FileNotFoundException(  
 "property file '" + PROPERTIES\_RESOURCE + "' not found in the classpath");  
 }  
 } catch (final IOException exio) {  
 setDefaultProperties();  
  
 LOGGER.warn("Loading default properties", exio);  
 }  
 }  
  
 private void setDefaultProperties() {  
 this.applicationProperties.setProperty(REPOSITORY\_FACTORY\_KEY,  
 "eapli.base.persistence.jpa.JpaRepositoryFactory");  
 this.applicationProperties.setProperty(UI\_MENU\_LAYOUT\_KEY, "horizontal");  
 this.applicationProperties.setProperty(PERSISTENCE\_UNIT\_KEY, "eapli"  
 + ".base");  
 this.applicationProperties.setProperty(HIGH\_CALORIES\_DISH\_LIMIT, "300");  
 }  
  
 public Boolean isMenuLayoutHorizontal() {  
 return "horizontal"  
 .equalsIgnoreCase(this.applicationProperties.getProperty(UI\_MENU\_LAYOUT\_KEY));  
 }  
  
 public String getPersistenceUnitName() {  
 return this.applicationProperties.getProperty(PERSISTENCE\_UNIT\_KEY);  
 }  
  
 public String getRepositoryFactory() {  
 return this.applicationProperties.getProperty(REPOSITORY\_FACTORY\_KEY);  
 }  
  
 public Integer getHighCaloriesDishLimit() {  
 return Integer.valueOf(this.applicationProperties.getProperty(HIGH\_CALORIES\_DISH\_LIMIT));  
 }  
  
 @SuppressWarnings({ "rawtypes", "unchecked" })  
 public Map getExtendedPersistenceProperties() {  
 final Map ret = new HashMap();  
 ret.put(SCHEMA\_GENERATION\_KEY,  
 this.applicationProperties.getProperty(SCHEMA\_GENERATION\_KEY));  
 return ret;  
 }  
  
 public String getProperty(final String prop) {  
 return this.applicationProperties.getProperty(prop);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\AppSettings.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Application.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base;  
  
/\*\*  
 \* A "global" static class with the application registry of well known objects  
 \*  
 \* @author Paulo Gandra Sousa  
 \*  
 \*/  
public class Application {  
  
 public static final String VERSION = "1.4.0";  
 public static final String COPYRIGHT = "(C) 2016 - 2021, ISEP's Professors of EAPLI";  
  
 private static final AppSettings SETTINGS = new AppSettings();  
  
 public static AppSettings settings() {  
 return SETTINGS;  
 }  
  
 private Application() {  
 // private visibility to ensure singleton & utility  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Application.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\aplication\ClassController.java

package eapli.base.Classe.aplication;  
  
import eapli.base.Classe.domain.\*;  
import eapli.base.Classe.domain.Classe;  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
  
import java.time.LocalTime;  
import java.util.List;  
import java.util.Optional;  
  
public class ClassController {  
  
  
 private ClassRepository repository;  
 private final ScheduleClassService scs;  
  
 public ClassController(final AuthorizationService authz, final ClassRepository repository, final TeacherRepository teacherRepository) {  
 scs = new ScheduleClassService(authz, repository, teacherRepository);  
  
 }  
  
 public ClasseDTO scheduleClass(Classe\_Title title, Classe\_Start\_Time startTime, Classe\_Finish\_Time finishTime, Classe\_Start\_Date start\_date, Classe\_Finish\_Date finishDate, DayOfWeek day, Acronym teacherAcronym) throws IntegrityViolationException, ConcurrencyException {  
  
  
 final Classe newClasse = new Classe(title, start\_date, finishDate, startTime, finishTime, day, teacherAcronym);  
  
 if (checkClassConflict(newClasse)) {  
 throw new IllegalStateException("There is a scheduling conflict with this class.");  
 } else {  
 return toDTO(this.repository.save(newClasse));  
 }  
 }  
  
  
 public Teacher findTeacherBySystemUser(SystemUser systemUser) {  
 return scs.findTeacherBySystemUser(systemUser);  
 }  
  
 public boolean checkClassConflict(Classe classeToCheck) {  
 List<Classe> allClasses = (List<Classe>) repository.findAll();  
 for (Classe existingClass : allClasses) {  
 if (existingClass.getTitle().equals(classeToCheck.getTitle())) {  
 // Skip checking against the same class  
 continue;  
 }  
  
 // Only compare classes on the same date  
 if (existingClass.getStartDate().equals(classeToCheck.getStartDate())) {  
 // Check for time overlap  
 if (isTimeOverlapUserInput(classeToCheck.getStartTime(), classeToCheck.getFinishTime(), existingClass)) {  
 return true; // Conflict found  
 }  
 }  
 }  
  
 return false; // No conflict found  
 }  
  
 /\*\*  
 \* Checks if the class specified by userInputStartTime and userInputEndTime overlaps  
 \* with the existing class's start and end times.  
 \*  
 \* @param userInputStartTime The start time of the class to check.  
 \* @param userInputEndTime The end time of the class to check.  
 \* @param existingClass The class with which to compare the given class.  
 \* @return True if the given class overlaps with the existing class, false otherwise.  
 \*/  
 public boolean isTimeOverlapUserInput(Classe\_Start\_Time userInputStartTime, Classe\_Finish\_Time userInputEndTime, Classe existingClass) {  
 // Extract the LocalTime from userInputStartTime and userInputEndTime  
 LocalTime userInputStart = userInputStartTime.getStartTime();  
 LocalTime userInputEnd = userInputEndTime.getFinish\_time();  
  
 // Convert start and end times of user input to minutes since midnight  
 int userInputStartMinutes = userInputStart.getHour() \* 60 + userInputStart.getMinute();  
 int userInputEndMinutes = userInputEnd.getHour() \* 60 + userInputEnd.getMinute();  
  
 // Extract the LocalTime from existingClass's start and end time  
 LocalTime existingStartTime = existingClass.getStartTime().getStartTime();  
 LocalTime existingFinishTime = existingClass.getFinishTime().getFinish\_time();  
  
 // Convert start and end times of existing class to minutes since midnight  
 int existingStartMinutes = existingStartTime.getHour() \* 60 + existingStartTime.getMinute();  
 int existingEndMinutes = existingFinishTime.getHour() \* 60 + existingFinishTime.getMinute();  
  
 // Check for overlap: does user input class start during existing class?  
 if (userInputStartMinutes >= existingStartMinutes && userInputStartMinutes < existingEndMinutes) {  
 return true; // Overlap found  
 }  
  
 // Check for overlap: does existing class start during user input class?  
 if (existingStartMinutes >= userInputStartMinutes && existingStartMinutes < userInputEndMinutes) {  
 return true; // Overlap found  
 }  
  
 // If neither condition above is met, there is no overlap  
 return false; // No overlap found  
 }  
  
  
 public Classe updateClassSchedule(Classe\_Title title,Classe\_Start\_Time newStartTime, Classe\_Finish\_Time newFinishTime) throws IntegrityViolationException, ConcurrencyException {  
  
 Optional<Classe> optionalClass = this.repository.ofIdentity(title);  
 if (optionalClass.isPresent()) {  
 Classe classToUpdate = optionalClass.get();  
 classToUpdate.changeStartTime(newStartTime);  
 classToUpdate.changeFinishTime(newFinishTime);  
 return this.repository.save(classToUpdate);  
 } else {  
 throw new IllegalArgumentException("No class found with title: " + title);  
 }  
 }  
  
 public List<Classe> getAllClasses(Acronym acronym) {  
 return repository.findClassesByTeacher(String.valueOf(acronym));  
 }  
  
  
  
 public ClasseDTO toDTO(Classe classe) {  
 return new ClasseDTO(  
 classe.getTitle().getTitle(),  
 classe.getStartTime().getStartTime(),  
 classe.getFinishTime().getFinish\_time(),  
 classe.getStartDate().getStart\_date(),  
 classe.getFinishDate().getFinish\_date(),  
 classe.getDay().getDay(),  
 classe.getAcronym().getValue()  
 );  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\aplication\ClassController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\aplication\ClasseDTO.java

package eapli.base.Classe.aplication;  
  
  
import java.time.LocalDate;  
import java.time.LocalTime;  
  
public class ClasseDTO {  
 public String title;  
 public LocalTime startTime;  
 public LocalTime finishTime;  
 public LocalDate startDate;  
 public LocalDate finishDate;  
 public int dayOfWeek;  
 public String teacherAcronym;  
  
 public ClasseDTO( String title, LocalTime startTime, LocalTime finishTime, LocalDate startDate, LocalDate finishDate, int dayOfWeek, String teacherAcronym) {  
 this.title = title;  
 this.startTime = startTime;  
 this.finishTime = finishTime;  
 this.startDate = startDate;  
 this.finishDate = finishDate;  
 this.dayOfWeek = dayOfWeek;  
 this.teacherAcronym = teacherAcronym;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\aplication\ClasseDTO.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\aplication\ScheduleClassService.java

package eapli.base.Classe.aplication;  
  
import eapli.base.Classe.domain.Classe;  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.ApplicationService;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
@ApplicationService  
class ScheduleClassService {  
 private final AuthorizationService authz;  
 private final ClassRepository classRepository;  
 private final TeacherRepository teacherRepository;  
  
 public ScheduleClassService(final AuthorizationService authz, final ClassRepository classRepository, final TeacherRepository teacherRepository) {  
 this.authz = authz;  
 this.classRepository = classRepository;  
 this.teacherRepository = teacherRepository;  
 }  
  
 /\*\*  
 \* Schedules a new class  
 \*  
 \* @param classe The class to be scheduled  
 \* @return the scheduled class  
 \*/  
 public Classe scheduleClass(Classe classe) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
 return classRepository.save(classe);  
 }  
  
 /\*\*  
 \* Returns all teachers  
 \*  
 \* @return all teachers  
 \*/  
 public Iterable<Teacher> teachers() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
 return teacherRepository.findAll();  
 }  
 public Teacher findTeacherBySystemUser(SystemUser systemUser) {  
 return teacherRepository.findTeacherBySystemUser(systemUser);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\aplication\ScheduleClassService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe.java

package eapli.base.Classe.domain;  
  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.\*;  
  
@Entity  
@Inheritance(strategy = InheritanceType.JOINED)  
public class Classe implements AggregateRoot<Classe\_Title> {  
  
 @Column(unique = true)  
 @EmbeddedId  
 private Classe\_Title title;  
  
 private Classe\_Start\_Date start\_date;  
 private Classe\_Finish\_Date finish\_date;  
 private Classe\_Start\_Time start\_time;  
 private Classe\_Finish\_Time finish\_time;  
 private DayOfWeek day;  
 private Acronym acronym;  
  
  
 protected Classe() {  
 }  
  
 public Classe(Classe\_Title title, Classe\_Start\_Date start\_date, Classe\_Finish\_Date finish\_date, Classe\_Start\_Time start\_time, Classe\_Finish\_Time finish\_time, DayOfWeek day, Acronym acronym) {  
 Preconditions.nonNull(title, "Title should not be null");  
 this.title = title;  
 this.start\_date = start\_date;  
 this.finish\_date = finish\_date;  
 this.start\_time = start\_time;  
 this.finish\_time = finish\_time;  
 this.day = day;  
 this.acronym = acronym;  
 }  
  
 public void changeTitle(Classe\_Title newTitle) {  
 Preconditions.nonNull(newTitle, "Title should not be null");  
 this.title = newTitle;  
 }  
  
 public void changeStartTime(Classe\_Start\_Time newStartTime) {  
 Preconditions.nonNull(newStartTime, "Start time should not be null");  
 this.start\_time = newStartTime;  
 }  
  
 public void changeFinishTime(Classe\_Finish\_Time newFinishTime) {  
 Preconditions.nonNull(newFinishTime, "Finish time should not be null");  
 this.finish\_time = newFinishTime;  
 }  
  
 @Override  
 public Classe\_Title identity() {  
 return this.title;  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 return other instanceof Classe && this.title.equals(((Classe)other).title);  
 }  
  
 public Classe\_Title getTitle() {  
 return title;  
 }  
  
 public Classe\_Start\_Date getStartDate() {  
 return start\_date;  
 }  
  
 public Classe\_Finish\_Date getFinishDate() {  
 return finish\_date;  
 }  
  
 public Classe\_Start\_Time getStartTime() {  
 return start\_time;  
 }  
  
 public Classe\_Finish\_Time getFinishTime() {  
 return finish\_time;  
 }  
  
 public DayOfWeek getDay() {  
 return day;  
 }  
  
 public Acronym getAcronym() {  
 return acronym;  
 }  
  
  
 public void setTitle(Classe\_Title title) {  
 this.title = title;  
 }  
  
 public void setStart\_date(Classe\_Start\_Date start\_date) {  
 this.start\_date = start\_date;  
 }  
  
 public void setFinish\_date(Classe\_Finish\_Date finish\_date) {  
 this.finish\_date = finish\_date;  
 }  
  
 public void setStart\_time(Classe\_Start\_Time start\_time) {  
 this.start\_time = start\_time;  
 }  
  
 public void setFinish\_time(Classe\_Finish\_Time finish\_time) {  
 this.finish\_time = finish\_time;  
 }  
  
 public void setDay(DayOfWeek day) {  
 this.day = day;  
 }  
  
 public void setAcronym(Acronym acronym) {  
 this.acronym = acronym;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Finish\_Date.java

package eapli.base.Classe.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import org.springframework.format.annotation.DateTimeFormat;  
  
import javax.persistence.Column;  
import javax.persistence.Embeddable;  
import java.time.LocalDate;  
  
@Embeddable  
public class Classe\_Finish\_Date implements ValueObject {  
 @DateTimeFormat(pattern = "yyyy-MM-dd")  
 @Column(nullable = false)  
 private LocalDate finish\_date;  
  
 // No argument constructor, used by JPA/Hibernate  
 protected Classe\_Finish\_Date() {}  
  
 public Classe\_Finish\_Date(LocalDate finish\_date) {  
 if (finish\_date == null) {  
 throw new IllegalArgumentException("Finish date cannot be null");  
 }  
 this.finish\_date = finish\_date;  
 }  
  
 public static Classe\_Finish\_Date valueOf(LocalDate finish\_date) {  
 return new Classe\_Finish\_Date(finish\_date);  
 }  
  
 @Override  
 public String toString() {  
 return finish\_date.toString();  
 }  
  
 public LocalDate getFinish\_date() {  
 return finish\_date;  
 }  
  
 public void setFinish\_date(LocalDate finish\_date) {  
 if (finish\_date == null) {  
 throw new IllegalArgumentException("Finish date cannot be null");  
 }  
 this.finish\_date = finish\_date;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Finish\_Date.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Finish\_Time.java

package eapli.base.Classe.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
import java.time.LocalTime;  
  
@Embeddable  
public class Classe\_Finish\_Time implements ValueObject {  
 private LocalTime finish\_time;  
  
 protected Classe\_Finish\_Time() {  
 }  
  
 public Classe\_Finish\_Time(LocalTime finish\_time) throws IllegalArgumentException{  
 Preconditions.ensure(finish\_time != null, "Invalid finish time");  
 this.finish\_time = finish\_time;  
 }  
  
 public static Classe\_Finish\_Time from(LocalTime finish\_time) throws IllegalArgumentException{  
 return new Classe\_Finish\_Time(finish\_time);  
 }  
  
 public static Classe\_Finish\_Time valueOf(LocalTime finish\_time) throws IllegalArgumentException{  
 return from(finish\_time);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Finish Time: %s", finish\_time);  
 }  
  
 public LocalTime getFinish\_time() {  
 return finish\_time;  
 }  
  
 public void setFinish\_time(LocalTime finish\_time) {  
 Preconditions.ensure(finish\_time != null, "Invalid finish time");  
 this.finish\_time = finish\_time;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Finish\_Time.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Start\_Date.java

package eapli.base.Classe.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import org.springframework.format.annotation.DateTimeFormat;  
  
import javax.persistence.Column;  
import javax.persistence.Embeddable;  
import java.time.LocalDate;  
import java.util.Objects;  
  
@Embeddable  
public class Classe\_Start\_Date implements ValueObject {  
 @DateTimeFormat(pattern = "yyyy-MM-dd")  
 @Column(nullable = false)  
 private LocalDate start\_date;  
 protected Classe\_Start\_Date() {  
 }  
  
 public Classe\_Start\_Date(LocalDate start\_date) {  
 if (start\_date == null) {  
 throw new IllegalArgumentException("Start date must not be null");  
 }  
 this.start\_date = start\_date;  
 }  
  
 public static Classe\_Start\_Date valueOf(LocalDate start\_date) {  
 return new Classe\_Start\_Date(start\_date);  
 }  
  
 public LocalDate getStart\_date() {  
 return this.start\_date;  
 }  
  
 @Override  
 public String toString() {  
 return start\_date != null ? start\_date.toString() : "";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Start\_Date.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Start\_Time.java

package eapli.base.Classe.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
import javax.persistence.Column;  
import java.time.LocalTime;  
  
@Embeddable  
public class Classe\_Start\_Time implements ValueObject {  
  
 @Column(nullable = false)  
 private LocalTime start\_time;  
  
  
 protected Classe\_Start\_Time() {  
  
 }  
  
 public Classe\_Start\_Time(LocalTime start\_time) {  
 Preconditions.ensure(start\_time != null, "Invalid start time");  
 this.start\_time = start\_time;  
 }  
  
 public static Classe\_Start\_Time from(LocalTime start\_time) {  
 return new Classe\_Start\_Time(start\_time);  
 }  
  
 public static Classe\_Start\_Time valueOf(LocalTime start\_time) {  
 return from(start\_time);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Start Time: %s", start\_time);  
 }  
  
 public LocalTime getStartTime() {  
 return this.start\_time;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Start\_Time.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Title.java

package eapli.base.Classe.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
import javax.persistence.Column;  
import java.util.Objects;  
  
@Embeddable  
public class Classe\_Title implements ValueObject, Comparable<Classe\_Title> {  
  
 @Column(nullable = false)  
 private String title;  
  
  
 protected Classe\_Title() {  
 }  
  
 public Classe\_Title(String title) {  
 Preconditions.ensure(title != null && !title.trim().isEmpty(), "Title invalido");  
 this.title = title;  
 }  
  
 public static Classe\_Title from(String title) {  
 return new Classe\_Title(title);  
 }  
  
 public static Classe\_Title valueOf(String title) {  
 return from(title);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Title : %s", title);  
 }  
  
 public String getTitle() {  
 return this.title;  
 }  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 Classe\_Title other = (Classe\_Title) o;  
 return Objects.equals(title, other.title);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(title);  
 }  
  
  
 @Override  
 public int compareTo(Classe\_Title o) {  
 return this.title.compareTo(o.title);  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\Classe\_Title.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\DayOfWeek.java

package eapli.base.Classe.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
  
@Embeddable  
public class DayOfWeek implements ValueObject {  
  
 private int day;  
  
 protected DayOfWeek() {  
 }  
  
 public DayOfWeek(int day) throws IllegalArgumentException {  
 Preconditions.ensure(day >= 1 && day <= 7, "Invalid day of week");  
 this.day = day;  
 }  
  
 public static DayOfWeek valueOf(int day) throws IllegalArgumentException {  
 return new DayOfWeek(day);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Day of Week: %d", day);  
 }  
  
 public int getDay() {  
 return this.day;  
 }  
  
 public void setDay(int day) {  
 Preconditions.ensure(day >= 1 && day <= 7, "Invalid day of week");  
 this.day = day;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\domain\DayOfWeek.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\repository\ClassRepository.java

package eapli.base.Classe.repository;  
  
import eapli.base.Classe.domain.Classe;  
import eapli.base.Classe.domain.Classe\_Title;  
import eapli.framework.domain.repositories.DomainRepository;  
  
import java.util.List;  
  
public interface ClassRepository extends DomainRepository<Classe\_Title, Classe> {  
 List<Classe> findClassesByTeacher(String teacherAcronym);  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Classe\repository\ClassRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\CourseController.java

/\*  
package eapli.base.Course.Application;  
  
import eapli.base.Course.Domain.Course;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/courses")  
public class CourseController {  
 private final CourseService courseService;  
  
 @Autowired  
 public CourseController(CourseService courseService) {  
 this.courseService = courseService;  
 }  
  
 @GetMapping("/search")  
 public List<Course> getCoursesByName(@RequestParam String courseName) {  
 //return courseService.getCoursesByName(courseName); Necessário corrigir isto com prioridade com brevidade possível  
 return null;  
 }  
  
 @PostMapping("/{id}/open-enrollment")  
 public Course openEnrollment(@PathVariable Long id) {  
 return courseService.openEnrollments(id);  
 }  
  
 @PostMapping("/{id}/close-enrollment")  
 public Course closeEnrollment(@PathVariable Long id) {  
 return courseService.closeEnrollments(id);  
 }  
  
 @PostMapping("/{id}/open-course")  
 public Course openCourse(@PathVariable Long id) {  
 return courseService.openCourse(id);  
 }  
  
 @PostMapping("/{id}/close-course")  
 public Course closeCourse(@PathVariable Long id) {  
 return courseService.closeCourse(id);  
 }  
  
}  
\*/

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\CourseController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\CourseService.java

package eapli.base.Course.Application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Domain.Course\_ID;  
import eapli.base.Course.Domain.Course\_Name;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.framework.application.ApplicationService;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import javax.persistence.EntityNotFoundException;  
import java.util.List;  
import java.util.Optional;  
  
@ApplicationService  
public class CourseService {  
 private final CourseRepository courseRepository;  
 private final AuthorizationService authz;  
 @Autowired  
 public CourseService(final AuthorizationService authz, CourseRepository courseRepository) {  
 this.authz = authz;  
 this.courseRepository = courseRepository;  
 }  
  
 public Course createCourse(Course course) {  
 return courseRepository.save(course);  
 }  
  
 public Course openEnrollments(Course\_Name courseName) {  
 Optional<Course> courseOpt = courseRepository.ofIdentity(courseName);  
 if (courseOpt.isPresent()) {  
 Course course = courseOpt.get();  
 course.setEnrollmentOpen(true);  
 return courseRepository.save(course);  
 } else {  
 throw new EntityNotFoundException("Course not found with ID: " + courseName);  
 }  
 }  
 public Course closeEnrollments(Course\_Name courseName) {  
 Optional<Course> courseOpt = courseRepository.ofIdentity(courseName);  
 if (courseOpt.isPresent()) {  
 Course course = courseOpt.get();  
 course.setEnrollmentOpen(false);  
 return courseRepository.save(course);  
 } else {  
 throw new EntityNotFoundException("Course not found with ID: " + courseName);  
 }  
 }  
  
 public Course openCourse(Course\_Name courseName) {  
 Optional<Course> courseOpt = courseRepository.ofIdentity(courseName);  
 if (courseOpt.isPresent()) {  
 Course course = courseOpt.get();  
 course.setIsOpen(true);  
 return courseRepository.save(course);  
 } else {  
 throw new EntityNotFoundException("Course not found with ID: " + courseName);  
 }  
 }  
  
 public Course closeCourse(Course\_Name courseName) {  
 Optional<Course> courseOpt = courseRepository.ofIdentity(courseName);  
 if (courseOpt.isPresent()) {  
 Course course = courseOpt.get();  
 course.setIsOpen(false);  
 return courseRepository.save(course);  
 } else {  
 throw new EntityNotFoundException("Course not found with ID: " + courseName);  
 }  
 }  
  
  
 /\*public Course openEnrollments(Course course) {  
 Course course = courseRepository.ofIdentity(courseId);  
 course.setEnrollmentOpen(true);  
 return courseRepository.save(course);  
 }  
  
 public Course closeEnrollments(Long courseId) {  
 Course course = courseRepository.ofIdentity(courseId);  
 course.setEnrollmentOpen(false);  
 return courseRepository.save(course);  
 }\*/  
  
 /\*public List<Course> getCoursesByName(String courseName) {  
 return courseRepository.findByCourseName(courseName);  
 }\*/  
  
 /\*public Course findCourseById(Long courseId) {  
 return courseRepository.findById(courseId)  
 .orElseThrow(() -> new IllegalArgumentException("Course not found"));  
 }  
  
 public Course openEnrollment(Long courseId) {  
 Course course = courseRepository.findById(courseId).orElseThrow(() -> new IllegalArgumentException("Curso não encontrado"));  
 course.setEnrollmentOpen(true);  
 return courseRepository.save(course);  
 }  
  
 public Course closeEnrollment(Long courseId) {  
 Course course = courseRepository.findById(courseId).orElseThrow(() -> new IllegalArgumentException("Curso não encontrado"));  
 course.setEnrollmentOpen(false);  
 return courseRepository.save(course);  
 }  
  
 public Course openCourse(Long courseId) {  
 Course course = courseRepository.findById(courseId)  
 .orElseThrow(() -> new IllegalArgumentException("Curso não encontrado"));  
 course.setIsOpen(true);  
 return courseRepository.save(course);  
 }  
  
 public Course closeCourse(Long courseId) {  
 Course course = courseRepository.findById(courseId)  
 .orElseThrow(() -> new IllegalArgumentException("Curso não encontrado"));  
 course.setIsOpen(false);  
 return courseRepository.save(course);  
 }\*/  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\CourseService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\CreateCourseController.java

package eapli.base.Course.Application;  
  
import eapli.base.Course.Domain.\*;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
  
public class CreateCourseController {  
  
 private final CourseRepository courseRepository;  
 private final TeacherRepository teacherRepository;  
  
 public CreateCourseController(final CourseRepository courseRepository, final TeacherRepository teacherRepository) {  
 this.courseRepository = courseRepository;  
 this.teacherRepository=teacherRepository;  
 }  
  
 public void createCourse(String name, int min, int max, String description, Teacher teacher) {  
 courseRepository.save(new Course(new Course\_Name(name),new Maximum\_Number\_Of\_Students(max),new Minimum\_Number\_Of\_Students(min),new Small\_Textual\_Description(description), teacher));  
 }  
  
 public Iterable<Teacher> listTeachers(){  
 return this.teacherRepository.findAll();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\CreateCourseController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\ListCoursesController.java

package eapli.base.Course.Application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Repository.CourseRepository;  
  
public class ListCoursesController {  
  
 private final ListCoursesService svc;  
  
 public ListCoursesController(final CourseRepository courseRepository) {  
 // dependency injection - to make this object more testable we don't create the  
 // infrastructure objects to avoid coupling to the implementation. This way, the controller  
 // can be used in different scenarios with different implementations of the repository. for  
 // instance, unit testing.  
 svc = new ListCoursesService(courseRepository);  
 }  
  
 public Iterable<Course> listCourses() {  
 return svc.courses();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\ListCoursesController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\ListCoursesService.java

package eapli.base.Course.Application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Repository.CourseRepository;  
  
public class ListCoursesService {  
  
 private final CourseRepository courseRepository;  
  
 public ListCoursesService(final CourseRepository courseRepository) {  
 this.courseRepository = courseRepository;  
 }  
  
 public Iterable<Course> courses() {  
 return courseRepository.findAll();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\ListCoursesService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\OpenCloseCourseController.java

package eapli.base.Course.Application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Domain.Course\_ID;  
import eapli.base.Course.Domain.Course\_Name;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
  
public class OpenCloseCourseController {  
  
 private final CourseService cs;  
  
 public OpenCloseCourseController(final AuthorizationService authz, final CourseRepository courseRepository) {  
 cs = new CourseService(authz,courseRepository);  
 }  
  
 public Course openCourse(Course\_Name courseName) {  
 return cs.openCourse(courseName);  
 }  
  
 public Course closeCourse(Course\_Name courseName) {  
 return cs.closeCourse(courseName);  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\OpenCloseCourseController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\OpenCloseEnrollementsController.java

package eapli.base.Course.Application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Domain.Course\_ID;  
import eapli.base.Course.Domain.Course\_Name;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
  
public class OpenCloseEnrollementsController {  
  
 private final CourseService cs;  
  
 public OpenCloseEnrollementsController(final AuthorizationService authz, final CourseRepository courseRepository) {  
 cs = new CourseService(authz,courseRepository);  
 }  
  
 public Course openEnrollments(Course\_Name courseName) {  
 return cs.openEnrollments(courseName);  
 }  
  
 public Course closeEnrollments(Course\_Name courseName) {  
 return cs.closeEnrollments(courseName);  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Application\OpenCloseEnrollementsController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Course.java

package eapli.base.Course.Domain;  
  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.framework.domain.model.AggregateRoot;  
  
import javax.persistence.\*;  
import java.util.HashSet;  
import java.util.Set;  
  
@Entity  
@Table(name = "COURSE")  
public class Course implements AggregateRoot<Course\_Name> {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 private Long id;  
  
 @Embedded  
 @Column(unique = true)  
 private Course\_Name courseName;  
  
 private Maximum\_Number\_Of\_Students maximumNumberOfStudents;  
 private Minimum\_Number\_Of\_Students minimumNumberOfStudents;  
 private Small\_Textual\_Description smallTextualDescription;  
  
 private boolean enrollmentOpen;  
 private boolean isOpen;  
  
  
 @ManyToOne  
 @JoinColumn(name = "teacher\_in\_charge\_id")  
 private Teacher teacherInCharge;  
  
 @ManyToMany  
 @JoinTable(  
 name = "course\_student",  
 joinColumns = @JoinColumn(name = "course\_id"),  
 inverseJoinColumns = @JoinColumn(name = "student\_id"))  
 private Set<Student> students;  
  
 public Course() {  
 }  
  
 public void addStudent(Student student) {  
 students.add(student);  
 }  
  
  
 public void setIsOpen(boolean isOpen) {  
 this.isOpen = isOpen;  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 return false;  
 }  
  
 @Override  
 public Course\_Name identity() {  
 return courseName;  
 }  
  
  
 public Course(Course\_Name courseName, Maximum\_Number\_Of\_Students maximumNumberOfStudents, Minimum\_Number\_Of\_Students minimumNumberOfStudents, Small\_Textual\_Description smallTextualDescription, Teacher teacher){  
 this.courseName=courseName;  
 this.maximumNumberOfStudents=maximumNumberOfStudents;  
 this.minimumNumberOfStudents=minimumNumberOfStudents;  
 this.smallTextualDescription=smallTextualDescription;  
 this.teacherInCharge=teacher;  
 this.students=new HashSet<>();  
  
 this.isOpen = false;  
 this.enrollmentOpen = false;  
 }  
  
 public Course\_Name getCourseName() {  
 return courseName;  
 }  
  
 public Maximum\_Number\_Of\_Students getMaximumNumberOfStudents() {  
 return maximumNumberOfStudents;  
 }  
  
 public Minimum\_Number\_Of\_Students getMinimumNumberOfStudents() {  
 return minimumNumberOfStudents;  
 }  
  
 public Small\_Textual\_Description getSmallTextualDescription() {  
 return smallTextualDescription;  
 }  
  
 public Teacher getTeacherInCharge() {  
 return teacherInCharge;  
 }  
  
 public Set<Student> getStudents() {  
 return students;  
 }  
  
 public void setCourseName(Course\_Name courseName) {  
 this.courseName = courseName;  
 }  
  
 public void setMaximumNumberOfStudents(Maximum\_Number\_Of\_Students maximumNumberOfStudents) {  
 this.maximumNumberOfStudents = maximumNumberOfStudents;  
 }  
  
 public void setMinimumNumberOfStudents(Minimum\_Number\_Of\_Students minimumNumberOfStudents) {  
 this.minimumNumberOfStudents = minimumNumberOfStudents;  
 }  
  
 public void setSmallTextualDescription(Small\_Textual\_Description smallTextualDescription) {  
 this.smallTextualDescription = smallTextualDescription;  
 }  
  
 public void setEnrollmentOpen(boolean enrollmentOpen) {  
 this.enrollmentOpen = enrollmentOpen;  
 }  
  
 public void setOpen(boolean open) {  
 isOpen = open;  
 }  
  
 public void setTeacherInCharge(Teacher teacherInCharge) {  
 this.teacherInCharge = teacherInCharge;  
 }  
  
 public void setStudents(Set<Student> students) {  
 this.students = students;  
 }  
  
 @Override  
 public String toString() {  
 return id.toString();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Course.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Course\_ID.java

package eapli.base.Course.Domain;  
  
import javax.persistence.\*;  
import lombok.\*;  
  
@Embeddable  
@EqualsAndHashCode  
public class Course\_ID implements Comparable<Course\_ID> {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 private Long id;  
  
 private String value;  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getValue() {  
 return value;  
 }  
  
 public void setValue(String value) {  
 this.value = value;  
 }  
  
 protected Course\_ID(){}  
  
 public Course\_ID(Long id){  
 this.id=id;  
 }  
 @Override  
 public int compareTo(Course\_ID o) {  
 return 0;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Course\_ID.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Course\_Name.java

package eapli.base.Course.Domain;  
import javax.persistence.\*;  
  
import eapli.framework.domain.model.ValueObject;  
import lombok.\*;  
  
@Embeddable  
public class Course\_Name implements ValueObject, Comparable<Course\_Name>{  
 private String value;  
  
 public Course\_Name() {  
  
 }  
  
 public String getValue() {  
 return value;  
 }  
  
 public Course\_Name(String value){  
 this.value=value;  
 }  
  
 public void setValue(String value) {  
 this.value = value;  
 }  
  
 @Override  
 public int compareTo(Course\_Name o) {  
 return 0;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Course\_Name.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Maximum\_Number\_Of\_Students.java

package eapli.base.Course.Domain;  
import javax.persistence.\*;  
  
import eapli.framework.domain.model.ValueObject;  
import lombok.\*;  
  
@Embeddable  
public class Maximum\_Number\_Of\_Students implements ValueObject {  
  
 private int max\_num;  
  
 protected Maximum\_Number\_Of\_Students(){}  
  
 public Maximum\_Number\_Of\_Students(int value){this.max\_num=value;  
 }  
  
 public int getValue() {  
 return max\_num;  
 }  
  
 public void setValue(int value) {  
 this.max\_num = value;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Maximum\_Number\_Of\_Students.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Minimum\_Number\_Of\_Students.java

package eapli.base.Course.Domain;  
import javax.persistence.\*;  
  
import eapli.framework.domain.model.ValueObject;  
import lombok.\*;  
  
  
@Embeddable  
public class Minimum\_Number\_Of\_Students implements ValueObject {  
  
 private int min\_num;  
  
 public Minimum\_Number\_Of\_Students(){  
 }  
 public Minimum\_Number\_Of\_Students(int value){  
 this.min\_num=value;  
 }  
  
 public int getValue() {  
 return min\_num;  
 }  
  
 public void setValue(int value) {  
 this.min\_num = value;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Minimum\_Number\_Of\_Students.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Small\_Textual\_Description.java

package eapli.base.Course.Domain;  
import javax.persistence.\*;  
  
import eapli.framework.domain.model.ValueObject;  
import lombok.\*;  
  
  
@Embeddable  
public class Small\_Textual\_Description implements ValueObject {  
   
 private String desc;  
  
 public Small\_Textual\_Description(String value){  
 this.desc=value;  
 }  
  
 public Small\_Textual\_Description() {  
  
 }  
  
 public String getValue() {  
 return desc;  
 }  
  
 public void setValue(String value) {  
 this.desc = value;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Domain\Small\_Textual\_Description.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Repository\CourseRepository.java

package eapli.base.Course.Repository;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Domain.Course\_ID;  
import eapli.base.Course.Domain.Course\_Name;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.framework.domain.repositories.DomainRepository;  
import org.springframework.stereotype.Repository;  
  
import javax.persistence.EntityNotFoundException;  
  
@Repository  
public interface CourseRepository extends DomainRepository<Course\_Name,Course> {  
  
 Iterable<Course> findCourseByName(String courseName);  
  
  
 /\*default Course openCourse(Long courseId) {  
 Course course = this.ofIdentity(courseId)  
 .orElseThrow(() -> new EntityNotFoundException("Course not found with ID: " + courseId));  
 course.setIsOpen(true);  
 return this.save(course);  
 }  
 default Course closeCourse(Long courseId) {  
 Course course = this.ofIdentity(courseId)  
 .orElseThrow(() -> new EntityNotFoundException("Course not found with ID: " + courseId));  
 course.setIsOpen(false);  
 return this.save(course);  
 }  
  
 default Course openEnrollments(Long courseId) {  
 Course course = this.ofIdentity(courseId)  
 .orElseThrow(() -> new EntityNotFoundException("Course not found with ID: " + courseId));  
 course.setEnrollmentOpen(true);  
 return this.save(course);  
 }  
  
 default Course closeEnrollments(Long courseId) {  
 Course course = this.ofIdentity(courseId)  
 .orElseThrow(() -> new EntityNotFoundException("Course not found with ID: " + courseId));  
 course.setEnrollmentOpen(false);  
 return this.save(course);  
 }\*/  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Course\Repository\CourseRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\ApproveEnrollmentController.java

package eapli.base.Enrollment.Application;  
  
import eapli.base.Enrollment.Domain.Enrollment;  
import eapli.base.Enrollment.Repository.EnrollmentRepository;  
  
public class ApproveEnrollmentController {  
  
 private final EnrollmentRepository enrollmentRepository;  
  
 public ApproveEnrollmentController(EnrollmentRepository enrollmentRepository) {  
 this.enrollmentRepository = enrollmentRepository;  
 }  
  
 public void approveEnrollment(Enrollment enrollment, int isToBeAccepted) {  
 if (isToBeAccepted == 1) {  
 enrollment.acceptEnrollment();  
 } else {  
 enrollment.rejectEnrollment();  
 }  
 enrollmentRepository.save(enrollment);  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\ApproveEnrollmentController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\BulkEnrollmentController.java

package eapli.base.Enrollment.Application;  
  
import eapli.base.Enrollment.Domain.BulkEnrollment.BulkEnrollment;  
import eapli.base.Enrollment.Repository.EnrollmentRepository;  
  
public class BulkEnrollmentController {  
 private BulkEnrollment bulkEnrollment;  
 private EnrollmentRepository enrollmentRepository;  
  
 public BulkEnrollmentController(EnrollmentRepository enrollmentRepository) {  
 this.enrollmentRepository=enrollmentRepository;  
 }  
  
 public void bulkEnrollment(String csvFilePath) {  
 bulkEnrollment.bulkEnrollmentReadFile(csvFilePath);  
 //save is given in bulkEnrollmentReadFile  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\BulkEnrollmentController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\ListEnrollmentsController.java

package eapli.base.Enrollment.Application;  
  
import eapli.base.Enrollment.Domain.Enrollment;  
import eapli.base.Enrollment.Repository.EnrollmentRepository;  
  
  
public class ListEnrollmentsController {  
 private final ListEnrollmentsService svc;  
  
 public ListEnrollmentsController(final EnrollmentRepository enrollmentRepository) {  
  
 svc = new ListEnrollmentsService(enrollmentRepository);  
 }  
  
  
  
 public Iterable<Enrollment> listEnrollments() {  
 return svc.enrollments();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\ListEnrollmentsController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\ListEnrollmentsService.java

package eapli.base.Enrollment.Application;  
  
import eapli.base.Enrollment.Domain.Enrollment;  
import eapli.base.Enrollment.Repository.EnrollmentRepository;  
  
public class ListEnrollmentsService {  
  
 private final EnrollmentRepository enrollmentRepository;  
  
 public ListEnrollmentsService(final EnrollmentRepository enrollmentRepository) {  
 this.enrollmentRepository = enrollmentRepository;  
 }  
  
 public Iterable<Enrollment> enrollments() {  
 return enrollmentRepository.findAll();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\ListEnrollmentsService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\RequestEnrollmentController.java

package eapli.base.Enrollment.Application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Enrollment.Domain.Enrollment;  
import eapli.base.Enrollment.Domain.EnrollmentDescription;  
import eapli.base.Enrollment.Repository.EnrollmentRepository;  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
  
public class RequestEnrollmentController {  
 private final EnrollmentRepository enrollmentRepository;  
 private final StudentRepository studentRepository;  
 public RequestEnrollmentController(final EnrollmentRepository enrollmentRepository, final StudentRepository studentRepository) {  
 this.enrollmentRepository = enrollmentRepository;  
 this.studentRepository = studentRepository;  
 }  
  
 public void requestEnrollment(Student student, Course course, String description) {  
 enrollmentRepository.save(new Enrollment(student,course,new EnrollmentDescription(description)));  
 }  
  
 public Student findStudentBySystemUser(SystemUser systemUser) {  
 return studentRepository.findStudentBySystemUser(systemUser);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Application\RequestEnrollmentController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\BulkEnrollment\BulkEnrollment.java

package eapli.base.Enrollment.Domain.BulkEnrollment;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.Enrollment.Application.RequestEnrollmentController;  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
  
import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
  
public class BulkEnrollment {  
 private final RequestEnrollmentController requestEnrollmentController;  
 private final StudentRepository studentRepository;  
 private final CourseRepository courseRepository;  
  
 public BulkEnrollment(RequestEnrollmentController requestEnrollmentController, StudentRepository studentRepository, CourseRepository courseRepository) {  
 this.requestEnrollmentController = requestEnrollmentController;  
 this.studentRepository = studentRepository;  
 this.courseRepository = courseRepository;  
 }  
  
 public void bulkEnrollmentReadFile(String csvFilePath) {  
  
 try (BufferedReader reader = new BufferedReader(new FileReader(csvFilePath))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] data = line.split(",");  
  
 try {  
 Student student = studentRepository.findStudentByMechanographicNumber(data[0].trim());  
 Iterable<Course> it = courseRepository.findCourseByName(data[1].trim());  
 Course course = it.iterator().next();  
 String description = data[2].trim();  
 try {  
 requestEnrollmentController.requestEnrollment(student, course, description);  
 } catch (final IntegrityViolationException e) {  
 System.out.println("That enrollment is invalid.");  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\BulkEnrollment\BulkEnrollment.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\Enrollment.java

package eapli.base.Enrollment.Domain;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.framework.domain.model.AggregateRoot;  
  
import javax.persistence.\*;  
  
@Entity  
@Inheritance(strategy = InheritanceType.JOINED)  
public class Enrollment implements AggregateRoot<Long> {  
  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name = "EnrollmentID", nullable = false)  
 private Long id;  
  
 @Column(name = "EnrollmentStudent", nullable = false)  
 private Student student;  
  
 @Column(name = "EnrollmentCourse", nullable = false)  
 private Course course;  
  
 @Column(name = "Description", nullable = false)  
 private EnrollmentDescription description;  
  
 @Column(name="EnrollmentState", nullable = false)  
 private EnrollmentState state;  
  
 protected Enrollment(){}  
  
  
 public Enrollment(Student student, Course course, EnrollmentDescription description) {  
 this.student = student;  
 this.course = course;  
 this.description = description;  
 this.state = EnrollmentState.PENDING;//valor default(Só fica aceite a mando do manager)  
 }  
  
  
 public void acceptEnrollment() {  
 this.state = EnrollmentState.ACCEPTED;  
 }  
  
 public void rejectEnrollment() {  
 this.state = EnrollmentState.REJECTED;  
 }  
  
  
  
 @Override  
 public Long identity() {  
 return id;  
 }  
  
 public Student getStudent() {  
 return student;  
 }  
  
 public void setStudent(Student student) {  
 this.student = student;  
 }  
  
 public Course getCourse() {  
 return course;  
 }  
  
 public void setCourse(Course course) {  
 this.course = course;  
 }  
  
 public EnrollmentDescription getDescription() {  
 return description;  
 }  
  
 public void setDescription(EnrollmentDescription description) {  
 this.description = description;  
 }  
  
 public EnrollmentState getState() {  
 return state;  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 return false;  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\Enrollment.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\EnrollmentDescription.java

package eapli.base.Enrollment.Domain;  
  
import javax.persistence.Embeddable;  
  
@Embeddable  
public class EnrollmentDescription {  
 private String value;  
  
 public EnrollmentDescription(String value) {  
 this.value = value;  
 }  
  
 public EnrollmentDescription() {  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\EnrollmentDescription.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\EnrollmentID.java

package eapli.base.Enrollment.Domain;  
  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
  
public class EnrollmentID implements Comparable<EnrollmentID> {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 private Long id;  
  
 @Override  
 public int compareTo(EnrollmentID o) {  
 return 0;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\EnrollmentID.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\EnrollmentState.java

package eapli.base.Enrollment.Domain;  
  
public enum EnrollmentState {  
 ACCEPTED, REJECTED, PENDING;  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Domain\EnrollmentState.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Repository\EnrollmentRepository.java

package eapli.base.Enrollment.Repository;  
  
import eapli.base.Enrollment.Domain.Enrollment;  
import eapli.framework.domain.repositories.DomainRepository;  
  
public interface EnrollmentRepository extends DomainRepository<Long, Enrollment> {  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Enrollment\Repository\EnrollmentRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\aplication\ExtraClassController.java

package eapli.base.ExtraClasse.aplication;  
  
import eapli.base.Classe.aplication.ClasseDTO;  
import eapli.base.Classe.domain.\*;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Day;  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.base.ExtraClasse.domain.ExtraClasse;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Finish\_Time;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Start\_Time;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Title;  
import eapli.base.ExtraClasse.repository.ExtraClasseRepository;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
  
import java.time.LocalTime;  
import java.util.List;  
import java.util.Optional;  
  
public class ExtraClassController {  
  
 private final ExtraClasseRepository repository;  
 private TeacherRepository teacherRepository;  
  
 public ExtraClassController(AuthorizationService authorizationService, ExtraClasseRepository repository, TeacherRepository teacherRepository) {  
 this.repository = repository;  
 this.teacherRepository = teacherRepository;  
 }  
  
  
 public ExtraClasseDTO scheduleExtraClass(ExtraClasse\_Title title, ExtraClasse\_Start\_Time startTime, ExtraClasse\_Finish\_Time finishTime, ExtraClasse\_Day day, Acronym teacherAcronym) throws IntegrityViolationException, ConcurrencyException {  
  
 final ExtraClasse newExtraClasse = new ExtraClasse(title, startTime, finishTime, day, teacherAcronym);  
  
 if (checkClassConflict(newExtraClasse)) {  
 throw new IllegalStateException("There is a scheduling conflict with this class.");  
 } else {  
 return toDTO(this.repository.save(newExtraClasse));  
 }  
 }  
  
 public boolean checkClassConflict(ExtraClasse classeToCheck) {  
 Iterable<ExtraClasse> allExtraClasses = repository.findAll();  
 for (ExtraClasse existingClass : allExtraClasses) {  
 if (existingClass.getTitle().equals(classeToCheck.getTitle())) {  
 // Skip checking against the same class  
 continue;  
 }  
  
 // Only compare classes on the same day  
 if (existingClass.getDay().equals(classeToCheck.getDay())) {  
 // Check for time overlap  
 if (isTimeOverlapUserInput(classeToCheck.getStart\_time(), classeToCheck.getFinish\_time(), existingClass)) {  
 return true; // Conflict found  
 }  
 }  
 }  
  
 return false; // No conflict found  
 }  
  
 public List<Teacher> findTeacherByAcronym(Acronym teacherAcronym){  
 return teacherRepository.findTeacherByAcronym(teacherAcronym);  
 }  
  
 public boolean isTimeOverlapUserInput(ExtraClasse\_Start\_Time userInputStartTime, ExtraClasse\_Finish\_Time userInputEndTime, ExtraClasse existingClass) {  
 // Extract the LocalTime from userInputStartTime and userInputEndTime  
 LocalTime userInputStart = userInputStartTime.getStart\_time();  
 LocalTime userInputEnd = userInputEndTime.getFinish\_time();  
  
 // Convert start and end times of user input to minutes since midnight  
 int userInputStartMinutes = userInputStart.getHour() \* 60 + userInputStart.getMinute();  
 int userInputEndMinutes = userInputEnd.getHour() \* 60 + userInputEnd.getMinute();  
  
 // Extract the LocalTime from existingClass's start and end time  
 LocalTime existingStartTime = existingClass.getStart\_time().getStart\_time();  
 LocalTime existingFinishTime = existingClass.getFinish\_time().getFinish\_time();  
  
 // Convert start and end times of existing class to minutes since midnight  
 int existingStartMinutes = existingStartTime.getHour() \* 60 + existingStartTime.getMinute();  
 int existingEndMinutes = existingFinishTime.getHour() \* 60 + existingFinishTime.getMinute();  
  
 // Check for overlap: does user input class start during existing class?  
 if (userInputStartMinutes >= existingStartMinutes && userInputStartMinutes < existingEndMinutes) {  
 return true; // Overlap found  
 }  
  
 // Check for overlap: does existing class start during user input class?  
 if (existingStartMinutes >= userInputStartMinutes && existingStartMinutes < userInputEndMinutes) {  
 return true; // Overlap found  
 }  
  
 // If neither condition above is met, there is no overlap  
 return false; // No overlap found  
 }  
 public ExtraClasseDTO toDTO(ExtraClasse extraClasse) {  
 return new ExtraClasseDTO(  
 extraClasse.getTitle().getTitle(),  
 extraClasse.getStart\_time().getStart\_time(),  
 extraClasse.getFinish\_time().getFinish\_time(),  
 extraClasse.getDay().getDay(),  
 extraClasse.getAcronym().getValue()  
 );  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\aplication\ExtraClassController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\aplication\ExtraClasseDTO.java

package eapli.base.ExtraClasse.aplication;  
  
import java.time.LocalDate;  
import java.time.LocalTime;  
  
public class ExtraClasseDTO {  
 public String title;  
 public LocalTime startTime;  
 public LocalTime finishTime;  
 public int dayOfWeek;  
 public String teacherAcronym;  
  
 public ExtraClasseDTO( String title, LocalTime startTime, LocalTime finishTime, int dayOfWeek, String teacherAcronym) {  
 this.title = title;  
 this.startTime = startTime;  
 this.finishTime = finishTime;  
 this.dayOfWeek = dayOfWeek;  
 this.teacherAcronym = teacherAcronym;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\aplication\ExtraClasseDTO.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\aplication\ScheduleExtraClasseService.java

package eapli.base.ExtraClasse.aplication;  
  
import eapli.base.ExtraClasse.domain.ExtraClasse;  
import eapli.base.ExtraClasse.repository.ExtraClasseRepository;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.ApplicationService;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
@ApplicationService  
public class ScheduleExtraClasseService {  
 private final AuthorizationService authz;  
 private final ExtraClasseRepository extraClassRepository;  
 private final TeacherRepository teacherRepository;  
  
 public ScheduleExtraClasseService(final AuthorizationService authz, final ExtraClasseRepository extraClassRepository, final TeacherRepository teacherRepository) {  
 this.authz = authz;  
 this.extraClassRepository = extraClassRepository;  
 this.teacherRepository = teacherRepository;  
 }  
  
 /\*\*  
 \* Schedules a new extra class  
 \*  
 \* @param extraClasse The extra class to be scheduled  
 \* @return the scheduled extra class  
 \*/  
 public ExtraClasse scheduleExtraClass(ExtraClasse extraClasse) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
 return extraClassRepository.save(extraClasse);  
 }  
  
 /\*\*  
 \* Returns all teachers  
 \*  
 \* @return all teachers  
 \*/  
 public Iterable<Teacher> teachers() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
 return teacherRepository.findAll();  
 }  
  
 /\*\*  
 \* Find teacher by system user  
 \*  
 \* @param systemUser system user to find the teacher  
 \* @return Teacher instance  
 \*/  
 public Teacher findTeacherBySystemUser(SystemUser systemUser) {  
 return teacherRepository.findTeacherBySystemUser(systemUser);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\aplication\ScheduleExtraClasseService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse.java

package eapli.base.ExtraClasse.domain;  
  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.validations.Preconditions;  
import javax.persistence.\*;  
  
@Entity  
@Inheritance(strategy = InheritanceType.JOINED)  
public class ExtraClasse implements AggregateRoot<ExtraClasse\_Title> {  
  
 @EmbeddedId  
 private ExtraClasse\_Title title;  
 private ExtraClasse\_Start\_Time start\_time;  
 private ExtraClasse\_Finish\_Time finish\_time;  
 private ExtraClasse\_Day day;  
 private Acronym acronym;  
  
  
 public ExtraClasse(ExtraClasse\_Title title, ExtraClasse\_Start\_Time start\_time, ExtraClasse\_Finish\_Time finish\_time, ExtraClasse\_Day day, Acronym acronym) {  
 Preconditions.nonNull(title, "Title should not be null");  
 this.title = title;  
 this.start\_time = start\_time;  
 this.finish\_time = finish\_time;  
 this.day = day;  
 this.acronym = acronym;  
 }  
  
 public ExtraClasse() {  
 }  
  
  
 public ExtraClasse\_Title getTitle() {  
 return this.title;  
 }  
  
 @Override  
 public ExtraClasse\_Title identity() {  
 return this.title;  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 if (!(other instanceof ExtraClasse)) {  
 return false;  
 }  
 return this.title.equals(((ExtraClasse) other).getTitle());  
 }  
  
 @Override  
 public int compareTo(ExtraClasse\_Title other) {  
 return this.title.compareTo(other);  
 }  
  
 public void setTitle(ExtraClasse\_Title title) {  
 this.title = title;  
 }  
  
 public ExtraClasse\_Start\_Time getStart\_time() {  
 return start\_time;  
 }  
  
 public void setStart\_time(ExtraClasse\_Start\_Time start\_time) {  
 this.start\_time = start\_time;  
 }  
  
 public ExtraClasse\_Finish\_Time getFinish\_time() {  
 return finish\_time;  
 }  
  
 public void setFinish\_time(ExtraClasse\_Finish\_Time finish\_time) {  
 this.finish\_time = finish\_time;  
 }  
  
 public ExtraClasse\_Day getDay() {  
 return day;  
 }  
  
 public void setDay(ExtraClasse\_Day day) {  
 this.day = day;  
 }  
  
 public Acronym getAcronym() {  
 return acronym;  
 }  
  
 public void setAcronym(Acronym acronym) {  
 this.acronym = acronym;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Day.java

package eapli.base.ExtraClasse.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
import javax.persistence.Embeddable;  
  
@Embeddable  
public class ExtraClasse\_Day implements ValueObject {  
  
 private int day;  
  
 protected ExtraClasse\_Day() {  
  
 }  
  
 public ExtraClasse\_Day(int day) {  
 Preconditions.ensure(day >= 1 && day <= 7, "Invalid day of week");  
 this.day = day;  
 }  
  
 public static ExtraClasse\_Day from(int day) throws IllegalArgumentException {  
 return new ExtraClasse\_Day(day);  
 }  
  
 public static ExtraClasse\_Day valueOf(int day) throws IllegalArgumentException {  
 return from(day);  
 }  
  
  
  
 @Override  
 public String toString() {  
 return String.format("Day of Week: %d", day);  
 }  
  
 public int getDay() {  
 return day;  
 }  
  
 public void setDay(int day) {  
 this.day = day;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Day.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Finish\_Time.java

package eapli.base.ExtraClasse.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
import javax.persistence.Embeddable;  
import java.time.LocalTime;  
  
@Embeddable  
public class ExtraClasse\_Finish\_Time implements ValueObject {  
 private LocalTime finish\_time;  
  
  
 protected ExtraClasse\_Finish\_Time() {}  
  
   
 public ExtraClasse\_Finish\_Time(LocalTime finish\_time) {  
 Preconditions.ensure(finish\_time != null, "Invalid finish time");  
 this.finish\_time = finish\_time;  
 }  
  
 public static ExtraClasse\_Finish\_Time from(LocalTime finish\_time) throws IllegalArgumentException{  
 return new ExtraClasse\_Finish\_Time(finish\_time);  
 }  
  
 public static ExtraClasse\_Finish\_Time valueOf(LocalTime finish\_time) {  
 return new ExtraClasse\_Finish\_Time(finish\_time);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Finish Time: %s", finish\_time);  
 }  
  
 public LocalTime getFinish\_time() {  
 return finish\_time;  
 }  
  
 public void setFinish\_time(LocalTime finish\_time) {  
 Preconditions.ensure(finish\_time != null, "Invalid finish time");  
 this.finish\_time = finish\_time;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Finish\_Time.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Start\_Time.java

package eapli.base.ExtraClasse.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
import javax.persistence.Embeddable;  
import java.time.LocalTime;  
  
@Embeddable  
public class ExtraClasse\_Start\_Time implements ValueObject {  
 private LocalTime start\_time;  
  
  
 protected ExtraClasse\_Start\_Time() {}  
  
  
 public ExtraClasse\_Start\_Time(LocalTime start\_time) {  
 Preconditions.ensure(start\_time != null, "Invalid start time");  
 this.start\_time = start\_time;  
 }  
  
 public static ExtraClasse\_Start\_Time from(LocalTime start\_time) throws IllegalArgumentException {  
 return new ExtraClasse\_Start\_Time(start\_time);  
 }  
  
 public static ExtraClasse\_Start\_Time valueOf(LocalTime start\_time) {  
 return new ExtraClasse\_Start\_Time(start\_time);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Start Time: %s", start\_time);  
 }  
  
 public LocalTime getStart\_time() {  
 return start\_time;  
 }  
  
 public void setStart\_time(LocalTime start\_time) {  
 Preconditions.ensure(start\_time != null, "Invalid start time");  
 this.start\_time = start\_time;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Start\_Time.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Title.java

package eapli.base.ExtraClasse.domain;  
  
import eapli.base.Classe.domain.Classe\_Title;  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
import java.util.Objects;  
  
@Embeddable  
public class ExtraClasse\_Title implements ValueObject, Comparable<ExtraClasse\_Title> {  
  
 private String title;  
  
 protected ExtraClasse\_Title() { } // needed for JPA  
  
 private ExtraClasse\_Title(String title){  
 this.title = title;  
 }  
  
 public static ExtraClasse\_Title from(String title){  
 Preconditions.ensure(!(title == null || title.isEmpty()), "Title invalido");  
 return new ExtraClasse\_Title(title);  
 }  
  
 public static ExtraClasse\_Title valueOf(String title){  
 return from(title);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Title : %s", title);  
 }  
  
 @Override  
 public int compareTo(ExtraClasse\_Title o) {  
 return this.title.compareTo(o.title);  
 }  
  
  
 public String getTitle() {  
 return title;  
 }  
  
 public void setTitle(String title) {  
 this.title = title;  
 }  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 Classe\_Title other = (Classe\_Title) o;  
 return Objects.equals(title, other.getTitle());  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(title);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\domain\ExtraClasse\_Title.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\repository\ExtraClasseRepository.java

package eapli.base.ExtraClasse.repository;  
  
import eapli.base.Classe.domain.Classe;  
import eapli.base.Classe.domain.Classe\_Title;  
import eapli.base.ExtraClasse.domain.ExtraClasse;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Title;  
import eapli.framework.domain.repositories.DomainRepository;  
  
import java.util.List;  
  
public interface ExtraClasseRepository extends DomainRepository<ExtraClasse\_Title, ExtraClasse> {  
 List<ExtraClasse> findExtraClassesByTeacher(String teacherAcronym);  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\ExtraClasse\repository\ExtraClasseRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Manager\Manager.java

package eapli.base.Manager;  
  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class Manager extends SystemUser {  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Manager\Manager.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\BoardUpdateService.java

/\*package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.Board;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.systemUserManagement.SystemUserRepository;  
  
import java.util.ArrayList;  
import java.util.List;  
  
  
public class BoardUpdateService {  
 private final SharedBoardRepository boardRepository;  
 private SharedBoard sharedBoard;  
  
 public BoardUpdateService(SharedBoardRepository boardRepository) {  
 this.boardRepository = boardRepository;  
 }  
  
 public List<Board> getBoardUpdateHistory(Board boardId) {  
 Board board = boardRepository.findByBoardId(boardId);  
 if (board != null) {  
 return board.;  
 }  
 return new ArrayList<>(); // Retornar uma lista vazia se a board não for encontrada  
 }  
}\*/

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\BoardUpdateService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\CreateBoardPostItController.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class CreateBoardPostItController {  
  
 private final CreateBoardPostItService svc;  
  
 public CreateBoardPostItController(final SharedBoardRepository sharedBoardRepository) {  
 svc = new CreateBoardPostItService(sharedBoardRepository);  
 }  
  
 public boolean getBoardAccessType(SharedBoard sharedBoard, SystemUser systemUser){  
 return svc.getBoardAccessType(sharedBoard, systemUser);  
 }  
  
 public void createPostIt(String content, SystemUser systemUser){  
 svc.createPostIt(content, systemUser);  
 }  
  
 public Iterable<SharedBoard> findBoardsBySystemUser(SystemUser systemUser){  
 return svc.findBoardsBySystemUser(systemUser);  
 }  
  
 public void addPostIt(SharedBoard sharedBoard, int row, int column) throws InterruptedException {  
 svc.addPostIt(sharedBoard, row, column);  
 }  
 public void undoPostIt(){  
 svc.undoPostIt();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\CreateBoardPostItController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\CreateBoardPostItService.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.CreateBoardPostItThread;  
import eapli.base.SharedBoard.domain.PostIt;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class CreateBoardPostItService {  
  
 private final SharedBoardRepository sharedBoardRepository;  
 private PostIt postIt;  
 private PostIt postItBackup;  
  
 public CreateBoardPostItService(final SharedBoardRepository sharedBoardRepository) {  
 this.sharedBoardRepository = sharedBoardRepository;  
 }  
  
 public Iterable<SharedBoard> findBoardsBySystemUser(SystemUser systemUser){  
 return sharedBoardRepository.findBoardsBySystemUser(systemUser);  
 }  
  
 public boolean getBoardAccessType(SharedBoard sharedBoard, SystemUser systemUser){  
 return sharedBoard.getAccessType(systemUser).equals(SharedBoard.AccessType.WRITE);  
 }  
  
 public void createPostIt(String content, SystemUser systemUser){  
 if (postIt != null) {  
 postItBackup = new PostIt(postIt.getContent(), postIt.getAuthor());  
 }  
 this.postIt=new PostIt(content, systemUser);  
 }  
  
  
 public void undoPostIt() {  
 if (postItBackup != null) {  
 postIt = new PostIt(postItBackup.getContent(), postItBackup.getAuthor());  
 postItBackup = null;  
 }  
 }  
 public void addPostIt(SharedBoard sharedBoard, int row, int column) throws InterruptedException {  
 CreateBoardPostItThread createBoardPostItThread = new CreateBoardPostItThread(sharedBoard.getBoard(), row, column, postIt);  
 Thread thread = new Thread(createBoardPostItThread);  
 thread.start();  
 thread.join();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\CreateBoardPostItService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardArchiveController.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.systemUserManagement.SystemUserRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import org.springframework.stereotype.Controller;  
  
@Controller  
public class SharedBoardArchiveController {  
  
  
 private final SharedBoardArchiveService sharedBoardArchiveService;  
  
 public SharedBoardArchiveController(final AuthorizationService authz , final SharedBoardRepository sharedBoardRepository, final SystemUserRepository systemUserRepository) {  
 sharedBoardArchiveService = new SharedBoardArchiveService(authz,sharedBoardRepository, systemUserRepository);  
 }  
  
 public void archiveBoard(Shared\_Board\_Title title, Username currentUser) {  
 sharedBoardArchiveService.archiveBoard(title, currentUser);  
 }  
  
 public Iterable<SharedBoard> findSharedBoardsByOwner(SystemUser owner) {  
 return sharedBoardArchiveService.findSharedBoardsByOwner(owner);  
 }  
  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardArchiveController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardArchiveService.java

package eapli.base.SharedBoard.aplication;  
  
 import eapli.base.SharedBoard.domain.SharedBoard;  
 import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
 import eapli.base.SharedBoard.repository.SharedBoardRepository;  
 import eapli.base.systemUserManagement.SystemUserRepository;  
 import eapli.framework.infrastructure.authz.application.AuthorizationService;  
 import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
 import eapli.framework.infrastructure.authz.domain.model.Username;  
  
 import java.util.List;  
  
 public class SharedBoardArchiveService {  
  
 private final SystemUserRepository userRepository;  
 private final SharedBoardRepository sharedBoardRepository;  
 private final AuthorizationService authz;  
  
 public SharedBoardArchiveService(final AuthorizationService authz, final SharedBoardRepository sharedBoardRepository, final SystemUserRepository systemUserRepository) {  
 this.authz = authz;  
 this.userRepository = systemUserRepository;  
 this.sharedBoardRepository = sharedBoardRepository;  
 }  
  
 public void archiveBoard(Shared\_Board\_Title title, Username currentUser) {  
 Iterable<SharedBoard> boardOptional = sharedBoardRepository.findByBoardId(title);  
 if (!boardOptional.iterator().hasNext()) {  
 throw new IllegalStateException("Board not found");  
 }  
 SharedBoard board = boardOptional.iterator().next();  
 if (!board.getOwner().username().equals(currentUser)) {  
 throw new IllegalStateException("Only the owner can archive the board");  
 }  
 board.archiveBoard();  
 sharedBoardRepository.save(board);  
 }  
  
 public List<SharedBoard> findSharedBoardsByOwner(SystemUser owner) {  
 return (List<SharedBoard>) sharedBoardRepository.findSharedBoardsByOwner(owner);  
 }  
  
 }

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardArchiveService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardController.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.systemUserManagement.SystemUserRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import org.springframework.stereotype.Controller;  
  
@Controller  
public class SharedBoardController {  
  
 private final SharedBoardService sharedBoardService;  
  
 public SharedBoardController(final SharedBoardRepository sharedBoardRepository, final SystemUserRepository systemUserRepository) {  
 sharedBoardService = new SharedBoardService(sharedBoardRepository, systemUserRepository);  
 }  
  
 public void createSharedBoard(SystemUser myUser, String sharedBoardTitle, int numberOfRows, int numberOfColumns){  
 sharedBoardService.createSharedBoard(myUser, sharedBoardTitle, numberOfRows, numberOfColumns);  
 }  
  
 public void shareBoard(String ownerUsername, String userToShareUsername, SharedBoard.AccessType accessType) throws InterruptedException {  
 sharedBoardService.shareBoard(ownerUsername, userToShareUsername,accessType);  
 }  
  
 /\* public Iterable<SharedBoard> listSharedBoards() {  
 return sharedBoardService.listSharedBoards();  
 }  
  
 \*/  
  
  
 public Iterable<SharedBoard> findSharedBoardsByOwner(SystemUser owner) {  
 return sharedBoardService.findSharedBoardsByOwner(owner);  
 }  
  
  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardService.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.\*;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.systemUserManagement.SystemUserRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
import java.util.List;  
import java.util.Optional;  
  
public class SharedBoardService {  
 private final SystemUserRepository userRepository;  
 private final SharedBoardRepository sharedBoardRepository;  
 private SharedBoard sharedBoard;  
  
 public SharedBoardService(final SharedBoardRepository sharedBoardRepository, final SystemUserRepository systemUserRepository) {  
 this.userRepository = systemUserRepository;  
 this.sharedBoardRepository = sharedBoardRepository;  
 }  
  
 public void createSharedBoard(SystemUser myUser, String sharedBoardTitle, int numberOfRows, int numberOfColumns){  
 this.sharedBoard = sharedBoardRepository.save(new SharedBoard(myUser, new Shared\_Board\_Title(sharedBoardTitle), new NumberofRows(numberOfRows), new NumberofColumns(numberOfColumns)));  
 }  
  
 public void shareBoard(String ownerUsername, String userToShareUsername, SharedBoard.AccessType accessType) throws InterruptedException {  
 Iterable<SystemUser> ownerOptional = userRepository.findByUsername(Username.valueOf(ownerUsername));  
 Iterable<SystemUser> userToShareOptional = userRepository.findByUsername(Username.valueOf(userToShareUsername));  
  
 if (!ownerOptional.iterator().hasNext() || !userToShareOptional.iterator().hasNext()) {  
 throw new IllegalStateException("Owner or user to share not found");  
 }  
  
 SystemUser owner = ownerOptional.iterator().next();  
 SystemUser userToShare = userToShareOptional.iterator().next();  
  
 SharedBoardThread sharingThread = new SharedBoardThread(sharedBoard, owner, userToShare, accessType);  
 Thread thread = new Thread(sharingThread);  
 thread.start();  
 thread.join();  
  
 }  
  
 public Iterable<SharedBoard> listSharedBoards() {  
 return sharedBoardRepository.findAll();  
 }  
  
 public Iterable<SharedBoard> findSharedBoardsByOwner(SystemUser owner) {  
 return sharedBoardRepository.findSharedBoardsByOwner(owner);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\SharedBoardService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\UpdateBoardPostItController.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class UpdateBoardPostItController {  
  
 private final UpdateBoardPostItService svc;  
  
 public UpdateBoardPostItController(final SharedBoardRepository sharedBoardRepository) {  
 svc = new UpdateBoardPostItService(sharedBoardRepository);  
 }  
  
 public boolean getBoardAccessType(SharedBoard sharedBoard, SystemUser systemUser){  
 return svc.getBoardAccessType(sharedBoard, systemUser);  
 }  
  
 public Iterable<SharedBoard> findBoardsBySystemUser(SystemUser systemUser){  
 return svc.findBoardsBySystemUser(systemUser);  
 }  
  
 public void createPostIt(String content, SystemUser systemUser){  
 svc.createPostIt(content, systemUser);  
 }  
  
 public void updatePostIt(SharedBoard sharedBoard, int row, int column) throws InterruptedException {  
 svc.updatePostIt(sharedBoard, row, column);  
 }  
  
 public void movePostIt(SharedBoard sharedBoard, int row, int column, int newRow, int newColumn, SystemUser systemUser) throws InterruptedException {  
 svc.movePostIt(sharedBoard, row, column, newRow, newColumn, systemUser);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\UpdateBoardPostItController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\UpdateBoardPostItService.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.\*;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class UpdateBoardPostItService {  
  
 private final SharedBoardRepository sharedBoardRepository;  
 private PostIt postIt;  
  
 public UpdateBoardPostItService(final SharedBoardRepository sharedBoardRepository) {  
 this.sharedBoardRepository = sharedBoardRepository;  
 }  
  
 public Iterable<SharedBoard> findBoardsBySystemUser(SystemUser systemUser){  
 return sharedBoardRepository.findBoardsBySystemUser(systemUser);  
 }  
  
 public boolean getBoardAccessType(SharedBoard sharedBoard, SystemUser systemUser){  
 return sharedBoard.getAccessType(systemUser).equals(SharedBoard.AccessType.WRITE);  
 }  
  
 public void createPostIt(String content, SystemUser systemUser){  
 this.postIt=new PostIt(content, systemUser);  
 }  
  
 public void updatePostIt(SharedBoard sharedBoard, int row, int column) throws InterruptedException {  
 UpdateBoardPostItThread updateBoardPostItThread = new UpdateBoardPostItThread(sharedBoard.getBoard(), row, column, postIt);  
 Thread thread = new Thread(updateBoardPostItThread);  
 thread.start();  
 thread.join();  
 }  
  
 public void movePostIt(SharedBoard sharedBoard, int row, int column, int newRow, int newColumn, SystemUser systemUser) throws InterruptedException {  
 MoveBoardPostItThread moveBoardPostItThread = new MoveBoardPostItThread(sharedBoard.getBoard(), row, column, newRow, newColumn, systemUser);  
 Thread thread = new Thread(moveBoardPostItThread);  
 thread.start();  
 thread.join();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\aplication\UpdateBoardPostItService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\Board.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import javax.persistence.\*;  
  
@Entity  
public class Board implements ValueObject {  
  
 private NumberofColumns numberC;  
 private NumberofRows numberR;  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 private Long id;  
 @Transient  
 private PostIt[][] boardData;  
  
 public Board(NumberofRows numberR, NumberofColumns numberC) {  
 this.numberR = numberR;  
 this.numberC = numberC;  
 initializeBoard(numberR, numberC);  
 }  
  
 protected Board() {  
 }  
  
 public void initializeBoard(NumberofRows numberR, NumberofColumns numberC){  
 this.boardData = new PostIt[numberR.getValue()][numberC.getValue()];  
 for (int i = 0; i < numberR.getValue(); i++) {  
 for (int j = 0; j < numberC.getValue(); j++) {  
 boardData[i][j]=null;  
 }  
 }  
 }  
  
 public void printBoard() {  
 for (int i = 0; i < numberR.getValue(); i++) {  
 for (int j = 0; j < numberC.getValue(); j++) {  
 System.out.print((boardData[i][j] != null ? boardData[i][j].getContent() : "#") + " ");  
 }  
 System.out.println();  
 }  
 }  
  
 public PostIt getCell(int row, int column) {  
 return boardData[row][column];  
 }  
  
 public synchronized void setCell(int row, int column, PostIt value) {  
 boardData[row][column] = value;  
 }  
  
 public synchronized boolean insertCell(int row, int column, PostIt value) {  
 if(getCell(row, column)==null){  
 setCell(row, column, value);  
 return true;  
 }return false;  
 }  
  
 public synchronized boolean updateCell(int row, int column, PostIt value) {  
 if(getCell(row, column)!=null){  
 if(getCell(row,column).getAuthor().equals(value.getAuthor())){  
 setCell(row, column, value);  
 return true;  
 }  
 return false;  
 }else{  
 setCell(row,column,value);  
 return true;  
 }  
 }  
  
 public synchronized int moveCell(int row, int column, int newRow, int newColumn, SystemUser systemUser){  
 if(getCell(row, column)!=null){  
 if(getCell(newRow, newColumn)==null){  
 if(getCell(row,column).getAuthor().equals(systemUser)){  
 setCell(newRow, newColumn, getCell(row,column));  
 setCell(row, column, null);  
 return 0;  
 } return 1;  
 }return 2;  
 } return 3;  
  
 }  
  
 public NumberofColumns getNumberC() {  
 return numberC;  
 }  
  
 public NumberofRows getNumberR() {  
 return numberR;  
 }  
  
 public PostIt[][] getBoardData() {  
 return boardData;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\Board.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\CreateBoardPostItThread.java

package eapli.base.SharedBoard.domain;  
  
public class CreateBoardPostItThread implements Runnable{  
  
 private final Board board;  
 private final int row;  
 private final int column;  
 private final PostIt postIt;  
  
 public CreateBoardPostItThread(Board board, int row, int column, PostIt postIt){  
 this.board=board;  
 this.row=row;  
 this.column=column;  
 this.postIt=postIt;  
 }  
  
 public void run(){  
// while (!Thread.interrupted()) {  
 if(board.insertCell(row, column, postIt)){  
 System.out.println("Post-it successfully added to the board.");  
 System.out.println("Updated Board: ");  
 board.printBoard();  
 }else{  
 System.out.println("The desired position is already occupied.");  
 }  
// }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\CreateBoardPostItThread.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\MoveBoardPostItThread.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class MoveBoardPostItThread implements Runnable{  
  
 private final Board board;  
 private final int row;  
 private final int column;  
 private final int newRow;  
 private final int newColumn;  
 private final SystemUser systemUser;  
  
 public MoveBoardPostItThread(Board board, int row, int column, int newRow, int newColumn, SystemUser systemUser){  
 this.board=board;  
 this.row=row;  
 this.column=column;  
 this.newRow=newRow;  
 this.newColumn=newColumn;  
 this.systemUser=systemUser;  
 }  
  
 public void run(){  
// while (!Thread.interrupted()) {  
 if(board.moveCell(row, column, newRow, newColumn, systemUser)==0){  
 System.out.println("Post-it successfully updated in the board.");  
 }else if(board.moveCell(row, column, newRow, newColumn, systemUser)==1){  
 System.out.println("You are not the author of the original post-it.");  
 }else if(board.moveCell(row, column, newRow, newColumn, systemUser)==2){  
 System.out.println("The desired position is already occupied.");  
 }else if(board.moveCell(row, column, newRow, newColumn, systemUser)==3) {  
 System.out.println("The original position is not occupied with a post-it.");  
 }  
 System.out.println("Updated Board: ");  
 board.printBoard();  
// }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\MoveBoardPostItThread.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\NumberofColumns.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
  
@Embeddable  
public class NumberofColumns implements ValueObject {  
 private int num\_col;  
  
 public NumberofColumns(int value){  
 this.num\_col=value;  
 }  
  
 protected NumberofColumns(){}  
  
 public static NumberofColumns from(int value) throws IllegalArgumentException {  
 Preconditions.ensure(value > 0, "Number of columns must be greater than 0");  
 return new NumberofColumns(value);  
 }  
  
 public static NumberofColumns valueOf(int value) throws IllegalArgumentException {  
 return from(value);  
 }  
  
 public int getValue(){  
 return num\_col;  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Number of Columns: %d", num\_col);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\NumberofColumns.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\NumberofRows.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
  
@Embeddable  
public class NumberofRows implements ValueObject {  
  
 private int num\_row;  
  
 public NumberofRows(int value){  
 this.num\_row=value;  
 }  
  
 protected NumberofRows(){}  
  
 public static NumberofRows from(int numberOfRows) throws IllegalArgumentException {  
 Preconditions.ensure(numberOfRows > 0, "Number of rows must be greater than zero");  
 return new NumberofRows(numberOfRows);  
 }  
  
 public static NumberofRows valueOf(int numberOfRows) throws IllegalArgumentException {  
 return from(numberOfRows);  
 }  
  
 public int getValue(){  
 return num\_row;  
 }  
  
 @Override  
 public String toString() {  
 return "Number of Rows: " + num\_row;  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\NumberofRows.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\PostIt.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import javax.persistence.Entity;  
  
@Entity  
public class PostIt {  
  
 private String content;  
  
 private SystemUser author;  
  
 protected PostIt(){}  
  
 public PostIt(String content, SystemUser systemUser){  
 this.content=content;  
 this.author=systemUser;  
 }  
  
 public String getContent(){return content;}  
  
 public SystemUser getAuthor(){return author;}  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\PostIt.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\SharedBoard.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.representations.dto.DTOable;  
import eapli.framework.representations.dto.GeneralDTO;  
import eapli.framework.validations.Preconditions;  
import eapli.framework.visitor.Visitable;  
import eapli.framework.visitor.Visitor;  
  
import javax.persistence.\*;  
import java.io.Serializable;  
import java.util.HashMap;  
import java.util.Map;  
  
@Entity  
@Table(name = "SharedBoard")  
public class SharedBoard implements AggregateRoot<Shared\_Board\_Title>, DTOable<GeneralDTO>, Visitable<GeneralDTO>, Serializable {  
  
 @ManyToOne  
 @JoinColumn(name = "owner\_id")  
 private SystemUser owner;  
  
 @ElementCollection // Isto indica que sharedUsers é uma coleção de instâncias de um tipo básico (neste caso, AccessType).  
 @CollectionTable(name="UserAccess", joinColumns=@JoinColumn(name="sharedboard\_id")) // Isto indica que a tabela para a coleção sharedUsers é UserAccess e ela está ligada à tabela SharedBoard através da coluna sharedboard\_id.  
 @MapKeyJoinColumn(name="systemuser\_id") // Isto indica que a chave do mapa em sharedUsers está mapeada para a coluna systemuser\_id.  
 @Column(name="access\_type") // // Isto indica que o valor de cada instância na coleção sharedUsers será armazenado na coluna access\_type  
 @Enumerated(EnumType.STRING) // Isto indica que o valor em sharedUsers será persistido como uma sting  
 private Map<SystemUser, AccessType> sharedUsers;  
  
 @EmbeddedId  
 private Shared\_Board\_Title title;  
  
 @JoinColumn(name = "board\_id")  
 private Board board;  
  
 private boolean isArchived = false;  
  
 public enum AccessType {  
 READ, WRITE  
 }  
  
 public SharedBoard(SystemUser owner, Shared\_Board\_Title title, NumberofRows numberofRows, NumberofColumns numberofColumns) {  
 Preconditions.ensure(owner != null, "Owner cannot be null");  
 Preconditions.ensure(title != null, "Title cannot be null");  
 this.board = new Board(numberofRows, numberofColumns);  
 this.title = title;  
 this.owner = owner;  
 this.sharedUsers = new HashMap<>();  
 }  
  
 public SharedBoard() {  
 }  
  
 public void shareBoard(SystemUser currentUser, SystemUser newUser, AccessType accessType) {  
 Preconditions.ensure(currentUser != null, "Current user cannot be null");  
 Preconditions.ensure(newUser != null, "User to share the board with cannot be null");  
  
 if (!this.owner.equals(currentUser)) {  
 throw new IllegalStateException("Only the owner can share the board");  
 }  
 if (!this.sharedUsers.containsKey(newUser)) {  
 this.sharedUsers.put(newUser, accessType);  
 }  
 }  
  
 public SystemUser getOwner() {  
 return owner;  
 }  
  
 public void archiveBoard() {  
 this.isArchived = true;  
  
 // Se o board ta arquivado as permissoes de todos os users vao para READ e só o Owner é que tem write  
 for (Map.Entry<SystemUser, AccessType> entry : sharedUsers.entrySet()) {  
 if (!entry.getKey().equals(this.owner)) {  
 entry.setValue(AccessType.READ);  
 }  
 }  
 }  
  
  
 public boolean isArchived() {  
 return isArchived;  
 }  
  
 public Shared\_Board\_Title getTitle() {  
 return title;  
 }  
  
 @Override  
 public Shared\_Board\_Title identity() {  
 return title;  
 }  
  
 public Board getBoard() {  
 return board;  
 }  
  
 public AccessType getAccessType(SystemUser user) {  
 return this.sharedUsers.get(user);  
 }  
  
 public Map<SystemUser, AccessType> getSharedUsers(){  
 return sharedUsers;  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 if (!(other instanceof SharedBoard)) {  
 return false;  
 }  
 SharedBoard otherBoard = (SharedBoard) other;  
 return this.identity().equals(otherBoard.identity());  
 }  
  
 @Override  
 public GeneralDTO toDTO() {  
 final GeneralDTO ret = new GeneralDTO("shared board");  
 ret.put("title", title.toString());  
 ret.put("owner", owner.username().toString());  
  
 return ret;  
 }  
  
 @Override  
 public void accept(Visitor<GeneralDTO> visitor) {  
 visitor.visit(toDTO());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\SharedBoard.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\SharedBoardThread.java

package eapli.base.SharedBoard.domain;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class SharedBoardThread implements Runnable {  
 private SharedBoard sharedBoard;  
 private SystemUser owner;  
 private SystemUser userToShare;  
 private SharedBoard.AccessType accessType;  
  
 public SharedBoardThread(SharedBoard sharedBoard, SystemUser owner, SystemUser userToShare, SharedBoard.AccessType accessType) {  
 this.sharedBoard = sharedBoard;  
 this.owner = owner;  
 this.userToShare = userToShare;  
 this.accessType = accessType;  
 }  
  
 @Override  
 public void run() {  
 sharedBoard.shareBoard(owner, userToShare, accessType);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\SharedBoardThread.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\Shared\_Board\_Title.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
import java.util.Objects;  
  
@Embeddable  
public class Shared\_Board\_Title implements ValueObject, Comparable<Shared\_Board\_Title> {  
  
 private String title;  
  
 protected Shared\_Board\_Title() {  
 }  
  
 public Shared\_Board\_Title(String title) throws IllegalArgumentException {  
 Preconditions.ensure(title != null && !title.isEmpty(), "Invalid title");  
 this.title = title;  
 }  
  
 public static Shared\_Board\_Title valueOf(String title) throws IllegalArgumentException {  
 return new Shared\_Board\_Title(title);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Title: %s", title);  
 }  
  
 public String getTitle() {  
 return this.title;  
 }  
  
 public void setTitle(String title) throws IllegalArgumentException {  
 Preconditions.ensure(title != null && !title.isEmpty(), "Invalid title");  
 this.title = title;  
 }  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
  
 Shared\_Board\_Title other = (Shared\_Board\_Title) o;  
 return Objects.equals(title, other.title);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(title);  
 }  
  
  
 @Override  
 public int compareTo(Shared\_Board\_Title o) {  
 return this.title.compareTo(o.title);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\Shared\_Board\_Title.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\UpdateBoardPostItThread.java

package eapli.base.SharedBoard.domain;  
  
public class UpdateBoardPostItThread implements Runnable{  
  
 private final Board board;  
 private final int row;  
 private final int column;  
 private final PostIt postIt;  
  
 public UpdateBoardPostItThread(Board board, int row, int column, PostIt postIt){  
 this.board=board;  
 this.row=row;  
 this.column=column;  
 this.postIt=postIt;  
 }  
  
 public void run(){  
// while (!Thread.interrupted()) {  
 if(board.updateCell(row, column, postIt)){  
 System.out.println("Post-it successfully updated in the board.");  
 System.out.println("Updated Board: ");  
 board.printBoard();  
 }else{  
 System.out.println("You are not the author of the post-it in the desired position.");  
 }  
// }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\UpdateBoardPostItThread.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\sharedBoardHistory.java

package eapli.base.SharedBoard.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
import lombok.\*;  
  
import javax.persistence.Embeddable;  
@Embeddable  
@EqualsAndHashCode  
@AllArgsConstructor(access = AccessLevel.PROTECTED)  
@NoArgsConstructor(access = AccessLevel.PROTECTED)  
@Getter  
public class sharedBoardHistory {  
  
 private String history;  
  
 public static sharedBoardHistory from(String history) throws Exception {  
 Preconditions.ensure(history != null && !history.isEmpty(), "Invalid history");  
 return new sharedBoardHistory(history);  
 }  
  
 public static sharedBoardHistory valueOf(String history) throws Exception {  
 return from(history);  
 }  
  
 @Override  
 public String toString() {  
 return String.format("History: %s", history);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\domain\sharedBoardHistory.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\repository\SharedBoardRepository.java

package eapli.base.SharedBoard.repository;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.framework.domain.repositories.DomainRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public interface SharedBoardRepository extends DomainRepository<Shared\_Board\_Title, SharedBoard> {  
  
 Iterable<SharedBoard> findSharedBoardsByOwner(SystemUser owner);  
  
 Iterable<SharedBoard> findByBoardId(Shared\_Board\_Title title);  
  
 Iterable<SharedBoard> findBoardsBySystemUser(SystemUser systemUser);  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\SharedBoard\repository\SharedBoardRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Date\_Of\_Birth.java

package eapli.base.Student\_Teacher;  
  
import lombok.\*;  
  
import javax.persistence.Embeddable;  
import java.time.LocalDate;  
import java.time.format.DateTimeFormatter;  
import java.util.Objects;  
@Embeddable  
@NoArgsConstructor(access = AccessLevel.PROTECTED)  
@Getter  
@EqualsAndHashCode  
public class Date\_Of\_Birth  
{  
 private LocalDate dateOfBirth;  
  
 public Date\_Of\_Birth(LocalDate dateOfBirth) {  
 this.dateOfBirth=dateOfBirth;  
 }  
  
 public static Date\_Of\_Birth valueOf(LocalDate dateOfBirth) {  
 return new Date\_Of\_Birth(dateOfBirth);  
 }  
  
 @Override  
 public String toString() {  
 return dateOfBirth.format(DateTimeFormatter.ofPattern("dd/MM/yyyy"));  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof Date\_Of\_Birth)) return false;  
 Date\_Of\_Birth that = (Date\_Of\_Birth) o;  
 return dateOfBirth.equals(that.dateOfBirth);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(dateOfBirth);  
 }  
  
 public LocalDate getDateOfBirth() {  
 return dateOfBirth;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Date\_Of\_Birth.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Student\Repository\StudentRepository.java

package eapli.base.Student\_Teacher.Student.Repository;  
  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.framework.domain.repositories.DomainRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import org.springframework.stereotype.Repository;  
  
import java.util.HashMap;  
import java.util.Map;  
import java.util.Optional;  
  
@Repository  
public interface StudentRepository extends DomainRepository<MechanographicNumber, Student> {  
 Map<MechanographicNumber, Student> students = new HashMap<>();  
  
 Iterable <Student> findStudentByMechanographicNumberReturnList(String mechanographicNumber);  
 Student findStudentByMechanographicNumber(String mechanographicNumber);  
  
 Student findStudentBySystemUser(SystemUser systemUser);  
  
 default Student findStudentByMechanographicNumber(MechanographicNumber mechanographicNumber) {  
 return students.get(mechanographicNumber);  
 }  
  
 default void addStudent(Student student) {  
 students.put(student.getMechanographicNumber(), student);  
 }  
  
 default void removeStudent(Student student){  
 students.remove(student);  
 }  
  
 default void removeStudentByMechanographicNum(MechanographicNumber mechanographicNumber){  
 students.remove(mechanographicNumber);  
 }  
  
 default Student getStudentByMechanographicNum(MechanographicNumber mechanographicNumber){  
 return students.get(mechanographicNumber);  
 }  
  
 Optional<Student> findByUsername(Username name);  
  
 Optional<Student> findByMechanographicNumber(MechanographicNumber number);  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Student\Repository\StudentRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Student\domain\MechanographicNumber.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.Student\_Teacher.Student.domain;  
  
import javax.persistence.Embeddable;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.strings.util.StringPredicates;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt  
 \*/  
@Embeddable  
public class MechanographicNumber implements ValueObject, Comparable<MechanographicNumber> {  
  
 private static final long serialVersionUID = 1L;  
  
 private String number;  
  
 public MechanographicNumber(final String mechanographicNumber) {  
 if (StringPredicates.isNullOrEmpty(mechanographicNumber)) {  
 throw new IllegalArgumentException(  
 "Mechanographic Number should neither be null nor empty");  
 }  
 this.number = mechanographicNumber;  
 }  
  
 protected MechanographicNumber() {  
 // for ORM  
 }  
  
 public static MechanographicNumber valueOf(final String mechanographicNumber) {  
 if (!isValidmechanographicNumber(mechanographicNumber)) {  
 throw new IllegalArgumentException("Invalid mechanographic number format");  
 }  
 return new MechanographicNumber(mechanographicNumber);  
 }  
  
 private static boolean isValidmechanographicNumber(String mechanographicNumber) {  
 final int lowestYearAcceptable = 2000;  
 final int mechanographicNumberLength = 9;  
 int year = Integer.parseInt(mechanographicNumber.substring(0, 4));  
  
 //valida o length, (9) e valida o ano minimo (2000)  
 return (year >= lowestYearAcceptable) && mechanographicNumber.length() == mechanographicNumberLength;  
 }  
  
 @Override  
 public boolean equals(final Object o) {  
 if (this == o) {  
 return true;  
 }  
 if (!(o instanceof MechanographicNumber)) {  
 return false;  
 }  
  
 final MechanographicNumber that = (MechanographicNumber) o;  
 return this.number.equals(that.number);  
 }  
  
 @Override  
 public int hashCode() {  
 return this.number.hashCode();  
 }  
  
 @Override  
 public String toString() {  
 return this.number;  
 }  
  
 @Override  
 public int compareTo(final MechanographicNumber arg0) {  
 return number.compareTo(arg0.number);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Student\domain\MechanographicNumber.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Student\domain\Student.java

package eapli.base.Student\_Teacher.Student.domain;  
  
  
import eapli.base.Student\_Teacher.Date\_Of\_Birth;  
import eapli.base.Student\_Teacher.Tax\_Payer\_Number;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.domain.model.DomainEntities;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import javax.persistence.\*;  
  
  
@Entity  
public class Student implements AggregateRoot<MechanographicNumber> {  
  
 @Id  
 private MechanographicNumber mechanographicNumber;  
  
 @OneToOne  
 private SystemUser systemUser;  
  
 @Embedded  
 @AttributeOverride(name = "number", column = @Column(name = "taxPayerNumber"))  
 private Tax\_Payer\_Number taxPayerNumber;  
  
 @Embedded  
 private Date\_Of\_Birth dateOfBirth;  
  
  
 public Student(){}  
  
 public Student(SystemUser systemUser, MechanographicNumber mechanographicNumber, Tax\_Payer\_Number tax\_payer\_number, Date\_Of\_Birth dateOfBirth){  
 this.systemUser=systemUser;  
 this.mechanographicNumber = mechanographicNumber;  
 this.taxPayerNumber=tax\_payer\_number;  
 this.dateOfBirth=dateOfBirth;  
 }  
  
 @Override  
 public String toString() {  
 return "Student{" +  
 "mechanographicNumber=" + mechanographicNumber +  
 ", taxPayerNumber=" + taxPayerNumber +  
 ", dateOfBirth=" + dateOfBirth +  
 '}';  
 }  
  
 public MechanographicNumber getMechanographicNumber() {  
 return this.mechanographicNumber;  
 }  
  
 public Tax\_Payer\_Number getTaxPayerNumber() {  
 return this.taxPayerNumber;  
 }  
  
  
 @Override  
 public boolean sameAs(Object other) {  
 Student student = (Student) other;  
 return this.equals(student) && getTaxPayerNumber().equals(student.getTaxPayerNumber()) && student.getDateOfBirth().equals(student.getDateOfBirth());  
 }  
  
 private Date\_Of\_Birth getDateOfBirth() {  
 return this.dateOfBirth;  
 }  
  
 @Override  
 public boolean equals(final Object o) {  
 return DomainEntities.areEqual(this, o);  
 }  
  
 @Override  
 public MechanographicNumber identity() {  
 return mechanographicNumber;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Student\domain\Student.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Tax\_Payer\_Number.java

package eapli.base.Student\_Teacher;  
  
import lombok.\*;  
  
import javax.persistence.Embeddable;  
  
@Embeddable  
@NoArgsConstructor(access = AccessLevel.PROTECTED)  
@Getter  
@EqualsAndHashCode  
public class Tax\_Payer\_Number  
{  
 private String number;  
  
 public Tax\_Payer\_Number(String number) {  
 this.number=number;  
 }  
  
 public static Tax\_Payer\_Number valueOf(String number) {  
 // Validate the taxpayer number format and throw an IllegalArgumentException if invalid  
 if (!isValidTaxPayerNumber(number)) {  
 throw new IllegalArgumentException("Invalid tax payer number format");  
 }  
 return new Tax\_Payer\_Number(number);  
 }  
  
 private static boolean isValidTaxPayerNumber(String number) {  
 // Validate the taxpayer number format  
 // For example, the taxpayer number may need to have a certain length and format, or satisfy some other criteria  
 // Return true if the taxpayer number is valid, and false otherwise  
 return true; // Replace with actual validation logic  
 }  
  
 @Override  
 public String toString() {  
 return number;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Tax\_Payer\_Number.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Teacher\Domain\Acronym.java

package eapli.base.Student\_Teacher.Teacher.Domain;  
  
import javax.persistence.Embeddable;  
import java.util.Objects;  
  
@Embeddable  
public class Acronym {  
 private String value;  
  
 protected Acronym() {  
 }  
  
 public Acronym(String value) {  
 this.value = value.toUpperCase();  
 }  
  
 public static Acronym valueOf(String value) {  
 return new Acronym(value);  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 Acronym acronym = (Acronym) o;  
 return value.equals(acronym.value);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(value);  
 }  
  
 @Override  
 public String toString() {  
 return value;  
 }  
  
 public String getValue() {  
 return value;  
 }  
  
 public void setValue(String value) {  
 this.value = value.toUpperCase();  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Teacher\Domain\Acronym.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Teacher\Domain\Teacher.java

package eapli.base.Student\_Teacher.Teacher.Domain;  
  
import eapli.base.Student\_Teacher.Date\_Of\_Birth;  
import eapli.base.Student\_Teacher.Tax\_Payer\_Number;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import javax.persistence.\*;  
import java.io.Serializable;  
import java.util.Objects;  
  
@Entity  
public class Teacher implements AggregateRoot<Long>, Serializable  
{  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 private Long id;  
 @Column(unique = true)  
 private Acronym acronym;  
  
 @OneToOne  
 private SystemUser systemUser;  
 private Tax\_Payer\_Number tax\_payer\_number;  
 private Date\_Of\_Birth Date\_of\_Birth;  
 //private SystemUser systemUser;  
 // O id pode ser o acronym  
  
 public Teacher(){}  
  
 public Teacher(SystemUser systemUser, Acronym acronym, Tax\_Payer\_Number tax\_payer\_number, Date\_Of\_Birth date){  
 this.systemUser=systemUser;  
 this.acronym=acronym;  
 this.tax\_payer\_number=tax\_payer\_number;  
 this.Date\_of\_Birth=date;  
  
 }  
  
 public Acronym getAcronym() {  
 return this.acronym;  
 }  
 @Override  
 public boolean sameAs(Object other)  
 {  
 return Objects.equals(this.getId(), ((Teacher) other).getId());  
 }  
  
 @Override  
 public String toString(){return acronym.toString();}  
  
 public Long getId() {  
 return id;  
 }  
  
 public Tax\_Payer\_Number getTax\_payer\_number() {  
 return tax\_payer\_number;  
 }  
  
 public SystemUser getSystemUser() {  
 return systemUser;  
 }  
  
 public Date\_Of\_Birth getDate\_of\_Birth() {  
 return Date\_of\_Birth;  
 }  
  
 @Override  
 public Long identity() {  
 return id;  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Teacher\Domain\Teacher.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Teacher\Repository\TeacherRepository.java

package eapli.base.Student\_Teacher.Teacher.Repository;  
  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.framework.domain.repositories.DomainRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
import java.util.List;  
import java.util.Optional;  
  
public interface TeacherRepository extends DomainRepository<Long, Teacher> {  
 List<Teacher> findTeacherByAcronym(Acronym acronym);  
  
 Optional<Teacher> ofIdentity(Username id);  
  
 Teacher findTeacherBySystemUser(SystemUser systemUser);  
  
 void deleteOfIdentity(Username entityId);  
}  
  
 /\*  
 public static Optional<Teacher> findTeacherByAcronym(Acronym acronym) {  
  
 for (Teacher teacher : teachers) {  
 if (teacher.getAcronym().equals(acronym)) {  
 return Optional.of(teacher);  
 }  
 }  
 return Optional.empty();  
 }  
  
  
  
 public default Optional<Teacher> findById(Long id) {  
 for (Teacher teacher : teachers) {  
 if (teacher.getId().equals(id)) {  
 return Optional.of(teacher);  
 }  
 }  
 return Optional.empty();  
 }  
  
 \*/

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\Student\_Teacher\Teacher\Repository\TeacherRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupFactory.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application;  
  
/\*\*  
 \* A simple factory to obtain the desired implementation of the  
 \* {@link AcceptRefuseSignupController}.  
 \*  
 \* @author Paulo Gandra de Sousa 16/05/2019  
 \*  
 \*/  
public final class AcceptRefuseSignupFactory {  
 private AcceptRefuseSignupFactory() {  
 // ensure utility  
 }  
  
 public static AcceptRefuseSignupRequestController build() {  
 // decide and try  
  
 // return new AcceptRefuseSignupRequestControllerTxImpl();  
 return new AcceptRefuseSignupRequestControllerEventfullImpl();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupFactory.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupRequestController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application;  
  
import eapli.base.clientusermanagement.domain.SignupRequest;  
  
/\*\*  
 \* The controller for the use case "Accept or refuse signup request".  
 \* <p>  
 \* <p/>  
 \* Approving a signup request needs to create/change more than one aggregate  
 \* instance, as such it violates the rule that one controller should only  
 \* create/change one aggregate instance.  
 \* <p>  
 \* <p/>  
 \* We provide two implementations of this controller for comparison of different  
 \* approaches. The (traditional) approach  
 \* {@link AcceptRefuseSignupRequestControllerTxImpl} creates a wrapping  
 \* transaction and performs the needed steps inside that transaction. It makes  
 \* it simple to develop and understand but hardcodes the process and fails the  
 \* Single Responsibility Principle (SRP) as now this class will have more than  
 \* one reason to change.  
 \* <p>  
 \* <p/>  
 \* The other approach uses events  
 \* {@link AcceptRefuseSignupRequestControllerEventfullImpl} and decouples the  
 \* different steps of the process. This makes the system more flexible if there  
 \* is the need to change the actual process, enforces the SRP, but increases the  
 \* complexity, e.g., there is the need to have an event bus, there is no global  
 \* status, if something fails in the middle of the process it is not easy to  
 \* compensate (rollback) and there is no temporal coupling so the system needs  
 \* to handle "waiting" for something to happen before proceeding (see  
 \* {@link AddClientUserOnSignupAcceptedController})  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*/  
public interface AcceptRefuseSignupRequestController {  
  
 SignupRequest acceptSignupRequest(SignupRequest theSignupRequest);  
  
 SignupRequest refuseSignupRequest(SignupRequest theSignupRequest);  
  
 Iterable<SignupRequest> listPendingSignupRequests();  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupRequestController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupRequestControllerEventfullImpl.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application;  
  
import org.springframework.transaction.annotation.Transactional;  
  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.base.clientusermanagement.domain.events.SignupAcceptedEvent;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.domain.events.DomainEvent;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.pubsub.EventPublisher;  
import eapli.framework.infrastructure.pubsub.impl.inprocess.service.InProcessPubSub;  
import eapli.framework.validations.Preconditions;  
  
/\*\*  
 \* the controller for the use case "Accept or refuse signup request"  
 \*  
 \* this implementation makes use of domain events to (1) follow the rule that  
 \* one controller should only modify one aggregate, and (2) notify other parts  
 \* of the system to react accordingly. For an alternative transactional approach  
 \* see {@link AcceptRefuseSignupRequestControllerTxImpl}  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*/  
@UseCaseController  
public class AcceptRefuseSignupRequestControllerEventfullImpl  
 implements AcceptRefuseSignupRequestController{  
  
 private final SignupRequestRepository signupRequestsRepository = PersistenceContext  
 .repositories().signupRequests();  
 private final AuthorizationService authorizationService = AuthzRegistry.authorizationService();  
 private final EventPublisher dispatcher = InProcessPubSub.publisher();  
  
 @Override  
 @SuppressWarnings("squid:S1226")  
 public SignupRequest acceptSignupRequest(SignupRequest theSignupRequest) {  
 authorizationService.ensureAuthenticatedUserHasAnyOf(BaseRoles.MANAGER,  
 BaseRoles.ADMIN);  
  
 Preconditions.nonNull(theSignupRequest);  
  
 theSignupRequest = markSignupRequestAsAccepted(theSignupRequest);  
 return theSignupRequest;  
 }  
  
 /\*\*  
 \* modify Signup Request to accepted  
 \*  
 \* @param theSignupRequest  
 \* @return  
 \* @throws ConcurrencyException  
 \* @throws IntegrityViolationException  
 \*/  
 @SuppressWarnings("squid:S1226")  
 private SignupRequest markSignupRequestAsAccepted(SignupRequest theSignupRequest) {  
 // do just what is needed in the scope of this use case  
 theSignupRequest.accept();  
 theSignupRequest = signupRequestsRepository.save(theSignupRequest);  
  
 // notify interested parties (if any)  
 final DomainEvent event = new SignupAcceptedEvent(theSignupRequest);  
 dispatcher.publish(event);  
  
 return theSignupRequest;  
 }  
  
 @Override  
 @Transactional  
 public SignupRequest refuseSignupRequest(final SignupRequest theSignupRequest) {  
 authorizationService.ensureAuthenticatedUserHasAnyOf(BaseRoles.MANAGER,  
 BaseRoles.ADMIN);  
  
 Preconditions.nonNull(theSignupRequest);  
  
 theSignupRequest.refuse();  
 return signupRequestsRepository.save(theSignupRequest);  
 }  
  
 /\*\*  
 \*  
 \* @return  
 \*/  
 @Override  
 public Iterable<SignupRequest> listPendingSignupRequests() {  
 return signupRequestsRepository.pendingSignupRequests();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupRequestControllerEventfullImpl.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupRequestControllerTxImpl.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application;  
  
import java.util.HashSet;  
import java.util.Set;  
  
import eapli.base.clientusermanagement.domain.ClientUserBuilder;  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserManagementService;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
/\*\*  
 \* The transactional controller for the use case "accept/refuse a signup  
 \* request".  
 \* <p>  
 \* following the guideline that a controller should only change one Aggregate,  
 \* we shouldn't be changing all these entities here, but should instead use  
 \* asynchronous events. However in this case we will take advantage of  
 \* TransactionalContext  
 \*  
 \* @todo handle the scenario where in the meantime the username is already used  
 \* by some other user  
 \*  
 \* @author AJS on 08/04/2016.  
 \*/  
@UseCaseController  
public class AcceptRefuseSignupRequestControllerTxImpl  
 implements AcceptRefuseSignupRequestController {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final UserManagementService userService = AuthzRegistry.userService();  
  
 private final TransactionalContext txCtx = PersistenceContext.repositories()  
 .newTransactionalContext();  
 private final ClientUserRepository clientUserRepository = PersistenceContext  
 .repositories().clientUsers(txCtx);  
 private final SignupRequestRepository signupRequestsRepository = PersistenceContext  
 .repositories().signupRequests(txCtx);  
  
 /\*  
 \* (non-Javadoc)  
 \*  
 \* @see eapli.base.clientusermanagement.application.  
 \* AcceptRefuseSignupRequestController#acceptSignupRequest(eapli.base.  
 \* clientusermanagement.domain.SignupRequest)  
 \*/  
 @Override  
 public SignupRequest acceptSignupRequest(SignupRequest theSignupRequest) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.MANAGER, BaseRoles.ADMIN);  
  
 if (theSignupRequest == null) {  
 throw new IllegalArgumentException();  
 }  
  
 // explicitly begin a transaction  
 txCtx.beginTransaction();  
  
 final SystemUser newUser = createSystemUserForClientUser(theSignupRequest);  
 createClientUser(theSignupRequest, newUser);  
 theSignupRequest = acceptTheSignupRequest(theSignupRequest);  
  
 // explicitly commit the transaction  
 txCtx.commit();  
  
 return theSignupRequest;  
 }  
  
 private SignupRequest acceptTheSignupRequest(final SignupRequest theSignupRequest) {  
 theSignupRequest.accept();  
 return this.signupRequestsRepository.save(theSignupRequest);  
 }  
  
 private void createClientUser(final SignupRequest theSignupRequest,  
 final SystemUser newUser) {  
 final ClientUserBuilder clientUserBuilder = new ClientUserBuilder();  
 clientUserBuilder.withMecanographicNumber(theSignupRequest.mecanographicNumber())  
 .withSystemUser(newUser);  
 this.clientUserRepository.save(clientUserBuilder.build());  
 }  
  
 //  
 // add system user  
 //  
 private SystemUser createSystemUserForClientUser(final SignupRequest theSignupRequest) {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.CLIENT\_USER);  
 return userService.registerUser(theSignupRequest.username(), theSignupRequest.password(),  
 theSignupRequest.name(), theSignupRequest.email(), roles);  
 }  
  
 /\*  
 \* (non-Javadoc)  
 \*  
 \* @see eapli.base.clientusermanagement.application.  
 \* AcceptRefuseSignupRequestController#refuseSignupRequest(eapli.base.  
 \* clientusermanagement.domain.SignupRequest)  
 \*/  
 @Override  
 public SignupRequest refuseSignupRequest(SignupRequest theSignupRequest) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.MANAGER, BaseRoles.ADMIN);  
  
 if (theSignupRequest == null) {  
 throw new IllegalArgumentException();  
 }  
  
 // explicitly begin a transaction  
 txCtx.beginTransaction();  
  
 theSignupRequest.refuse();  
 theSignupRequest = signupRequestsRepository.save(theSignupRequest);  
  
 // explicitly commit the transaction  
 txCtx.commit();  
  
 return theSignupRequest;  
 }  
  
 /\*  
 \* (non-Javadoc)  
 \*  
 \* @see eapli.base.clientusermanagement.application.  
 \* AcceptRefuseSignupRequestController#listPendingSignupRequests()  
 \*/  
 @Override  
 public Iterable<SignupRequest> listPendingSignupRequests() {  
 return signupRequestsRepository.pendingSignupRequests();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\AcceptRefuseSignupRequestControllerTxImpl.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\ClientUserService.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.clientusermanagement.domain.ClientUser;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
import java.util.Optional;  
  
/\*\*  
 \* @author mcn  
 \*  
 \* Classe destinada a ser usada pelo sistema de reporting para obter informação  
 \*/  
public class ClientUserService {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final ClientUserRepository repo = PersistenceContext.repositories().clientUsers();  
  
 // Código destinado a procurar um utilizador pelo seu número mecanográfico.  
 public Optional<ClientUser> findClientUserByMecNumber(  
 final String mecNumber) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.ADMIN, BaseRoles.MANAGER);  
 return repo.ofIdentity(MechanographicNumber.valueOf(mecNumber));  
 }  
  
 // Código destinado procurar um utilizador pelo seu username.  
 public Optional<ClientUser> findClientUserByUsername(final Username user) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.ADMIN, BaseRoles.MANAGER);  
 return repo.findByUsername(user);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\ClientUserService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\ListClientUsersController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application;  
  
import eapli.base.clientusermanagement.domain.ClientUser;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
  
/\*\*  
 \*  
 \* @author losa  
 \*/  
public class ListClientUsersController {  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
  
 private final ClientUserRepository repo = PersistenceContext.repositories().clientUsers();  
  
 public Iterable<ClientUser> activeClientUsers() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.MANAGER, BaseRoles.ADMIN);  
  
 return this.repo.findAllActive();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\ListClientUsersController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\eventhandlers\AddClientUserOnSignupAcceptedController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application.eventhandlers;  
  
import java.util.Optional;  
  
import eapli.base.clientusermanagement.domain.ClientUser;  
import eapli.base.clientusermanagement.domain.ClientUserBuilder;  
import eapli.base.clientusermanagement.domain.events.NewUserRegisteredFromSignupEvent;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.framework.functional.Functions;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.repositories.UserRepository;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*  
 \*/  
/\* package \*/ class AddClientUserOnSignupAcceptedController {  
  
 private final UserRepository repo = PersistenceContext.repositories().users();  
 private final ClientUserRepository clientUserRepository = PersistenceContext  
 .repositories().clientUsers();  
  
 public ClientUser addClientUser(final NewUserRegisteredFromSignupEvent event) {  
 final Optional<SystemUser> newUser = findUser(event);  
  
 return newUser.map(u -> {  
 final ClientUserBuilder clientUserBuilder = new ClientUserBuilder();  
 clientUserBuilder.withMecanographicNumber(event.mecanographicNumber())  
 .withSystemUser(u);  
 return clientUserRepository.save(clientUserBuilder.build());  
 }).orElseThrow(IllegalStateException::new);  
 }  
  
 @SuppressWarnings("squid:S1488")  
 private Optional<SystemUser> findUser(final NewUserRegisteredFromSignupEvent event) {  
 // since we are using events, the actual user may not yet be  
 // created, so lets give it a time and wait  
 final Optional<SystemUser> newUser = Functions  
 .retry(() -> repo.ofIdentity(event.username()), 1000, 3);  
 return newUser;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\eventhandlers\AddClientUserOnSignupAcceptedController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\eventhandlers\NewUserRegisteredFromSignupWatchDog.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.application.eventhandlers;  
  
import eapli.base.clientusermanagement.domain.events.NewUserRegisteredFromSignupEvent;  
import eapli.framework.domain.events.DomainEvent;  
import eapli.framework.infrastructure.pubsub.EventHandler;  
  
/\*\*  
 \* @author Paulo Gandra de Sousa  
 \*  
 \*/  
public class NewUserRegisteredFromSignupWatchDog implements EventHandler {  
  
 /\*  
 \* (non-Javadoc)  
 \*  
 \* @see eapli.framework.domain.events.EventHandler#onEvent(eapli.framework.  
 \* domain. events.DomainEvent)  
 \*/  
 @Override  
 public void onEvent(final DomainEvent domainevent) {  
 assert domainevent instanceof NewUserRegisteredFromSignupEvent;  
  
 final NewUserRegisteredFromSignupEvent event = (NewUserRegisteredFromSignupEvent) domainevent;  
  
 final AddClientUserOnSignupAcceptedController  
 controller = new AddClientUserOnSignupAcceptedController();  
 controller.addClientUser(event);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\application\eventhandlers\NewUserRegisteredFromSignupWatchDog.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\ApprovalStatus.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt  
 \*/  
public enum ApprovalStatus {  
 PENDING, ACCEPTED, REFUSED;  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\ApprovalStatus.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\ClientUser.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.domain.model.DomainEntities;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import javax.persistence.EmbeddedId;  
import javax.persistence.Entity;  
import javax.persistence.OneToOne;  
import javax.persistence.Version;  
  
/\*\*  
 \* A Client User.  
 \*  
 \* This class represents client users. It follows a DDD approach where User  
 \* is the root entity of the Base User Aggregate and all of its properties  
 \* are instances of value objects. A Client User references a System User  
 \*  
 \* This approach may seem a little more complex than just having String or  
 \* native type attributes but provides for real semantic of the domain and  
 \* follows the Single Responsibility Pattern  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt  
 \*  
 \*/  
@Entity  
public class ClientUser implements AggregateRoot<MechanographicNumber> {  
  
 @Version  
 private Long version;  
  
 @EmbeddedId  
 private MechanographicNumber mechanographicNumber;  
  
 /\*\*  
 \* cascade = CascadeType.NONE as the systemUser is part of another aggregate  
 \*/  
 @OneToOne()  
 private SystemUser systemUser;  
  
 public ClientUser(final SystemUser user, final MechanographicNumber mechanographicNumber) {  
 if (mechanographicNumber == null || user == null) {  
 throw new IllegalArgumentException();  
 }  
 this.systemUser = user;  
 this.mechanographicNumber = mechanographicNumber;  
 }  
  
 protected ClientUser() {  
 // for ORM only  
 }  
  
 public SystemUser user() {  
 return this.systemUser;  
 }  
  
 @Override  
 public boolean equals(final Object o) {  
 return DomainEntities.areEqual(this, o);  
 }  
  
 @Override  
 public int hashCode() {  
 return DomainEntities.hashCode(this);  
 }  
  
 @Override  
 public boolean sameAs(final Object other) {  
 return DomainEntities.areEqual(this, other);  
 }  
  
 public MechanographicNumber mechanographicNumber() {  
 return identity();  
 }  
  
 @Override  
 public MechanographicNumber identity() {  
 return this.mechanographicNumber;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\ClientUser.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\ClientUserBuilder.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.framework.domain.model.DomainFactory;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
/\*\*  
 \* A factory for User entities.  
 \*  
 \* This class demonstrates the use of the factory (DDD) pattern using a fluent  
 \* interface. it acts as a Builder (GoF).  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
public class ClientUserBuilder implements DomainFactory<ClientUser> {  
  
 private SystemUser systemUser;  
 private MechanographicNumber mechanographicNumber;  
  
 public ClientUserBuilder withSystemUser(final SystemUser systemUser) {  
 this.systemUser = systemUser;  
 return this;  
 }  
  
 public ClientUserBuilder withMecanographicNumber(final MechanographicNumber mechanographicNumber) {  
 this.mechanographicNumber = mechanographicNumber;  
 return this;  
 }  
  
 public ClientUserBuilder withMecanographicNumber(final String mecanographicNumber) {  
 this.mechanographicNumber = new MechanographicNumber(mecanographicNumber);  
 return this;  
 }  
  
 @Override  
 public ClientUser build() {  
 // since the factory knows that all the parts are needed it could throw  
 // an exception. however, we will leave that to the constructor  
 return new ClientUser(this.systemUser, this.mechanographicNumber);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\ClientUserBuilder.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\SignupRequest.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.domain.model.DomainEntities;  
import eapli.framework.general.domain.model.EmailAddress;  
import eapli.framework.infrastructure.authz.domain.model.Name;  
import eapli.framework.infrastructure.authz.domain.model.Password;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.\*;  
import java.util.Calendar;  
  
/\*\*  
 \* A Signup Request. This class represents the Signup Request created right  
 \* after a person applies for a Client User account.  
 \*  
 \* <p>  
 \* It follows a DDD approach where all of its properties are instances of value  
 \* objects. This approach may seem a little more complex than just having String  
 \* or native type attributes but provides for real semantic of the domain and  
 \* follows the Single Responsibility Pattern.  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt  
 \*  
 \*/  
@Entity  
public class SignupRequest implements AggregateRoot<Username> {  
  
 private static final long serialVersionUID = 1L;  
  
 @Version  
 private Long version;  
  
 @EmbeddedId  
 private Username username;  
 private Password password;  
 private Name name;  
 private EmailAddress email;  
  
 private MechanographicNumber mechanographicNumber;  
  
 @Enumerated(EnumType.STRING)  
 private ApprovalStatus approvalStatus;  
 @Temporal(TemporalType.DATE)  
 private Calendar createdOn;  
  
 /\* package \*/ SignupRequest(final Username username, final Password password, final Name name,  
 final EmailAddress email, final MechanographicNumber mechanographicNumber,  
 final Calendar createdOn) {  
 Preconditions.noneNull(username, password, name, email, mechanographicNumber);  
  
 this.username = username;  
 this.password = password;  
 this.name = name;  
 this.email = email;  
 this.mechanographicNumber = mechanographicNumber;  
 // by default  
 approvalStatus = ApprovalStatus.PENDING;  
 this.createdOn = createdOn;  
 }  
  
 protected SignupRequest() {  
 // for ORM only  
 }  
  
 public void accept() {  
 approvalStatus = ApprovalStatus.ACCEPTED;  
 }  
  
 public void refuse() {  
 approvalStatus = ApprovalStatus.REFUSED;  
 }  
  
 @Override  
 public boolean equals(final Object o) {  
 return DomainEntities.areEqual(this, o);  
 }  
  
 @Override  
 public int hashCode() {  
 return DomainEntities.hashCode(this);  
 }  
  
 @Override  
 public boolean sameAs(final Object other) {  
 if (!(other instanceof SignupRequest)) {  
 return false;  
 }  
  
 final SignupRequest that = (SignupRequest) other;  
 if (this == that) {  
 return true;  
 }  
  
 return username.equals(that.username) && password.equals(that.password)  
 && name.equals(that.name) && email.equals(that.email)  
 && mechanographicNumber.equals(that.mechanographicNumber);  
 }  
  
 public MechanographicNumber mecanographicNumber() {  
 return mechanographicNumber;  
 }  
  
 @Override  
 public Username identity() {  
 return username;  
 }  
  
 public Username username() {  
 return username;  
 }  
  
 public Name name() {  
 return name;  
 }  
  
 public boolean isPending() {  
 return approvalStatus == ApprovalStatus.PENDING;  
 }  
  
 public EmailAddress email() {  
 return email;  
 }  
  
 public Password password() {  
 return password;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\SignupRequest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\SignupRequestBuilder.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.framework.domain.model.DomainFactory;  
import eapli.framework.general.domain.model.EmailAddress;  
import eapli.framework.infrastructure.authz.application.PasswordPolicy;  
import eapli.framework.infrastructure.authz.domain.model.Name;  
import eapli.framework.infrastructure.authz.domain.model.Password;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.time.util.CurrentTimeCalendars;  
import org.springframework.security.crypto.password.PasswordEncoder;  
import org.springframework.stereotype.Component;  
  
import java.util.Calendar;  
  
/\*\*  
 \* A factory for Signup Requests.  
 \*  
 \* <p>  
 \* This class demonstrates the use of the factory (DDD) pattern using a fluent  
 \* interface. it acts as a Builder (GoF).  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
@Component  
public class SignupRequestBuilder implements DomainFactory<SignupRequest> {  
  
 private final PasswordPolicy policy;  
 private final PasswordEncoder encoder;  
  
 private String username;  
 private Password password;  
 private String firstName;  
 private String lastName;  
 private String email;  
 private String mecanographicNumber;  
 private Calendar createdOn;  
  
 public SignupRequestBuilder(final PasswordPolicy policy, final PasswordEncoder encoder) {  
 this.policy = policy;  
 this.encoder = encoder;  
 }  
  
 public SignupRequestBuilder withData(final String username, final String rawPassword,  
 final String email, final String number) {  
 withUsername(username);  
 withPassword(rawPassword);  
 withEmail(email);  
 withMecanographicNumber(number);  
 return this;  
 }  
  
 public SignupRequestBuilder withUsername(final String username) {  
 this.username = username;  
 return this;  
 }  
  
 public SignupRequestBuilder withPassword(final String rawPassword) {  
 password = Password.encodedAndValid(rawPassword, policy, encoder)  
 .orElseThrow(IllegalArgumentException::new);  
 return this;  
 }  
  
 public SignupRequestBuilder withName(final String firstName, final String lastName) {  
 this.firstName = firstName;  
 this.lastName = lastName;  
 return this;  
 }  
  
 public SignupRequestBuilder withEmail(final String email) {  
 this.email = email;  
 return this;  
 }  
  
 public SignupRequestBuilder withMecanographicNumber(final String mecanographicNumber) {  
 this.mecanographicNumber = mecanographicNumber;  
 return this;  
 }  
  
 public SignupRequestBuilder createdOn(final Calendar createdOn) {  
 this.createdOn = createdOn;  
 return this;  
 }  
  
 @Override  
 public SignupRequest build() {  
 // since the factory knows that all the parts are needed it could throw  
 // an exception. however, we will leave that to the constructor  
 if (createdOn != null) {  
 createdOn = CurrentTimeCalendars.now();  
 }  
 return new SignupRequest(Username.valueOf(username), password,  
 Name.valueOf(firstName, lastName), EmailAddress.valueOf(email),  
 MechanographicNumber.valueOf(mecanographicNumber), createdOn);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\SignupRequestBuilder.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\events\NewUserRegisteredFromSignupEvent.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain.events;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.framework.domain.events.DomainEvent;  
import eapli.framework.domain.events.DomainEventBase;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
/\*\*  
 \* @author Paulo Gandra de Sousa  
 \*  
 \*/  
public class NewUserRegisteredFromSignupEvent extends DomainEventBase implements DomainEvent {  
  
 private static final long serialVersionUID = 1L;  
  
 private final MechanographicNumber mechanographicNumber;  
 private final Username newUser;  
  
 public NewUserRegisteredFromSignupEvent(final MechanographicNumber mechanographicNumber,  
 final Username newUser) {  
 this.mechanographicNumber = mechanographicNumber;  
 this.newUser = newUser;  
 }  
  
 public MechanographicNumber mecanographicNumber() {  
 return mechanographicNumber;  
 }  
  
 public Username username() {  
 return newUser;  
 }  
  
 @Override  
 public String toString() {  
 return "NewUserFromsignup(" + username() + ")";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\events\NewUserRegisteredFromSignupEvent.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\events\SignupAcceptedEvent.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain.events;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.framework.domain.events.DomainEventBase;  
import eapli.framework.general.domain.model.EmailAddress;  
import eapli.framework.infrastructure.authz.domain.model.Name;  
import eapli.framework.infrastructure.authz.domain.model.Password;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
/\*\*  
 \* @author Paulo Gandra de Sousa  
 \*  
 \*/  
public class SignupAcceptedEvent extends DomainEventBase {  
  
 private static final long serialVersionUID = 1L;  
  
 private final SignupRequest theSignupRequest;  
  
 public SignupAcceptedEvent(final SignupRequest theSignupRequest) {  
 this.theSignupRequest = theSignupRequest;  
 }  
  
 public Username username() {  
 return theSignupRequest.username();  
 }  
  
 public Password password() {  
 return theSignupRequest.password();  
 }  
  
 public Name name() {  
 return theSignupRequest.name();  
 }  
  
 public EmailAddress email() {  
 return theSignupRequest.email();  
 }  
  
 public MechanographicNumber mecanographicNumber() {  
 return theSignupRequest.mecanographicNumber();  
 }  
  
 @Override  
 public String toString() {  
 return "SignupAccepted(" + username() + ")";  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\domain\events\SignupAcceptedEvent.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\repositories\ClientUserRepository.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.repositories;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.clientusermanagement.domain.ClientUser;  
import eapli.framework.domain.repositories.DomainRepository;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
import java.util.Optional;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
public interface ClientUserRepository extends DomainRepository<MechanographicNumber, ClientUser> {  
  
 /\*\*  
 \* returns the client user (utente) whose username is given  
 \*  
 \* @param name  
 \* the username to search for  
 \* @return  
 \*/  
 Optional<ClientUser> findByUsername(Username name);  
  
 /\*\*  
 \* returns the client user (utente) with the given mecanographic number  
 \*  
 \* @param number  
 \* @return  
 \*/  
 default Optional<ClientUser> findByMechanographicNumber(final MechanographicNumber number) {  
 return ofIdentity(number);  
 }  
  
 public Iterable<ClientUser> findAllActive();  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\repositories\ClientUserRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\repositories\SignupRequestRepository.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.repositories;  
  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.framework.domain.repositories.DomainRepository;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
public interface SignupRequestRepository extends DomainRepository<Username, SignupRequest> {  
  
 Iterable<SignupRequest> pendingSignupRequests();  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\clientusermanagement\repositories\SignupRequestRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\CreateExamController.java

package eapli.base.exammanagement.application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.domain.ExamTime;  
import eapli.base.exammanagement.domain.ExamDate;  
import eapli.base.exammanagement.domain.ExamTitle;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
  
import java.sql.Date;  
import java.sql.Time;  
  
/\*\*  
 \* CreateExamController - Criar um Exame  
 \*   
 \* Created by João Cruz  
 \*  
 \*/  
  
@UseCaseController  
public class CreateExamController {  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private ExamRepository examRepository;  
  
 public void setExamRepository(ExamRepository examRepository) {  
 this.examRepository = examRepository;  
 }  
  
 public Exam createExam(String examName, Date examDate, Time openTime, Time closeTime, Course course) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
  
 ExamTitle examTitle = new ExamTitle(examName);  
 ExamDate examDateObj = new ExamDate(examDate);  
 ExamTime examTime = new ExamTime(openTime, closeTime);  
  
 Exam newExam = new Exam(examTitle, examDateObj, examTime, course);  
 examRepository.save(newExam); // Salva no repositório do Exame  
 return newExam;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\CreateExamController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\CreateExamService.java

package eapli.base.exammanagement.application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.ApplicationService;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
  
/\*\*  
 \* Adicionar Serviço de Exame  
 \* Created and Edited by João Cruz  
 \* Inspired and Based on Daniel Braga's and EAPLI's code  
 \*/  
  
@ApplicationService  
public class CreateExamService {  
 private final AuthorizationService authz;  
 private final ExamRepository examRepository;  
 private final CourseRepository courseRepository;  
  
 public CreateExamService(final AuthorizationService authz, final ExamRepository examRepository, final CourseRepository courseRepository) {  
 this.authz = authz;  
 this.examRepository = examRepository;  
 this.courseRepository = courseRepository;  
 }  
  
 public Iterable<Exam> courseExams(Course course) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
  
 return examRepository.findByCourse(course);  
 }  
  
 public Iterable<Course> courses() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
  
 return courseRepository.findAll();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\CreateExamService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListCourseExamsController.java

package eapli.base.exammanagement.application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
  
import java.util.Iterator;  
  
public class ListCourseExamsController {  
  
 private final ListCourseExamsService svc;  
  
 public ListCourseExamsController(final AuthorizationService authz, final ExamRepository examRepository, final CourseRepository courseRepository) {  
 // dependency injection - to make this object more testable we don't create the  
 // infrastructure objects to avoid coupling to the implementation. This way, the controller  
 // can be used in different scenarios with different implementations of the repository. for  
 // instance, unit testing.  
 svc = new ListCourseExamsService(authz, examRepository, courseRepository);  
 }  
  
 /\*\*  
 \* List all exams.  
 \*  
 \* @return  
 \*/  
 public Iterable<Exam> listCourseExams(Course course) {  
 return svc.courseExams(course);  
 }  
  
 public Iterable<Course> listCourses() {  
 return svc.courses();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListCourseExamsController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListCourseExamsService.java

package eapli.base.exammanagement.application;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.ApplicationService;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
  
import java.util.Iterator;  
  
@ApplicationService  
class ListCourseExamsService {  
 private final AuthorizationService authz;  
 private final ExamRepository examRepository;  
 private final CourseRepository courseRepository;  
  
 public ListCourseExamsService(final AuthorizationService authz, final ExamRepository examRepository, final CourseRepository courseRepository) {  
 this.authz = authz;  
 this.examRepository = examRepository;  
 this.courseRepository = courseRepository;  
 }  
  
 /\*\*  
 \* @return  
 \*/  
 public Iterable<Exam> courseExams(Course course) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
  
 return examRepository.findByCourse(course);  
 }  
  
 public Iterable<Course> courses() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.TEACHER, BaseRoles.ADMIN);  
  
 return courseRepository.findAll();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListCourseExamsService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListFutureExamsController.java

package eapli.base.exammanagement.application;  
  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import java.util.Optional;  
  
public class ListFutureExamsController {  
  
 private final ListFutureExamsService svc;  
  
 public ListFutureExamsController(final AuthorizationService authz, final ExamRepository examRepository, final StudentRepository studentRepository) {  
 // dependency injection - to make this object more testable we don't create the  
 // infrastructure objects to avoid coupling to the implementation. This way, the controller  
 // can be used in different scenarios with different implementations of the repository. for  
 // instance, unit testing.  
 svc = new ListFutureExamsService(authz, examRepository, studentRepository);  
 }  
  
 /\*\*  
 \* List all exams.  
 \*  
 \* @return  
 \*/  
 public Iterable<Exam> listFutureExams(Student student) {  
 return svc.futureExams(student);  
 }  
  
 public Student findStudentBySystemUser(SystemUser systemUser){  
 return svc.findStudentBySystemUser(systemUser);  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListFutureExamsController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListFutureExamsService.java

package eapli.base.exammanagement.application;  
  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import java.util.Optional;  
  
public class ListFutureExamsService {  
  
 private final AuthorizationService authz;  
 private final ExamRepository examRepository;  
  
 private final StudentRepository studentRepository;  
  
 public ListFutureExamsService(final AuthorizationService authz, final ExamRepository examRepository, final StudentRepository studentRepository) {  
 this.authz = authz;  
 this.examRepository = examRepository;  
 this.studentRepository=studentRepository;  
 }  
  
 /\*\*  
 \* @return  
 \*/  
 public Iterable<Exam> futureExams(Student student) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.STUDENT, BaseRoles.ADMIN);  
  
 return examRepository.findStudentFutureExams(student);  
 }  
  
 public Student findStudentBySystemUser(SystemUser systemUser){  
 return studentRepository.findStudentBySystemUser(systemUser);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\application\ListFutureExamsService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\Exam.java

package eapli.base.exammanagement.domain;  
  
import eapli.base.Course.Domain.Course;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.domain.model.DomainEntities;  
import eapli.framework.validations.Preconditions;  
import lombok.NoArgsConstructor;  
  
import javax.persistence.\*;  
import java.lang.String;  
  
@Entity  
@Table(name = "EXAM")  
public class Exam implements AggregateRoot<Long> {  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name = "IDEXAM")  
 private Long id;  
  
 @Embedded  
 private ExamTitle examTitle;  
  
 @Embedded  
 private ExamResult result;  
  
 @Embedded  
 private ExamTime examTime;  
  
 @Embedded  
 private ExamDate examDate;  
  
 @ManyToOne  
 @JoinColumn(name = "COURSE\_ID")  
 private Course examCourse;  
  
 public Exam(ExamTitle examTitle, ExamDate examDate, ExamTime examTime, Course course) {  
 Preconditions.noneNull(examTitle, examDate, course);  
  
 this.examTitle = examTitle;  
 this.examDate = examDate;  
 this.examTime = examTime;  
 this.examCourse = course;  
 this.result = null;  
 }  
  
 protected Exam(){}  
  
 @Override  
 public boolean equals(Object o) {  
 return DomainEntities.areEqual(this, o);  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 Exam exam = (Exam) other;  
 return this.equals(exam) && equals(exam.getExamCourse()) && getExamDate().equals(exam.getExamDate()) && getExamTime().equals(exam.getExamTime()) && getResult().equals(exam.getResult()) && getExamTitle().equals(exam.getExamTitle());  
 }  
  
 public ExamTitle getExamTitle() {  
 return this.examTitle;  
 }  
  
 public Course getExamCourse() {  
 return this.examCourse;  
 }  
  
 public ExamDate getExamDate() {  
 return this.examDate;  
 }  
  
 public ExamTime getExamTime() {  
 return this.examTime;  
 }  
  
 public ExamResult getResult() {  
 return this.result;  
 }  
  
 public void setResult(ExamResult result) {  
 this.result = result;  
 }  
  
 @Override  
 public String toString() {  
 return id.toString();  
 }  
  
 @Override  
 public Long identity() {  
 return this.id;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\Exam.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamDate.java

package eapli.base.exammanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import org.springframework.format.annotation.DateTimeFormat;  
  
import javax.persistence.Embeddable;  
import java.util.Date;  
  
@Embeddable  
public class ExamDate implements ValueObject {  
  
 @DateTimeFormat(pattern = "dd-MM-yyyy")  
 private Date examDate;  
  
 protected ExamDate() {  
 }  
  
 public ExamDate(Date examDate) {  
 if (!isValidExamDate(examDate)) {  
 throw new IllegalArgumentException("Invalid exam date format");  
 }  
 this.examDate = examDate;  
 }  
  
 private static boolean isValidExamDate(Date examDate) {  
 Date date = new Date(System.currentTimeMillis());  
 return examDate.after(date);  
 }  
  
 public Date getExamDate() {  
 return examDate;  
 }  
  
 public void setExamDate(Date examDate) {  
 this.examDate = examDate;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 ExamDate that = (ExamDate) o;  
 return examDate.equals(that.examDate);  
 }  
  
 @Override  
 public String toString() {  
 return examDate.toString();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamDate.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamResult.java

package eapli.base.exammanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
  
import javax.persistence.Embeddable;  
  
@Embeddable  
public class ExamResult implements ValueObject {  
  
 private float number;  
  
 protected ExamResult(){  
 }  
  
 public ExamResult(float number) {  
 this.number = number;  
 }  
  
 public static ExamResult valueOf(float number) {  
 if (!isValidExamResult(number)) {  
 throw new IllegalArgumentException("Invalid exam result format");  
 }  
 return new ExamResult(number);  
 }  
  
 private static boolean isValidExamResult(float number) {  
 return number > 0 && number < 20;  
 }  
  
 public float getNumber() {  
 return number;  
 }  
  
 public void setNumber(float number) {  
 this.number = number;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 ExamResult that = (ExamResult) o;  
 return Float.compare(that.number, number) == 0;  
 }  
  
 @Override  
 public int hashCode() {  
 return Float.hashCode(number);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamResult.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamTime.java

package eapli.base.exammanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import org.springframework.format.annotation.DateTimeFormat;  
  
import javax.persistence.Embeddable;  
import java.util.Date;  
import java.util.Objects;  
  
@Embeddable  
public class ExamTime implements ValueObject {  
  
 @DateTimeFormat(pattern = "hh:mm")  
 private Date openDate;  
 @DateTimeFormat(pattern = "hh:mm")  
 private Date closeDate;  
  
 protected ExamTime() {  
 }  
  
 public ExamTime(Date openDate, Date closeDate) {  
 if (!isValidExamTime(openDate, closeDate)) {  
 throw new IllegalArgumentException("Invalid exam time format");  
 }  
 this.openDate = openDate;  
 this.closeDate = closeDate;  
 }  
  
 private static boolean isValidExamTime(Date openDate, Date closeDate) {  
 return closeDate.after(openDate);  
 }  
  
 public Date getOpenDate() {  
 return openDate;  
 }  
  
 public void setOpenDate(Date openDate) {  
 this.openDate = openDate;  
 }  
  
 public Date getCloseDate() {  
 return closeDate;  
 }  
  
 public void setCloseDate(Date closeDate) {  
 this.closeDate = closeDate;  
 }  
  
 @Override  
 public String toString() {  
 return openDate.toString() + " - " + closeDate.toString();  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 ExamTime examTime = (ExamTime) o;  
 return openDate.equals(examTime.openDate) && closeDate.equals(examTime.closeDate);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(openDate, closeDate);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamTime.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamTitle.java

package eapli.base.exammanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
import java.util.Objects;  
  
@Embeddable  
public class ExamTitle implements ValueObject {  
  
 private String title;  
  
 protected ExamTitle() {  
 }  
  
 public ExamTitle(String title) throws IllegalArgumentException {  
 Preconditions.ensure(title != null && !title.isEmpty(), "Invalid title");  
 this.title = title;  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Title: %s", title);  
 }  
  
 public String getTitle() {  
 return this.title;  
 }  
  
 public String setTitle() {  
 return this.title;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
  
 ExamTitle other = (ExamTitle) o;  
 return Objects.equals(title, other.title);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(title);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\domain\ExamTitle.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\repository\ExamRepository.java

package eapli.base.exammanagement.repository;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.framework.domain.repositories.DomainRepository;  
  
public interface ExamRepository extends DomainRepository<Long, Exam> {  
  
 Iterable<Exam> findStudentFutureExams(Student student);  
  
 Iterable<Exam> findByCourse(Course course);  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\exammanagement\repository\ExamRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\infrastructure\persistence\PersistenceContext.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.persistence;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import eapli.base.Application;  
import eapli.framework.util.Utility;  
  
/\*\*  
 \* provides easy access to the persistence layer. works as a factory of  
 \* factories  
 \*  
 \* @author Paulo Gandra Sousa  
 \*/  
@Utility  
public final class PersistenceContext {  
 private static final Logger LOGGER = LoggerFactory.getLogger(PersistenceContext.class);  
  
 private static RepositoryFactory theFactory;  
  
 private PersistenceContext() {  
 // ensure utility  
 }  
  
 public static RepositoryFactory repositories() {  
 if (theFactory == null) {  
 final String factoryClassName = Application.settings().getRepositoryFactory();  
 try {  
 theFactory = (RepositoryFactory) Class.forName(factoryClassName).newInstance();  
 } catch (ClassNotFoundException | IllegalAccessException | InstantiationException ex) {  
 LOGGER.error("Unable to dynamically load the Repository Factory", ex);  
 throw new IllegalStateException(  
 "Unable to dynamically load the Repository Factory: " + factoryClassName, ex);  
 }  
 }  
 return theFactory;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\infrastructure\persistence\PersistenceContext.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\infrastructure\persistence\RepositoryFactory.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.infrastructure.persistence;  
  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.ExtraClasse.aplication.ExtraClassController;  
import eapli.base.ExtraClasse.repository.ExtraClasseRepository;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.base.systemUserManagement.SystemUserRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.repositories.UserRepository;  
  
/\*\*  
 \* @author Paulo Gandra Sousa  
 \*  
 \*/  
public interface RepositoryFactory {  
  
 /\*\*  
 \* factory method to create a transactional context to use in the repositories  
 \*  
 \* @return  
 \*/  
 TransactionalContext newTransactionalContext();  
  
 /\*\*  
 \*  
 \* @param autoTx  
 \* the transactional context to enrol  
 \* @return  
 \*/  
 UserRepository users(TransactionalContext autoTx);  
  
 /\*\*  
 \* repository will be created in auto transaction mode  
 \*  
 \* @return  
 \*/  
 UserRepository users();  
  
 /\*\*  
 \*  
 \* @param autoTx  
 \* the transactional context to enroll  
 \* @return  
 \*/  
 ClientUserRepository clientUsers(TransactionalContext autoTx);  
  
 /\*\*  
 \* repository will be created in auto transaction mode  
 \*  
 \* @return  
 \*/  
 ClientUserRepository clientUsers();  
  
 /\*\*  
 \*  
 \* @param autoTx  
 \* the transactional context to enroll  
 \* @return  
 \*/  
 SignupRequestRepository signupRequests(TransactionalContext autoTx);  
  
 /\*\*  
 \* repository will be created in auto transaction mode  
 \*  
 \* @return  
 \*/  
 SignupRequestRepository signupRequests();  
  
 ExamRepository examRepository();  
  
 CourseRepository courseRepository();  
  
 MeetingRepository meetingRepository();  
  
 InviteRepository inviteRepository();  
  
 ClassRepository classRepository();  
  
 ExtraClasseRepository extraClassRepostory();  
  
 SystemUserRepository systemUserRepository();  
  
  
 TeacherRepository teacherRepository();  
  
 StudentRepository studentRepository();  
  
 ClientUserRepository clientUserRepository();  
  
 SharedBoardRepository sharedBoardRepository();  
 eapli.base.Enrollment.Repository.EnrollmentRepository enrollmentRepository();  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\infrastructure\persistence\RepositoryFactory.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\AcceptRefuseInvitesController.java

package eapli.base.meetingmanagement.application;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
public class AcceptRefuseInvitesController {  
  
 private final AcceptRefuseInvitesService svc;  
  
 public AcceptRefuseInvitesController(final AuthorizationService authz, final InviteRepository inviteRepository) {  
 // dependency injection - to make this object more testable we don't create the  
 // infrastructure objects to avoid coupling to the implementation. This way, the controller  
 // can be used in different scenarios with different implementations of the repository. for  
 // instance, unit testing.  
 svc = new AcceptRefuseInvitesService(authz, inviteRepository);  
 }  
  
 /\*\*  
 \* List all exams.  
 \*  
 \* @return  
 \*/  
// public Iterable<Invite> invitesReceived(Username username) {  
// return svc.invitesReceived(username);  
// }  
  
 public void acceptInvite(Invite invite, SystemUser systemUser){  
 svc.acceptInvite(invite, systemUser);  
 }  
  
 public void refuseInvite(Invite invite){  
 svc.refuseInvite(invite);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\AcceptRefuseInvitesController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\AcceptRefuseInvitesService.java

package eapli.base.meetingmanagement.application;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.InviteState;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
public class AcceptRefuseInvitesService {  
  
 private final AuthorizationService authz;  
 private final InviteRepository inviteRepository;  
  
 public AcceptRefuseInvitesService(final AuthorizationService authz, final InviteRepository inviteRepository) {  
 this.authz = authz;  
 this.inviteRepository = inviteRepository;  
 }  
  
 /\*\*  
 \* @return  
 \*/  
// public Iterable<Invite> invitesReceived(Username username) {  
// return inviteRepository.findInvitesByReceiverUsername(username);  
// }  
  
 public void acceptInvite(Invite invite, SystemUser systemUser){  
 invite.addMeetingParticipants(systemUser);  
 invite.setState(InviteState.ACCEPTED);  
 }  
  
 public void refuseInvite(Invite invite){  
 invite.setState(InviteState.REJECTED);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\AcceptRefuseInvitesService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\CancelMeetingController.java

package eapli.base.meetingmanagement.application;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.domain.MeetingTitle;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
public class CancelMeetingController {  
 private final CancelMeetingService cms;  
  
  
 public CancelMeetingController(final AuthorizationService authz, final MeetingRepository meetingRepository) {  
 cms = new CancelMeetingService(authz, meetingRepository);  
 }  
 public void cancelMeeting(MeetingTitle meetingTitle) {  
 cms.cancelMeeting(meetingTitle);  
 }  
  
 public Iterable<Meeting> allMeetings() {  
 return this.cms.allMeetings();  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\CancelMeetingController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\CancelMeetingService.java

package eapli.base.meetingmanagement.application;  
  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.domain.MeetingTitle;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
  
public class CancelMeetingService {  
  
 private final AuthorizationService authz;  
 private final MeetingRepository meetingRepository;  
  
 public CancelMeetingService(final AuthorizationService authz, final MeetingRepository meetingRepository) {  
 this.authz = authz;  
 this.meetingRepository = meetingRepository;  
 }  
  
 public void cancelMeeting(MeetingTitle meetingTitle) {  
 Meeting meeting = meetingRepository.findByMeetingById(meetingTitle);  
 meeting.cancelMeeting();  
 meetingRepository.save(meeting);  
 }  
  
 public Iterable<Meeting> allMeetings() {  
 return this.meetingRepository.findAll();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\CancelMeetingService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\ListParticipantController.java

package eapli.base.meetingmanagement.application;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.InviteState;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.domain.MeetingTitle;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
import java.util.AbstractMap;  
import java.util.List;  
  
public class ListParticipantController {  
  
 private final ListParticipantService listParticipantService;  
  
 public ListParticipantController(MeetingRepository meetingRepository, InviteRepository inviteRepository) {  
 this.listParticipantService = new ListParticipantService(meetingRepository, inviteRepository);  
 }  
 public Meeting getMeetingById(MeetingTitle meetingTitle) {  
 return listParticipantService.findByMeetingById(meetingTitle);  
 }  
  
 public Iterable<Invite> findInvitesByMeeting(Meeting meeting) {  
 return listParticipantService.findInvitesByMeeting(meeting);  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\ListParticipantController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\ListParticipantService.java

package eapli.base.meetingmanagement.application;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.InviteState;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.domain.MeetingTitle;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.validations.Preconditions;  
  
import java.util.AbstractMap;  
import java.util.AbstractMap.SimpleEntry;  
import java.util.List;  
  
public class ListParticipantService {  
 private final InviteRepository inviteRepository;  
  
 private final MeetingRepository meetingRepository;  
  
 public ListParticipantService(MeetingRepository meetingRepository, InviteRepository inviteRepository){  
 this.meetingRepository=meetingRepository;  
 this.inviteRepository=inviteRepository;  
 }  
  
 public Meeting findByMeetingById(MeetingTitle meetingTitle){  
 return meetingRepository.findByMeetingById(meetingTitle);  
 }  
  
 public Iterable<Invite> findInvitesByMeeting(Meeting meeting){  
 return inviteRepository.findInvitesByMeeting(meeting);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\ListParticipantService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\ScheduleMeetingController.java

package eapli.base.meetingmanagement.application;  
  
import eapli.base.meetingmanagement.domain.\*;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
public class ScheduleMeetingController {  
  
 private final AuthorizationService authz;  
 private final MeetingRepository meetingRepository;  
 private final InviteRepository inviteRepository;  
  
 public ScheduleMeetingController(final AuthorizationService authz, final MeetingRepository meetingRepository, InviteRepository inviteRepository) {  
 // dependency injection - to make this object more testable we don't create the  
 // infrastructure objects to avoid coupling to the implementation. This way, the controller  
 // can be used in different scenarios with different implementations of the repository. for  
 // instance, unit testing.  
 this.authz= authz;  
 this.meetingRepository=meetingRepository;  
 this.inviteRepository=inviteRepository;  
 }  
  
 public Meeting saveMeeting(final MeetingTitle meetingTitle, final SystemUser systemUser, final MeetingDuration meetingDuration, final MeetingDate meetingDate, final MeetingTime meetingTime){  
 return meetingRepository.save(new Meeting(meetingTitle, systemUser, meetingDuration, meetingDate, meetingTime));  
 }  
  
 public Invite saveInvite(final SystemUser sender, final SystemUser receiver, final Meeting meeting){  
 return inviteRepository.save(new Invite(sender, receiver, meeting));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\application\ScheduleMeetingController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\Invite.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.domain.model.DomainEntities;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.\*;  
  
@Entity  
@Table(name="Invite")  
public class Invite implements AggregateRoot<Long>{  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name="IDINVITE")  
 private Long id;  
  
 private Meeting meeting;  
  
 private SystemUser sender;  
  
 private SystemUser receiver;  
  
 private InviteState state;  
  
 public SystemUser getSender() {  
 return sender;  
 }  
  
 public SystemUser getReceiver() {  
 return receiver;  
 }  
  
 public Meeting getMeeting(){return meeting;}  
  
 protected Invite(){}  
  
 public Invite (SystemUser sender, SystemUser receiver, Meeting meeting){  
 Preconditions.noneNull(sender, receiver, meeting);  
  
 this.sender=sender;  
 this.receiver=receiver;  
 this.meeting=meeting;  
 this.state=InviteState.PENDING;  
 }  
  
 public void addMeetingParticipants(SystemUser systemUser){  
 this.meeting.addParticipants(systemUser);  
 }  
  
 public void setState(InviteState inviteState){  
 this.state=inviteState;  
 }  
  
 public InviteState getState() {  
 return state;  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 Invite invite = (Invite) other;  
 return this.equals(invite);  
 }  
  
 @Override  
 public Long identity() {  
 return id;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 return DomainEntities.areEqual(this, o);  
 }  
  
 @Override  
 public String toString() {  
 return "You've been invited by " + sender.email().toString() + " to a meeting on the date " + meeting.getMeetingDate().getMeetingDate().toString() +  
 " at " + meeting.getMeetingTime().getMeetingTime() + " with the dutarion of " + meeting.getMeetingDuration().toString();  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\Invite.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\InviteState.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
  
public enum InviteState implements ValueObject {  
 PENDING, ACCEPTED, REJECTED;  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\InviteState.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\Meeting.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.domain.ExamTitle;  
import eapli.framework.domain.model.AggregateRoot;  
import eapli.framework.domain.model.DomainEntities;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.\*;  
import java.util.AbstractMap;  
import java.util.ArrayList;  
import java.util.List;  
  
@Entity  
public class Meeting implements AggregateRoot<Long> {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name = "IDMEETING")  
 private Long id;  
  
 private MeetingTitle meetingTitle;  
  
 private SystemUser systemUser;  
  
 private MeetingDuration meetingDuration;  
  
 private MeetingDate meetingDate;  
  
 private MeetingTime meetingTime;  
 @OneToMany  
 private List<SystemUser> participants;  
  
 @Column(name="STATUS")  
 @Enumerated(EnumType.STRING)  
 private MeetingStatus status;  
  
 protected Meeting() {  
 }  
  
 public Meeting (MeetingTitle meetingTitle, SystemUser systemUser, MeetingDuration meetingDuration, MeetingDate meetingDate, MeetingTime meetingTime){  
 Preconditions.noneNull(systemUser, meetingDuration, meetingDate, meetingTime);  
  
 this.meetingTitle=meetingTitle;  
 this.systemUser=systemUser;  
 this.meetingDuration=meetingDuration;  
 this.meetingDate=meetingDate;  
 this.meetingTime=meetingTime;  
 this.participants=new ArrayList<>();  
 this.status = MeetingStatus.SCHEDULED;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 return DomainEntities.areEqual(this, o);  
 }  
  
 @Override  
 public boolean sameAs(Object other) {  
 Meeting meeting = (Meeting) other;  
 return this.equals(meeting) && getMeetingDate().equals(meeting.getMeetingDate()) && getMeetingDuration().equals(meeting.getMeetingDuration()) && getMeetingTime().equals(meeting.getMeetingTime());  
 }  
  
 public void addParticipants(SystemUser systemUser){  
 participants.add(systemUser);  
 }  
  
 @Override  
 public Long identity() {  
 return id;  
 }  
  
 public MeetingTitle getMeetingTitle() {  
 return meetingTitle;  
 }  
  
 public MeetingDate getMeetingDate() {  
 return meetingDate;  
 }  
  
 public MeetingTime getMeetingTime() {  
 return meetingTime;  
 }  
  
 public MeetingDuration getMeetingDuration() {  
 return meetingDuration;  
 }  
  
 public SystemUser getSystemUser() {  
 return systemUser;  
 }  
  
 public List<SystemUser> getParticipants() {  
 return participants;  
 }  
  
 public MeetingStatus getStatus() {  
 return status;  
 }  
 public void cancelMeeting() {  
 this.status = MeetingStatus.CANCELED;  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\Meeting.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingDate.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import org.springframework.format.annotation.DateTimeFormat;  
  
import javax.persistence.Embeddable;  
import java.util.Date;  
  
@Embeddable  
public class MeetingDate implements ValueObject {  
  
 @DateTimeFormat(pattern = "dd-MM-yyyy")  
 private Date meetingDate;  
  
 protected MeetingDate(){}  
  
 public MeetingDate(Date meetingDate) {  
 if (!isValidMeetingDate(meetingDate)) {  
 throw new IllegalArgumentException("Invalid meeting date format");  
 }  
 this.meetingDate=meetingDate;  
 }  
  
 private static boolean isValidMeetingDate(Date meetingDate) {  
 Date date = new Date(System.currentTimeMillis());  
 return meetingDate.after(date);  
 }  
  
 @Override  
 public String toString() {  
 return meetingDate.toString();  
 }  
  
 public Date getMeetingDate(){return meetingDate;}  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingDate.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingDuration.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
  
import javax.persistence.Embeddable;  
  
@Embeddable  
public class MeetingDuration implements ValueObject {  
  
 private Integer value;  
  
 protected MeetingDuration(){}  
  
 public MeetingDuration(int value) {  
 if (!isValidMeetingDuration(value)) {  
 throw new IllegalArgumentException("Invalid meeting duration format");  
 }  
 this.value=value;  
 }  
  
 private static boolean isValidMeetingDuration(int value) {  
 return value>0;  
 }  
  
 public Integer getValue(){return value;}  
  
 @Override  
 public String toString() {  
 return value.toString();  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingDuration.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingStatus.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import lombok.\*;  
  
import javax.persistence.Embeddable;  
import java.util.Objects;  
  
  
public enum MeetingStatus {  
 SCHEDULED, COMPLETED, CANCELED;  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingStatus.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingTime.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.framework.domain.model.ValueObject;  
import org.springframework.format.annotation.DateTimeFormat;  
  
import javax.persistence.Embeddable;  
import java.util.Date;  
  
@Embeddable  
public class MeetingTime implements ValueObject {  
  
 @DateTimeFormat(pattern = "hh:mm")  
 private Date meetingTime;  
  
 protected MeetingTime(){}  
  
 public MeetingTime(Date meetingTime) {this.meetingTime=meetingTime;  
 }  
  
 public Date getMeetingTime(){return meetingTime;}  
  
 @Override  
 public String toString() {  
 return meetingTime.toString();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingTime.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingTitle.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.base.exammanagement.domain.ExamTitle;  
import eapli.framework.domain.model.ValueObject;  
import eapli.framework.validations.Preconditions;  
  
import javax.persistence.Embeddable;  
import java.util.Objects;  
  
@Embeddable  
public class MeetingTitle implements ValueObject {  
  
 private String title;  
  
 protected MeetingTitle() {  
 }  
  
 public MeetingTitle(String title) throws IllegalArgumentException {  
 Preconditions.ensure(title != null && !title.isEmpty(), "Invalid title");  
 this.title = title;  
 }  
  
 @Override  
 public String toString() {  
 return String.format("Title: %s", title);  
 }  
  
 public String getTitle() {  
 return this.title;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
  
 MeetingTitle other = (MeetingTitle) o;  
 return Objects.equals(title, other.title);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(title);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\domain\MeetingTitle.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\repository\InviteRepository.java

package eapli.base.meetingmanagement.repository;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.framework.domain.repositories.DomainRepository;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
import java.util.Optional;  
  
public interface InviteRepository extends DomainRepository<Long, Invite> {  
  
// Iterable<Invite> findInvitesByReceiverUsername(Username username);  
  
 Iterable<Invite> findInvitesByMeeting(Meeting meeting);  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\repository\InviteRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\repository\MeetingRepository.java

package eapli.base.meetingmanagement.repository;  
  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.domain.MeetingTitle;  
import eapli.framework.domain.repositories.DomainRepository;  
  
public interface MeetingRepository extends DomainRepository<Long, Meeting> {  
  
 Meeting findByMeetingById(MeetingTitle meetingTitle);  
  
 Iterable<Meeting> findAll();  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\meetingmanagement\repository\MeetingRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\myclientuser\application\MyClientUserService.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.myclientuser.application;  
  
import java.util.Optional;  
  
import eapli.base.clientusermanagement.domain.ClientUser;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*/  
public class MyClientUserService {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final ClientUserRepository repo = PersistenceContext.repositories().clientUsers();  
  
 public ClientUser me() {  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser myUser = s.authenticatedUser();  
 // TODO cache the client user object  
 final Optional<ClientUser> me = repo.findByUsername(myUser.identity());  
 return me.orElseThrow(IllegalStateException::new);  
 }  
  
 public ClientUser myUser() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.CLIENT\_USER);  
 final UserSession s = authz.session().orElseThrow(IllegalStateException::new);  
 final SystemUser me = s.authenticatedUser();  
 return repo.findByUsername(me.identity()).orElseThrow(IllegalStateException::new);  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\myclientuser\application\MyClientUserService.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\myclientuser\application\SignupController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.myclientuser.application;  
  
import java.util.Calendar;  
  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.base.clientusermanagement.domain.SignupRequestBuilder;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.UserBuilderHelper;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.time.util.CurrentTimeCalendars;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt  
 \*/  
@UseCaseController  
public class SignupController {  
  
 private final SignupRequestRepository signupRequestRepository = PersistenceContext  
 .repositories().signupRequests();  
  
 public SignupRequest signup(final String username, final String password,  
 final String firstName, final String lastName, final String email,  
 String mecanographicNumber, final Calendar createdOn) {  
  
 // there is no need for authorisation check in this method as even  
 // unauthenticated users may request a signup  
  
 final SignupRequestBuilder signupRequestBuilder = UserBuilderHelper.signupBuilder();  
 signupRequestBuilder.withUsername(username).withPassword(password)  
 .withName(firstName, lastName).withEmail(email).createdOn(createdOn)  
 .withMecanographicNumber(mecanographicNumber);  
  
 final SignupRequest newSignupRequest = signupRequestBuilder.build();  
 return this.signupRequestRepository.save(newSignupRequest);  
 }  
  
 public SignupRequest signup(final String username, final String password,  
 final String firstName, final String lastName, final String email,  
 String mecanographicNumber) {  
  
 return signup(username, password, firstName, lastName, email, mecanographicNumber,  
 CurrentTimeCalendars.now());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\myclientuser\application\SignupController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\systemUserManagement\SystemUserRepository.java

package eapli.base.systemUserManagement;  
  
import eapli.framework.domain.repositories.DomainRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
import java.util.List;  
  
public interface SystemUserRepository extends DomainRepository<Username, SystemUser> {  
  
 Iterable<SystemUser> findByUsername(Username name);  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\systemUserManagement\SystemUserRepository.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\AddUserController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.usermanagement.application;  
  
import java.util.Calendar;  
import java.util.Set;  
  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserManagementService;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.time.util.CurrentTimeCalendars;  
  
/\*\*  
 \*  
 \* Created by nuno on 21/03/16.  
 \* Edited by João Cruz  
 \*  
 \* Classe destinada a adicionar utilizadores.  
 \* É necessário ter em conta que o utilizador que está a adicionar outro utilizador tem de ter as devidas permissões conforme o ensureAuthenticatedUserHasAnyOf.  
 \* Para adicionar uma role existente pelo código, é necessário ir à classe BaseRoles e adicionar o role pretendido.  
 \*  
 \* Em caso de adicionar permissões para visualizar a opção, deve ser feita no MainMenuUI.  
 \*/  
@UseCaseController  
public class AddUserController {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final UserManagementService userSvc = AuthzRegistry.userService();  
  
 /\*\*  
 \* Get existing RoleTypes available to the user.  
 \*  
 \* @return a list of RoleTypes  
 \*/  
 public Role[] getRoleTypes() {  
 return BaseRoles.nonUserValues();  
 }  
  
 public SystemUser addUser(final String username, final String password, final String firstName, final String lastName, final String email, final Set<Role> roles, final Calendar createdOn) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.ADMIN, BaseRoles.MANAGER);  
  
 return userSvc.registerNewUser(username, password, firstName, lastName, email, roles, createdOn);  
 }  
  
 public SystemUser addUser(final String username, final String password, final String firstName, final String lastName, final String email, final Set<Role> roles) {  
 return addUser(username, password, firstName, lastName, email, roles, CurrentTimeCalendars.now());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\AddUserController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\DeactivateUserController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.usermanagement.application;  
  
import com.fasterxml.jackson.databind.ser.Serializers;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserManagementService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
  
/\*\*  
 \*  
 \* @author Fernando  
 \* Edited by João Cruz  
 \*  
 \* Código destinado a desativar um utilizador  
 \*/  
@UseCaseController  
public class DeactivateUserController {  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final UserManagementService userSvc = AuthzRegistry.userService();  
  
 public Iterable<SystemUser> activeUsers() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.ADMIN, BaseRoles.MANAGER);  
  
 return userSvc.activeUsers();  
 }  
  
 public SystemUser deactivateUser(final SystemUser user) {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.ADMIN, BaseRoles.MANAGER);  
  
 return userSvc.deactivateUser(user);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\DeactivateUserController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\ListUsersController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.usermanagement.application;  
  
import java.util.Optional;  
  
import com.fasterxml.jackson.databind.ser.Serializers;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.UserManagementService;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
  
/\*\*  
 \*  
 \* @author losa  
 \* Edited by João Cruz  
 \*  
 \* Classe destinada a listar os utilizadores  
 \*/  
@UseCaseController  
public class ListUsersController{  
  
 private final AuthorizationService authz = AuthzRegistry.authorizationService();  
 private final UserManagementService userSvc = AuthzRegistry.userService();  
  
 public Iterable<SystemUser> allUsers() {  
 authz.ensureAuthenticatedUserHasAnyOf(BaseRoles.ADMIN, BaseRoles.MANAGER);  
  
 return userSvc.allUsers();  
 }  
  
 public Optional<SystemUser> find(final Username u) {  
 return userSvc.userOfIdentity(u);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\ListUsersController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\eventhandlers\AddUserOnSignupAcceptedController.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.usermanagement.application.eventhandlers;  
  
import eapli.base.clientusermanagement.domain.events.NewUserRegisteredFromSignupEvent;  
import eapli.base.clientusermanagement.domain.events.SignupAcceptedEvent;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.base.usermanagement.domain.UserBuilderHelper;  
import eapli.framework.application.UseCaseController;  
import eapli.framework.domain.events.DomainEvent;  
import eapli.framework.domain.repositories.ConcurrencyException;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
import eapli.framework.infrastructure.authz.domain.repositories.UserRepository;  
import eapli.framework.infrastructure.pubsub.EventPublisher;  
import eapli.framework.infrastructure.pubsub.impl.inprocess.service.InProcessPubSub;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*/  
@UseCaseController  
/\* package \*/ class AddUserOnSignupAcceptedController {  
 private final UserRepository userRepository = PersistenceContext.repositories().users();  
 private final EventPublisher dispatcher = InProcessPubSub.publisher();  
  
 /\*\*  
 \*  
 \* @param theSignupRequest  
 \* @return  
 \* @throws ConcurrencyException  
 \* @throws IntegrityViolationException  
 \*/  
 public SystemUser addUser(final SignupAcceptedEvent theSignupRequest) {  
  
 final SystemUserBuilder userBuilder = UserBuilderHelper.builder();  
 userBuilder.withUsername(theSignupRequest.username()).withPassword(theSignupRequest.password()).withName(theSignupRequest.name()).withEmail(theSignupRequest.email()).withRoles(BaseRoles.CLIENT\_USER);  
 final SystemUser newUser = userRepository.save(userBuilder.build());  
  
 // notify interested parties  
 final DomainEvent event = new NewUserRegisteredFromSignupEvent(theSignupRequest.mecanographicNumber(), newUser.username());  
 dispatcher.publish(event);  
  
 return newUser;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\eventhandlers\AddUserOnSignupAcceptedController.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\eventhandlers\SignupAcceptedWatchDog.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.usermanagement.application.eventhandlers;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import eapli.base.clientusermanagement.domain.events.SignupAcceptedEvent;  
import eapli.framework.domain.events.DomainEvent;  
import eapli.framework.domain.repositories.IntegrityViolationException;  
import eapli.framework.infrastructure.pubsub.EventHandler;  
  
/\*\*  
 \* Classe destinada de registar um novo utilizador quando o seu pedido de registo é aceite.  
 \*\*/  
  
public class SignupAcceptedWatchDog implements EventHandler {  
 private static final Logger LOGGER = LoggerFactory.getLogger(SignupAcceptedWatchDog.class);  
  
 @Override  
 public void onEvent(final DomainEvent domainevent) {  
 assert domainevent instanceof SignupAcceptedEvent;  
  
 final SignupAcceptedEvent event = (SignupAcceptedEvent) domainevent;  
  
 final AddUserOnSignupAcceptedController controller = new AddUserOnSignupAcceptedController();  
 try {  
 controller.addUser(event);  
 } catch (final IntegrityViolationException e) {  
 // TODO provably should send some warning email...  
 LOGGER.error("Unable to register new user on signup event", e);  
 }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\application\eventhandlers\SignupAcceptedWatchDog.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\domain\BasePasswordPolicy.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.usermanagement.domain;  
  
import eapli.framework.infrastructure.authz.application.PasswordPolicy;  
import eapli.framework.strings.util.StringPredicates;  
  
/\*\*  
 \* Enforces that passwords must be at least 6 characters long and have at least  
 \* one digit and one capital letter.  
 \*  
 \* <p>  
 \* look into  
 \* https://documentation.cpanel.net/display/CKB/How+to+Determine+Password+Strength  
 \* for example rules of password strength  
 \*  
 \* @author Paulo Gandra de Sousa 24/05/2019  
 \*  
 \* Classe que define a politica de passwords, diz também se a password é forte ou fraca.  
 \*/  
public class BasePasswordPolicy implements PasswordPolicy {  
  
 /\*  
 \* (non-Javadoc)  
 \*  
 \* @see eapli.framework.infrastructure.authz.domain.model.PasswordPolicy#  
 \* meetsRequeriments(java.lang.String)  
 \*/  
 @Override  
 public boolean isSatisfiedBy(final String rawPassword) {  
 // sanity check  
 if (StringPredicates.isNullOrEmpty(rawPassword)) {  
 return false;  
 }  
  
 // at least 6 characters long  
 if (rawPassword.length() < 6) {  
 return false;  
 }  
  
 // at least one digit  
 if (!StringPredicates.containsDigit(rawPassword)) {  
 return false;  
 }  
  
 // at least one capital letter  
 return StringPredicates.containsCapital(rawPassword);  
 }  
  
 /\*\*  
 \* Check how strong a password is. just for demo purposes.  
 \*  
 \* <p>  
 \* look into  
 \* https://documentation.cpanel.net/display/CKB/How+to+Determine+Password+Strength  
 \* for example rules of password strength  
 \*  
 \* @param rawPassword  
 \* the string to check  
 \*  
 \* @return how strong a password is  
 \*/  
 public PasswordStrength strength(final String rawPassword) {  
 PasswordStrength passwordStrength = PasswordStrength.WEAK;  
 if (rawPassword.length() >= 12 || (rawPassword.length() >= 8  
 && StringPredicates.containsAny(rawPassword, "$#!%&?"))) {  
 passwordStrength = PasswordStrength.EXCELENT;  
 } else if (rawPassword.length() >= 8) {  
 passwordStrength = PasswordStrength.GOOD;  
 } else if (rawPassword.length() >= 6) {  
 passwordStrength = PasswordStrength.WEAK;  
 }  
 return passwordStrength;  
 }  
  
 public enum PasswordStrength {  
 WEAK, GOOD, EXCELENT,  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\domain\BasePasswordPolicy.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\domain\BaseRoles.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.usermanagement.domain;  
  
import eapli.framework.infrastructure.authz.domain.model.Role;  
  
/\*\*  
 \* @author Paulo Gandra Sousa  
 \*  
 \* Classe destinada a listar os utilizadores e criar novos roles.  
 \*/  
public final class BaseRoles {  
 /\*\*  
 \* USER - Poderá servir caso o utilizador não for nenhum dos roles do Projeto Integrador  
 \*/  
 public static final Role CLIENT\_USER = Role.valueOf("USER");  
 /\*\*  
 \* Base Administrator (Developer)  
 \*/  
 public static final Role ADMIN = Role.valueOf("ADMIN");  
 /\*\*  
 \*  
 \*/  
  
 // Roles do Projeto Integrador  
 public static final Role STUDENT = Role.valueOf("STUDENT");  
 public static final Role TEACHER = Role.valueOf("TEACHER");  
 public static final Role MANAGER = Role.valueOf("MANAGER");  
  
 /\*\*  
 \* get available role types for adding new users  
 \*  
 \* @return  
 \*/  
 public static Role[] nonUserValues() {  
 return new Role[] { STUDENT, TEACHER, MANAGER };  
 }  
  
 public boolean isCollaborator(final Role role) {  
 return role != CLIENT\_USER;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\domain\BaseRoles.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\domain\UserBuilderHelper.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.usermanagement.domain;  
  
import eapli.base.clientusermanagement.domain.SignupRequestBuilder;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
import eapli.framework.util.Utility;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra de Sousa 27/05/2019  
 \*  
 \* Classe que permite criar um utilizador.  
 \*/  
@Utility  
public class UserBuilderHelper {  
 private UserBuilderHelper() {  
 // ensure utility  
 }  
  
 public static SystemUserBuilder builder() {  
 return new SystemUserBuilder(new BasePasswordPolicy(), new PlainTextEncoder());  
 }  
  
 public static SignupRequestBuilder signupBuilder() {  
 return new SignupRequestBuilder(new BasePasswordPolicy(), new PlainTextEncoder());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\java\eapli\base\usermanagement\domain\UserBuilderHelper.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\main\resources\ecafeteria.sample.properties

# To change this template, choose Tools | Templates  
# and open the template in the editor.  
  
persistence.persistenceUnit=eapli.base  
#persistence.repositoryFactory=eapli.base.persistence.jpa.JpaRepositoryFactory  
persistence.repositoryFactory=eapli.base.persistence.inmemory.InMemoryRepositoryFactory  
ui.menu.layout=horizontal  
HighCaloriesDishLimit=300

### [File Ends] sem4pi-22-23-61-master\base.core\src\main\resources\ecafeteria.sample.properties

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\Classe\Domain\ClasseTest.java

package base.Classe.Domain;  
  
import eapli.base.Classe.domain.\*;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import org.junit.jupiter.api.Test;  
  
import java.time.LocalDate;  
import java.time.LocalTime;  
  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
public class ClasseTest {  
  
  
 final SystemUserBuilder userBuilder = new SystemUserBuilder(new BasePasswordPolicy(), new PlainTextEncoder());  
 final SystemUser newUser = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT, BaseRoles.TEACHER)  
 .build();  
  
 private final Classe\_Title title = new Classe\_Title("Class Title");  
 private final Classe\_Start\_Date start\_date = new Classe\_Start\_Date(LocalDate.now());  
 private final Classe\_Finish\_Date finish\_date = new Classe\_Finish\_Date(LocalDate.now().plusDays(7));  
 private final Classe\_Start\_Time start\_time = new Classe\_Start\_Time(LocalTime.of(9, 0));  
 private final Classe\_Finish\_Time finish\_time = new Classe\_Finish\_Time(LocalTime.of(10, 30));  
 private final eapli.base.Classe.domain.DayOfWeek day = new eapli.base.Classe.domain.DayOfWeek(1);  
  
 private final Acronym acronym = new Acronym("XYZ");  
  
  
 @Test  
 void ensureClasseMustHaveStartDate() {  
 assertThrows(IllegalArgumentException.class, () -> new Classe(title, null, finish\_date, start\_time, finish\_time, day, acronym));  
 }  
  
 @Test  
 void ensureClasseMustHaveFinishDate() {  
 assertThrows(IllegalArgumentException.class, () -> new Classe(title, start\_date, null, start\_time, finish\_time, day, acronym));  
 }  
  
 @Test  
 void ensureClasseMustHaveStartTime() {  
 assertThrows(IllegalArgumentException.class, () -> new Classe(title, start\_date, finish\_date, null, finish\_time, day, acronym));  
 }  
  
 @Test  
 void ensureClasseMustHaveFinishTime() {  
 assertThrows(IllegalArgumentException.class, () -> new Classe(title, start\_date, finish\_date, start\_time, null, day, acronym));  
 }  
  
 @Test  
 void ensureClasseMustHaveTitle() {  
 assertThrows(IllegalArgumentException.class, () -> {  
 new Classe(null, start\_date, finish\_date, start\_time, finish\_time, day, acronym);  
 });  
 }  
 @Test  
 void ensureClasseMustHaveAcronym() {  
 assertThrows(IllegalArgumentException.class, () -> {  
 new Classe(title, start\_date, finish\_date, start\_time, finish\_time, day, null);  
 });  
 }  
  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\Classe\Domain\ClasseTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\SharedBoard\Domain\SharedBoardTest.java

package eapli.base.SharedBoard.Domain;  
  
import eapli.base.SharedBoard.domain.NumberofColumns;  
import eapli.base.SharedBoard.domain.NumberofRows;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.framework.infrastructure.authz.domain.model.\*;  
  
import org.junit.Before;  
import org.junit.Test;  
  
import java.util.Collections;  
import java.util.Set;  
  
import static junit.framework.TestCase.assertEquals;  
import static junit.framework.TestCase.assertFalse;  
  
  
public class SharedBoardTest {  
  
 private SystemUser owner;  
 private SystemUser currentUser;  
 private SystemUser newUser;  
 private Shared\_Board\_Title title;  
 private NumberofRows numberOfRows;  
 private NumberofColumns numberOfColumns;  
 private SharedBoard sharedBoard;  
  
 @Before  
 public void setUp() {  
 Set<Role> roles = Collections.emptySet();  
 // owner = new SystemUser();  
 currentUser = owner;  
 // newUser = new SystemUser();  
 title = new Shared\_Board\_Title("Sample Title");  
 numberOfRows = new NumberofRows(5);  
 numberOfColumns = new NumberofColumns(5);  
  
 sharedBoard = new SharedBoard(owner, title, numberOfRows, numberOfColumns);  
 }  
  
 @Test  
 public void testShareBoard() {  
 sharedBoard.shareBoard(currentUser, newUser, SharedBoard.AccessType.WRITE);  
 assertEquals(SharedBoard.AccessType.WRITE, sharedBoard.getAccessType(newUser));  
 }  
  
 @Test(expected = IllegalStateException.class)  
 public void testShareBoardNotOwner() {  
 // SystemUser notOwnerUser = new SystemUser(Username.valueOf("notOwner"), new Password("password"), new Name("Not Owner Name"), new EmailAddress("notowner@mail.com"), roles);  
 // sharedBoard.shareBoard(notOwnerUser, newUser, SharedBoard.AccessType.WRITE);  
 }  
  
 @Test  
 public void testArchiveBoard() {  
 sharedBoard.shareBoard(currentUser, newUser, SharedBoard.AccessType.WRITE);  
 sharedBoard.archiveBoard();  
 assertTrue(sharedBoard.isArchived());  
 assertEquals(SharedBoard.AccessType.READ, sharedBoard.getAccessType(newUser));  
 }  
  
 private void assertTrue(boolean archived) {  
  
 }  
  
 @Test  
 public void testSameAs() {  
 SharedBoard otherBoard = new SharedBoard(owner, title, numberOfRows, numberOfColumns);  
 assertTrue(sharedBoard.sameAs(otherBoard));  
 }  
  
 @Test  
 public void testNotSameAs() {  
 Shared\_Board\_Title otherTitle = new Shared\_Board\_Title("Other Title");  
 SharedBoard otherBoard = new SharedBoard(owner, otherTitle, numberOfRows, numberOfColumns);  
 assertFalse(sharedBoard.sameAs(otherBoard));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\SharedBoard\Domain\SharedBoardTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\SharedBoard\aplication\SharedBoardServiceTest.java

package eapli.base.SharedBoard.aplication;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.base.persistence.impl.inmemory.InMemorySharedBoardRepository;  
import eapli.base.systemUserManagement.SystemUserRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.authz.domain.model.Password;  
import org.junit.Before;  
import org.junit.Test;  
  
import java.util.Optional;  
  
import static junit.framework.TestCase.assertEquals;  
  
public class SharedBoardServiceTest {  
 private SharedBoardService sharedBoardService;  
 private SystemUserRepository systemUserRepository;  
 private InMemorySharedBoardRepository sharedBoardRepository;  
  
 @Before  
 public void setUp() {  
 sharedBoardRepository = new InMemorySharedBoardRepository();  
 sharedBoardService = new SharedBoardService(sharedBoardRepository, systemUserRepository);  
 }  
/\*  
 @Test  
 public void createSharedBoard() {  
 Username username = Username.valueOf("user");  
 Optional<Password> password = Password.encodedAndValid("adeus", 0xf, 4141);  
 SystemUser myUser = new SystemUser();  
  
 sharedBoardService.createSharedBoard(myUser, "Titulo teste", 3, 3);  
  
  
 Iterable<SharedBoard> sharedBoards = sharedBoardRepository.findSharedBoardsByOwner(myUser);  
 int count = 0;  
 for (SharedBoard board : sharedBoards) {  
 if (board.getTitle().getTitle().equals("TestTitle")) {  
 count++;  
 }  
 }  
 assertEquals(1, count);  
 }  
  
 \*/  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\SharedBoard\aplication\SharedBoardServiceTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\exammanagement\domain\CreateExamControllerTest.java

package eapli.base.exammanagement.domain;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.exammanagement.application.CreateExamController;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.persistence.impl.inmemory.InMemoryExamRepository;  
import org.junit.jupiter.api.BeforeEach;  
import org.junit.jupiter.api.Test;  
  
import java.sql.Date;  
import java.sql.Time;  
  
import static org.junit.jupiter.api.Assertions.assertEquals;  
import static org.junit.jupiter.api.Assertions.assertTrue;  
  
class CreateExamControllerTest {  
 private CreateExamController createExamController;  
 private ExamRepository examRepository;  
  
 @BeforeEach  
 void setUp() {  
 examRepository = new InMemoryExamRepository(); // InMemoryExamRepository para testar  
 createExamController = new CreateExamController();  
 createExamController.setExamRepository(examRepository);  
 }  
  
 @Test  
 void createExamValidDataReturnsNewExam() {  
 // Arrange  
 String examName = "Exame Fisica";  
 Date examDate = Date.valueOf("2023/06/18");  
 Time openTime = Time.valueOf("09:00");  
 Time closeTime = Time.valueOf("12:00");  
 Course course = new Course();  
  
 // Act  
 Exam result = createExamController.createExam(examName, examDate, openTime, closeTime, course);  
  
 // Assert  
 assertTrue(examRepository.contains(result)); // Verifica se guardou no ExamRepository  
 assertEquals(examName, result.getExamTitle().toString());  
 assertEquals(examDate.toString(), result.getExamDate().toString());  
 assertEquals(openTime.toString(), result.getExamTime().getOpenDate().toString());  
 assertEquals(closeTime.toString(), result.getExamTime().getCloseDate().toString());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\exammanagement\domain\CreateExamControllerTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\exammanagement\domain\ExamTest.java

package eapli.base.exammanagement.domain;  
  
import eapli.base.Course.Domain.\*;  
import org.junit.jupiter.api.Test;  
  
import java.text.ParseException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
import static org.junit.jupiter.api.Assertions.assertThrows;  
  
public class ExamTest {  
  
 Date date=new Date(System.currentTimeMillis());  
 SimpleDateFormat simpleDateFormat = new SimpleDateFormat("hh:mm");  
 Date time = simpleDateFormat.parse("10:30");  
 Date time2 = simpleDateFormat.parse("11:30");  
  
 public ExamTest() throws ParseException {  
 }  
  
 @Test  
 void ensureExamMustHaveTime() {  
 System.out.println("must have exam time");  
  
 assertThrows(IllegalArgumentException.class, () -> new Exam(null, new ExamDate(date), new Course(new Course\_Name("INF"), new Maximum\_Number\_Of\_Students(100), new Minimum\_Number\_Of\_Students(10), new Small\_Textual\_Description("Desc"))));  
 }  
  
 @Test  
 void ensureExamMustHaveDate() {  
 System.out.println("must have exam date");  
  
 assertThrows(IllegalArgumentException.class, () -> new Exam(new ExamTime(time,time2), null, new Course(new Course\_Name("INF"), new Maximum\_Number\_Of\_Students(100), new Minimum\_Number\_Of\_Students(10), new Small\_Textual\_Description("Desc"))));  
 }  
  
 @Test  
 void ensureExamMustHaveCourse() {  
 System.out.println("must have exam course");  
  
 assertThrows(IllegalArgumentException.class, () -> new Exam(new ExamTime(time,time2), new ExamDate(date), null));  
 }  
  
 @Test  
 void ensureExamCreation() {  
 assertThrows(IllegalArgumentException.class, () -> new Exam());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\exammanagement\domain\ExamTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\meetingmanagement\aplication\CancelMeetingTest.java

package eapli.base.meetingmanagement.aplication;  
  
import eapli.base.meetingmanagement.domain.\*;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import org.junit.jupiter.api.Test;  
  
import static org.junit.jupiter.api.Assertions.assertEquals;  
  
public class CancelMeetingTest {  
  
 @Test  
 void cancelMeeting\_MeetingStatusChangedToCanceled() {  
 System.out.println("cancelMeeting - Meeting status changed to CANCELED");  
  
 MeetingTitle meetingTitle = new MeetingTitle("Meeting 1");  
 SystemUser systemUser = new SystemUser();  
 MeetingDuration meetingDuration = new MeetingDuration(60);  
 MeetingDate meetingDate = new MeetingDate("2023-06-18");  
 MeetingTime meetingTime = new MeetingTime("14:30");  
 Meeting meeting = new Meeting(meetingTitle, systemUser, meetingDuration, meetingDate, meetingTime);  
  
  
 meeting.cancelMeeting();  
  
  
 assertEquals(MeetingStatus.CANCELED, meeting.getStatus());  
 }  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 private void assertTrue(boolean contains) {  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\meetingmanagement\aplication\CancelMeetingTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\meetingmanagement\domain\InviteTest.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.base.SharedBoard.domain.PostIt;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.app.backoffice.console.presentation.meeting.AcceptRefuseInvitesAction;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.meetingmanagement.application.AcceptRefuseInvitesController;  
import eapli.base.meetingmanagement.application.AcceptRefuseInvitesService;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.base.usermanagement.domain.UserBuilderHelper;  
import eapli.framework.infrastructure.authz.application.AuthorizationService;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.application.PasswordPolicy;  
import eapli.framework.infrastructure.authz.application.UserSession;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
//import eapli.base.core.usermanagement.domain.BasePasswordPolicy;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import org.junit.Before;  
import org.junit.jupiter.api.Assertions;  
import org.junit.jupiter.api.Test;  
  
import java.text.ParseException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
import static org.junit.jupiter.api.Assertions.assertThrows;  
  
  
  
public class InviteTest {  
  
 AcceptRefuseInvitesService service = new AcceptRefuseInvitesService(AuthzRegistry.authorizationService(), PersistenceContext.repositories().inviteRepository());  
  
 Date date=new Date(System.currentTimeMillis());  
 SimpleDateFormat simpleDateFormat = new SimpleDateFormat("hh:mm");  
 Date time = simpleDateFormat.parse("10:30");  
  
  
 final SystemUserBuilder userBuilder = new SystemUserBuilder(new BasePasswordPolicy(), new PlainTextEncoder());  
 final SystemUser newUser = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 final SystemUser newUser2 = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 final SystemUser newUser3 = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 final SystemUser newUser4 = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 final SystemUser newUser5 = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
 Meeting meeting = new Meeting(new MeetingTitle("meeting title"), newUser, new MeetingDuration(30), new MeetingDate(date), new MeetingTime(time));  
 Invite invite1 = new Invite(newUser, newUser2, meeting);  
 Invite invite2 = new Invite(newUser, newUser3, meeting);  
  
 @Before  
 public void setUp() {  
 service.acceptInvite(invite1, newUser2);  
 service.refuseInvite(invite2);  
 }  
  
 public InviteTest() throws ParseException {  
 }  
  
 @Test  
 void ensureInviteMustHaveSender() {  
 System.out.println("must have invite sender");  
 assertThrows(IllegalArgumentException.class, () -> new Invite(null, newUser, meeting));  
 }  
  
 @Test  
 void ensureInviteMustHaveReceiver() {  
 System.out.println("must have invite receiver");  
 assertThrows(IllegalArgumentException.class, () -> new Invite(newUser, null, meeting));  
 }  
  
 @Test  
 void ensureInviteMustHaveMeeting() {  
 System.out.println("must have meeting");  
 assertThrows(IllegalArgumentException.class, () -> new Invite(newUser, newUser2, null));  
 }  
  
 @Test  
 void EnsureInviteStateChangesToAcceptedWhenUserAccepts(){  
 Assertions.assertEquals(invite1.getState(),InviteState.ACCEPTED);  
 }  
  
 @Test  
 void EnsureMeetingContainsUserWhenInviteIsAccepted(){  
 Assertions.assertTrue(meeting.getParticipants().contains(newUser2));  
 }  
  
 @Test  
 void EnsureInviteStateChangesToRefusedWhenUserRefuses(){  
 Assertions.assertEquals(invite2.getState(),InviteState.REJECTED);  
 }  
  
 @Test  
 void EnsureMeetingDoesNotContainsUserWhenInviteIsRefused(){  
 Assertions.assertFalse(meeting.getParticipants().contains(newUser3));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\meetingmanagement\domain\InviteTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\meetingmanagement\domain\MeetingTest.java

package eapli.base.meetingmanagement.domain;  
  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.base.usermanagement.domain.UserBuilderHelper;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
import org.junit.jupiter.api.Test;  
  
import java.text.ParseException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
import static org.junit.jupiter.api.Assertions.assertThrows;  
  
public class MeetingTest {  
  
 Date date=new Date(System.currentTimeMillis());  
 SimpleDateFormat simpleDateFormat = new SimpleDateFormat("hh:mm");  
 Date time = simpleDateFormat.parse("10:30");  
  
 final SystemUserBuilder userBuilder = new SystemUserBuilder(new BasePasswordPolicy(), new PlainTextEncoder());  
 final SystemUser newUser = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 public MeetingTest() throws ParseException {  
 }  
  
 @Test  
 void ensureMeetingMustHaveDuration() {  
 System.out.println("must have meeting duration");  
  
 assertThrows(IllegalArgumentException.class, () -> new Meeting(new MeetingTitle("meeting title"), newUser, null, new MeetingDate(date), new MeetingTime(time)));  
 }  
  
 @Test  
 void ensureMeetingMustHaveDate() {  
 System.out.println("must have exam date");  
  
 assertThrows(IllegalArgumentException.class, () -> new Meeting(new MeetingTitle("meeting title"), newUser, new MeetingDuration(30), null, new MeetingTime(time)));  
 }  
  
 @Test  
 void ensureMeetingMustHaveTime() {  
 System.out.println("must have exam course");  
  
 assertThrows(IllegalArgumentException.class, () -> new Meeting(new MeetingTitle("meeting title"), newUser, new MeetingDuration(30), new MeetingDate(date), null));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\meetingmanagement\domain\MeetingTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\sharedboardmanagement\domain\CreatePostItTest.java

package eapli.base.sharedboardmanagement.domain;  
  
import eapli.base.SharedBoard.aplication.CreateBoardPostItController;  
import eapli.base.SharedBoard.domain.\*;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
import org.junit.Before;  
import org.junit.jupiter.api.Assertions;  
import org.junit.jupiter.api.Test;  
  
public class CreatePostItTest {  
  
 private final CreateBoardPostItController controller = new CreateBoardPostItController(  
 PersistenceContext.repositories().sharedBoardRepository());  
  
 final SystemUserBuilder userBuilder = new SystemUserBuilder(new BasePasswordPolicy(), new PlainTextEncoder());  
 final SystemUser newUser = userBuilder  
 .withUsername("systemUserTest")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 final SystemUser newUser2 = userBuilder  
 .withUsername("systemUserTest\_update")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 SharedBoard sharedBoard = new SharedBoard(newUser, new Shared\_Board\_Title("title"), new NumberofRows(10), new NumberofColumns(10));  
 @Before  
 public void setUp() {  
 sharedBoard.shareBoard(newUser, newUser2, SharedBoard.AccessType.READ);  
  
 sharedBoard.getBoard().insertCell(1,1,new PostIt("post-it\_1", newUser));  
 }  
  
  
 @Test  
 void EnsureUserWithReadPermissionCantWrite(){  
 Assertions.assertFalse(controller.getBoardAccessType(sharedBoard, newUser2));  
 }  
  
 @Test  
 void EnsurePostItIsNotAddedIfTheDesiredPositionIsOccupied(){  
 Assertions.assertFalse(sharedBoard.getBoard().insertCell(1,1,new PostIt("post-it\_2", newUser)));  
 }  
  
 @Test  
 void EnsurePostItIsAddedIfTheDesiredPositionIsFree(){  
 Assertions.assertTrue(sharedBoard.getBoard().insertCell(1,2,new PostIt("post-it\_3", newUser)));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\sharedboardmanagement\domain\CreatePostItTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\eapli\base\sharedboardmanagement\domain\UpdatePostItTest.java

package eapli.base.sharedboardmanagement.domain;  
  
import eapli.base.SharedBoard.aplication.CreateBoardPostItController;  
import eapli.base.SharedBoard.aplication.UpdateBoardPostItController;  
import eapli.base.SharedBoard.domain.\*;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.usermanagement.domain.BasePasswordPolicy;  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
import org.junit.Before;  
import org.junit.jupiter.api.Assertions;  
import org.junit.jupiter.api.Test;  
  
public class UpdatePostItTest {  
  
 private final UpdateBoardPostItController controller = new UpdateBoardPostItController(  
 PersistenceContext.repositories().sharedBoardRepository());  
  
 final SystemUserBuilder userBuilder = new SystemUserBuilder(new BasePasswordPolicy(), new PlainTextEncoder());  
 final SystemUser newUser = userBuilder  
 .withUsername("systemUserTest\_update")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 final SystemUser newUser2 = userBuilder  
 .withUsername("systemUserTest\_update")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 final SystemUser newUser3 = userBuilder  
 .withUsername("systemUserTest\_update")  
 .withPassword("password")  
 .withName("SystemUser", "Test")  
 .withEmail("systemusertest@email.org")  
 .withRoles(BaseRoles.STUDENT)  
 .build();  
  
 SharedBoard sharedBoard = new SharedBoard(newUser, new Shared\_Board\_Title("title"), new NumberofRows(10), new NumberofColumns(10));  
 @Before  
 public void setUp() {  
 sharedBoard.shareBoard(newUser, newUser2, SharedBoard.AccessType.WRITE);  
 sharedBoard.shareBoard(newUser, newUser3, SharedBoard.AccessType.READ);  
  
 sharedBoard.getBoard().insertCell(1,1,new PostIt("post-it\_1", newUser));  
 sharedBoard.getBoard().insertCell(1,2,new PostIt("post-it\_2", newUser2));  
 }  
  
  
 @Test  
 void EnsureUserWithReadPermissionCantWrite(){  
 Assertions.assertFalse(controller.getBoardAccessType(sharedBoard, newUser3));  
 }  
  
 @Test  
 void EnsurePostItIsNotUpdatedIfTheOriginalPositionIsNull(){  
 Assertions.assertFalse(sharedBoard.getBoard().updateCell(2,2,new PostIt("post-it",newUser)));  
 }  
  
 @Test  
 void EnsurePostItIsNotUpdatedIfUserIsNotTheAuthorOfTheOriginalPostIt(){  
 Assertions.assertFalse(sharedBoard.getBoard().updateCell(1,1,new PostIt("post-it",newUser2)));  
 }  
  
 @Test  
 void EnsurePostItIsUpdatedIfAllConditionsAreMet(){  
 Assertions.assertFalse(sharedBoard.getBoard().updateCell(1,1,new PostIt("post-it",newUser)));  
 }  
  
 @Test  
 void EnsurePostItIsNotMovedIfTheOriginalPositionIsNull(){  
 Assertions.assertEquals(sharedBoard.getBoard().moveCell(2,2,2,1,newUser),3);  
 }  
  
 @Test  
 void EnsurePostItIsNotMovedIfTheNewPositionIsOccupied(){  
 Assertions.assertEquals(sharedBoard.getBoard().moveCell(1,1,1,2,newUser),2);  
 }  
  
 @Test  
 void EnsurePostItIsNotMovedIfUserIsNotTheAuthorOfTheOriginalPostIt(){  
 Assertions.assertEquals(sharedBoard.getBoard().moveCell(1,1,2,2,newUser2),1);  
 }  
  
 @Test  
 void EnsurePostItIsMovedIfAllConditionsAreMet(){  
 Assertions.assertEquals(sharedBoard.getBoard().moveCell(1,1,3,3,newUser),0);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\eapli\base\sharedboardmanagement\domain\UpdatePostItTest.java

### [File Begins] sem4pi-22-23-61-master\base.core\src\test\java\eapli\base\clientusermanagement\domain\ClientUserTest.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.clientusermanagement.domain;  
  
import static org.junit.Assert.assertFalse;  
import static org.junit.Assert.assertTrue;  
  
import java.util.HashSet;  
import java.util.Set;  
  
import org.junit.Test;  
  
import eapli.base.usermanagement.domain.BaseRoles;  
import eapli.framework.infrastructure.authz.domain.model.NilPasswordPolicy;  
import eapli.framework.infrastructure.authz.domain.model.PlainTextEncoder;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.SystemUserBuilder;  
  
/\*\*  
 \* Created by Nuno Bettencourt [NMB] on 03/04/16.  
 \*/  
public class ClientUserTest {  
  
 private final String aMecanographicNumber = "abc";  
 private final String anotherMecanographicNumber = "xyz";  
  
 public static SystemUser dummyUser(final String username, final Role... roles) {  
 // should we load from spring context?  
 final SystemUserBuilder userBuilder = new SystemUserBuilder(new NilPasswordPolicy(), new PlainTextEncoder());  
 return userBuilder.with(username, "duMMy1", "dummy", "dummy", "a@b.ro").withRoles(roles).build();  
 }  
  
 private SystemUser getNewDummyUser() {  
 return dummyUser("dummy", BaseRoles.ADMIN);  
 }  
  
 private SystemUser getNewDummyUserTwo() {  
 return dummyUser("dummy-two", BaseRoles.ADMIN);  
 }  
  
 @Test  
 public void ensureClientUserEqualsPassesForTheSameMecanographicNumber() throws Exception {  
  
 final ClientUser aClientUser = new ClientUserBuilder().withMecanographicNumber("DUMMY")  
 .withSystemUser(getNewDummyUser()).build();  
  
 final ClientUser anotherClientUser = new ClientUserBuilder().withMecanographicNumber("DUMMY")  
 .withSystemUser(getNewDummyUser()).build();  
  
 final boolean expected = aClientUser.equals(anotherClientUser);  
  
 assertTrue(expected);  
 }  
  
 @Test  
 public void ensureClientUserEqualsFailsForDifferenteMecanographicNumber() throws Exception {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.MANAGER);  
  
 final ClientUser aClientUser = new ClientUserBuilder().withMecanographicNumber(aMecanographicNumber)  
 .withSystemUser(getNewDummyUser()).build();  
  
 final ClientUser anotherClientUser = new ClientUserBuilder()  
 .withMecanographicNumber(anotherMecanographicNumber).withSystemUser(getNewDummyUser()).build();  
  
 final boolean expected = aClientUser.equals(anotherClientUser);  
  
 assertFalse(expected);  
 }  
  
 @Test  
 public void ensureClientUserEqualsAreTheSameForTheSameInstance() throws Exception {  
 final ClientUser aClientUser = new ClientUser();  
  
 final boolean expected = aClientUser.equals(aClientUser);  
  
 assertTrue(expected);  
 }  
  
 @Test  
 public void ensureClientUserEqualsFailsForDifferenteObjectTypes() throws Exception {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.TEACHER);  
  
 final ClientUser aClientUser = new ClientUserBuilder().withMecanographicNumber("DUMMY")  
 .withSystemUser(getNewDummyUser()).build();  
  
 final boolean expected = aClientUser.equals(getNewDummyUser());  
  
 assertFalse(expected);  
 }  
  
 @Test  
 public void ensureClientUserIsTheSameAsItsInstance() throws Exception {  
 final ClientUser aClientUser = new ClientUserBuilder().withMecanographicNumber("DUMMY")  
 .withSystemUser(getNewDummyUser()).build();  
  
 final boolean expected = aClientUser.sameAs(aClientUser);  
  
 assertTrue(expected);  
 }  
  
 @Test  
 public void ensureTwoClientUserWithDifferentMecanographicNumbersAreNotTheSame() throws Exception {  
 final Set<Role> roles = new HashSet<>();  
 roles.add(BaseRoles.STUDENT);  
 final ClientUser aClientUser = new ClientUserBuilder().withMecanographicNumber(aMecanographicNumber)  
 .withSystemUser(getNewDummyUser()).build();  
  
 final ClientUser anotherClientUser = new ClientUserBuilder()  
 .withMecanographicNumber(anotherMecanographicNumber).withSystemUser(getNewDummyUser()).build();  
  
 final boolean expected = aClientUser.sameAs(anotherClientUser);  
  
 assertFalse(expected);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.core\src\test\java\eapli\base\clientusermanagement\domain\ClientUserTest.java

### [File Begins] sem4pi-22-23-61-master\base.infrastructure.application\pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
  
 <artifactId>base.infrastructure.application</artifactId>  
 <packaging>jar</packaging>  
 <name>base.infrastructure.application</name>  
  
 <properties></properties>  
  
 <dependencies>  
 </dependencies>  
  
</project>

### [File Ends] sem4pi-22-23-61-master\base.infrastructure.application\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\AppSettings.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base;  
  
import java.io.FileNotFoundException;  
import java.io.IOException;  
import java.io.InputStream;  
import java.util.HashMap;  
import java.util.Map;  
import java.util.Properties;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
/\*\*  
 \* the application settings.  
 \*  
 \* @author Paulo Gandra Sousa  
 \*/  
public class AppSettings {  
 private static final Logger LOGGER = LoggerFactory.getLogger(AppSettings.class);  
  
 private static final String PROPERTIES\_RESOURCE = "application.properties";  
 private static final String REPOSITORY\_FACTORY\_KEY = "persistence.repositoryFactory";  
 private static final String UI\_MENU\_LAYOUT\_KEY = "ui.menu.layout";  
 private static final String PERSISTENCE\_UNIT\_KEY = "persistence.persistenceUnit";  
 private static final String SCHEMA\_GENERATION\_KEY = "javax.persistence.schema-generation.database.action";  
 private static final String HIGH\_CALORIES\_DISH\_LIMIT = "HighCaloriesDishLimit";  
  
 private final Properties applicationProperties = new Properties();  
  
 public AppSettings() {  
 loadProperties();  
 }  
  
 private void loadProperties() {  
 try (InputStream propertiesStream = this.getClass().getClassLoader()  
 .getResourceAsStream(PROPERTIES\_RESOURCE)) {  
 if (propertiesStream != null) {  
 this.applicationProperties.load(propertiesStream);  
 } else {  
 throw new FileNotFoundException(  
 "property file '" + PROPERTIES\_RESOURCE + "' not found in the classpath");  
 }  
 } catch (final IOException exio) {  
 setDefaultProperties();  
  
 LOGGER.warn("Loading default properties", exio);  
 }  
 }  
  
 private void setDefaultProperties() {  
 this.applicationProperties.setProperty(REPOSITORY\_FACTORY\_KEY,  
 "eapli.base.persistence.jpa.JpaRepositoryFactory");  
 this.applicationProperties.setProperty(UI\_MENU\_LAYOUT\_KEY, "horizontal");  
 this.applicationProperties.setProperty(PERSISTENCE\_UNIT\_KEY, "eapli"  
 + ".base");  
 this.applicationProperties.setProperty(HIGH\_CALORIES\_DISH\_LIMIT, "300");  
 }  
  
 public Boolean isMenuLayoutHorizontal() {  
 return "horizontal"  
 .equalsIgnoreCase(this.applicationProperties.getProperty(UI\_MENU\_LAYOUT\_KEY));  
 }  
  
 public String getPersistenceUnitName() {  
 return this.applicationProperties.getProperty(PERSISTENCE\_UNIT\_KEY);  
 }  
  
 public String getRepositoryFactory() {  
 return this.applicationProperties.getProperty(REPOSITORY\_FACTORY\_KEY);  
 }  
  
 public Integer getHighCaloriesDishLimit() {  
 return Integer.valueOf(this.applicationProperties.getProperty(HIGH\_CALORIES\_DISH\_LIMIT));  
 }  
  
 @SuppressWarnings({ "rawtypes", "unchecked" })  
 public Map getExtendedPersistenceProperties() {  
 final Map ret = new HashMap();  
 ret.put(SCHEMA\_GENERATION\_KEY,  
 this.applicationProperties.getProperty(SCHEMA\_GENERATION\_KEY));  
 return ret;  
 }  
  
 public String getProperty(final String prop) {  
 return this.applicationProperties.getProperty(prop);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\AppSettings.java

### [File Begins] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\Application.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base;  
  
/\*\*  
 \* A "global" static class with the application registry of well known objects  
 \*  
 \* @author Paulo Gandra Sousa  
 \*  
 \*/  
public class Application {  
  
 public static final String VERSION = "1.4.0";  
 public static final String COPYRIGHT = "(C) 2016 - 2021, ISEP's Professors of EAPLI";  
  
 private static final AppSettings SETTINGS = new AppSettings();  
  
 public static AppSettings settings() {  
 return SETTINGS;  
 }  
  
 private Application() {  
 // private visibility to ensure singleton & utility  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\Application.java

### [File Begins] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\infrastructure\authz\AuthenticationCredentialHandler.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.infrastructure.authz;  
  
import eapli.framework.infrastructure.authz.application.Authenticator;  
import eapli.framework.infrastructure.authz.application.AuthzRegistry;  
import eapli.framework.infrastructure.authz.domain.model.Role;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra de Sousa 2022.11.24  
 \*  
 \*/  
public class AuthenticationCredentialHandler implements CredentialHandler {  
  
 private final Authenticator authenticationService = AuthzRegistry.authenticationService();  
  
 @Override  
 public boolean authenticated(String username, String password, Role onlyWithThis) {  
 return authenticationService.authenticate(username, password, onlyWithThis).isPresent();  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\infrastructure\authz\AuthenticationCredentialHandler.java

### [File Begins] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\infrastructure\authz\CredentialHandler.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy  
 \* of this software and associated documentation files (the "Software"), to deal  
 \* in the Software without restriction, including without limitation the rights  
 \* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
 \* copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in  
 \* all copies or substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
 \* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
 \* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
 \* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
 \* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
 \* SOFTWARE.  
 \*/  
package eapli.base.infrastructure.authz;  
  
import eapli.framework.infrastructure.authz.domain.model.Role;  
  
/\*\*  
 \*  
 \* @author Paulo Gandra de Sousa 2022.11.24  
 \*  
 \*/  
public interface CredentialHandler {  
 boolean authenticated(String username, String password, Role onlyWithThis);  
}

### [File Ends] sem4pi-22-23-61-master\base.infrastructure.application\src\main\java\eapli\base\infrastructure\authz\CredentialHandler.java

### [File Begins] sem4pi-22-23-61-master\base.infrastructure.application\src\main\resources\application.properties.sample

#####################################################  
##  
## eCafeteria  
##  
#####################################################  
  
persistence.persistenceUnit=eapli.eCafeteriaPU  
  
#persistence.repositoryFactory=eapli.ecafeteria.persistence.jpa.JpaRepositoryFactory  
persistence.repositoryFactory=eapli.ecafeteria.persistence.inmemory.InMemoryRepositoryFactory  
  
ui.menu.layout=horizontal  
  
HighCaloriesDishLimit=300  
  
UseEventfulControllers=true

### [File Ends] sem4pi-22-23-61-master\base.infrastructure.application\src\main\resources\application.properties.sample

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
   
 <parent>  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <relativePath>../</relativePath>  
 <version>1.4.0-SNAPSHOT</version>  
 </parent>  
   
 <artifactId>base.persistence.impl</artifactId>  
 <packaging>jar</packaging>  
  
 <name>base.persistence.impl</name>  
   
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.core</artifactId>  
 <version>${project.parent.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>eapli</groupId>  
 <artifactId>base.bootstrappers</artifactId>  
 <version>${project.parent.version}</version>  
 <type>jar</type>  
 </dependency>  
 </dependencies>  
</project>

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\pom.xml

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryClientUserRepository.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.inmemory;  
  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.clientusermanagement.domain.ClientUser;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.inmemory.InMemoryDomainRepository;  
  
import java.util.Optional;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
public class InMemoryClientUserRepository  
 extends InMemoryDomainRepository<ClientUser, MechanographicNumber>  
 implements ClientUserRepository {  
  
 static {  
 InMemoryInitializer.init();  
 }  
  
 @Override  
 public Optional<ClientUser> findByUsername(final Username name) {  
 return matchOne(e -> e.user().username().equals(name));  
 }  
  
 @Override  
 public Optional<ClientUser> findByMechanographicNumber(final MechanographicNumber number) {  
 return Optional.of(data().get(number));  
 }  
  
 @Override  
 public Iterable<ClientUser> findAllActive() {  
 return match(e -> e.user().isActive());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryClientUserRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryExamRepository.java

package eapli.base.persistence.impl.inmemory;  
  
import eapli.base.Course.Domain.Course;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.framework.infrastructure.repositories.impl.inmemory.InMemoryDomainRepository;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class InMemoryExamRepository extends InMemoryDomainRepository<Exam, Long> implements ExamRepository {  
 @Override  
 public boolean contains(Exam exam) {  
 return super.contains(exam);  
 }  
  
 @Override  
 public Exam save(Exam entity) {  
 return super.save(entity);  
 }  
  
  
 @Override  
 public Iterable<Exam> findAll() {  
 return super.findAll();  
 }  
  
 public List<Exam> findByTitle(String title) {  
 List<Exam> exams = new ArrayList<>();  
 for (Exam exam : findAll()) {  
 if (exam.getExamTitle().toString().equalsIgnoreCase(title)) {  
 exams.add(exam);  
 }  
 }  
 return exams;  
 }  
  
 @Override  
 public List<Exam> findStudentFutureExams(Student student) {  
 return null;  
 }  
  
 @Override  
 public Iterable<Exam> findByCourse(Course course) {  
 return null;  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryExamRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryInitializer.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.inmemory;  
  
import eapli.base.infrastructure.bootstrapers.BaseBootstrapper;  
  
final class InMemoryInitializer {  
  
 private static class LazyHolder {  
 private static final InMemoryInitializer INSTANCE = new InMemoryInitializer();  
  
 private LazyHolder() {  
 }  
 }  
  
 private InMemoryInitializer() {  
 // to ensure some default test data is available, specially when using  
 // in memory persistence  
 new BaseBootstrapper().execute();  
 }  
  
 private void initialize() {  
 // nothing to do; data has already been initialized in the singleton  
 // constructor.  
 }  
  
 public static void init() {  
 LazyHolder.INSTANCE.initialize();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryInitializer.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryInviteRepository.java

package eapli.base.persistence.impl.inmemory;  
  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.inmemory.InMemoryDomainRepository;  
  
public class InMemoryInviteRepository extends InMemoryDomainRepository<Invite, Long>  
 implements InviteRepository {  
  
 static {  
 InMemoryInitializer.init();  
 }  
// @Override  
// public Iterable<Invite> findInvitesByReceiverUsername(Username username) {  
// return match(e -> e.getReceiver().username().equals(username));  
// }  
  
 @Override  
 public Iterable<Invite> findInvitesByMeeting(Meeting meeting) {  
 return match(e -> e.getMeeting().getMeetingTitle().getTitle().equals(meeting.getMeetingTitle().getTitle()));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryInviteRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryRepositoryFactory.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.inmemory;  
  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.infrastructure.bootstrapers.BaseBootstrapper;  
import eapli.base.infrastructure.persistence.RepositoryFactory;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.repositories.UserRepository;  
import eapli.framework.infrastructure.authz.repositories.impl.inmemory.InMemoryUserRepository;  
  
/\*\*  
 \*  
 \* Created by nuno on 20/03/16.  
 \*/  
public abstract class InMemoryRepositoryFactory implements RepositoryFactory {  
  
 static {  
 // only needed because of the in memory persistence  
 new BaseBootstrapper().execute();  
 }  
  
 @Override  
 public UserRepository users(final TransactionalContext tx) {  
 return new InMemoryUserRepository();  
 }  
  
 @Override  
 public UserRepository users() {  
 return users(null);  
 }  
  
 @Override  
 public ClientUserRepository clientUsers(final TransactionalContext tx) {  
  
 return new InMemoryClientUserRepository();  
 }  
  
 @Override  
 public ClientUserRepository clientUsers() {  
 return clientUsers(null);  
 }  
  
 @Override  
 public SignupRequestRepository signupRequests() {  
 return signupRequests(null);  
 }  
  
 @Override  
 public ExamRepository examRepository() {  
 return null;  
 }  
  
 @Override  
 public InviteRepository inviteRepository() {  
 return new InMemoryInviteRepository();  
 }  
  
 @Override  
 public SignupRequestRepository signupRequests(final TransactionalContext tx) {  
 return new InMemorySignupRequestRepository();  
 }  
  
 @Override  
 public SharedBoardRepository sharedBoardRepository(){  
 return new InMemorySharedBoardRepository();  
 }  
  
 @Override  
 public TransactionalContext newTransactionalContext() {  
 // in memory does not support transactions...  
 return null;  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemoryRepositoryFactory.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemorySharedBoardRepository.java

package eapli.base.persistence.impl.inmemory;  
  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.repositories.impl.inmemory.InMemoryDomainRepository;  
  
public class InMemorySharedBoardRepository extends InMemoryDomainRepository<SharedBoard, Shared\_Board\_Title> implements SharedBoardRepository {  
  
 static {  
 InMemoryInitializer.init();  
 }  
  
  
 @Override  
 public Iterable<SharedBoard> findSharedBoardsByOwner(SystemUser owner) {  
 return match(e -> e.getOwner().username().toString().equals(owner.username().toString()));  
 }  
  
 @Override  
 public Iterable<SharedBoard> findByBoardId(Shared\_Board\_Title title) {  
 return match(e -> e.getTitle().getTitle().equals(title.getTitle()));  
 }  
  
 @Override  
 public Iterable<SharedBoard> findBoardsBySystemUser(SystemUser systemUser) {  
 return match(e -> e.getSharedUsers().containsKey(systemUser));  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemorySharedBoardRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemorySignupRequestRepository.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.inmemory;  
  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.inmemory.InMemoryDomainRepository;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
public class InMemorySignupRequestRepository extends  
 InMemoryDomainRepository<SignupRequest, Username> implements SignupRequestRepository {  
  
 static {  
 InMemoryInitializer.init();  
 }  
  
 @Override  
 public Iterable<SignupRequest> pendingSignupRequests() {  
 throw new UnsupportedOperationException("Not supported yet.");  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\inmemory\InMemorySignupRequestRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\BaseJpaReportingRepositoryBase.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaTransactionalContext;  
  
/\*\*  
 \* a base class for all reporting repositories to use the same persistence unit  
 \*  
 \* @param <T>  
 \* @param <K>  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*/  
/\* package \*/ class BaseJpaReportingRepositoryBase extends JpaTransactionalContext {  
  
 BaseJpaReportingRepositoryBase() {  
 super(Application.settings().getPersistenceUnitName(),  
 Application.settings().getExtendedPersistenceProperties());  
 }  
  
 BaseJpaReportingRepositoryBase(final String persistenceUnitName) {  
 super(persistenceUnitName, Application.settings().getExtendedPersistenceProperties());  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\BaseJpaReportingRepositoryBase.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\BasepaRepositoryBase.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaTransactionalRepository;  
  
/\*\*  
 \* a base class for all transactional repositories to use the same persistence  
 \* unit  
 \*  
 \* @param <T>  
 \* @param <K>  
 \*  
 \* @author Paulo Gandra de Sousa  
 \*/  
/\* package \*/ class BasepaRepositoryBase<T, K, I>  
 extends JpaTransactionalRepository<T, K, I> {  
  
 BasepaRepositoryBase(final String persistenceUnitName, final String identityFieldName) {  
 super(persistenceUnitName, Application.settings().getExtendedPersistenceProperties(),  
 identityFieldName);  
 }  
  
 BasepaRepositoryBase(final String identityFieldName) {  
 super(Application.settings().getPersistenceUnitName(),  
 Application.settings().getExtendedPersistenceProperties(), identityFieldName);  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\BasepaRepositoryBase.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaClassRepository.java

package eapli.base.persistence.impl.jpa;  
import eapli.base.Application;  
import eapli.base.Classe.domain.Classe;  
import eapli.base.Classe.domain.Classe\_Title;  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.TypedQuery;  
import java.util.List;  
import java.util.Spliterator;  
import java.util.function.Consumer;  
  
public class JpaClassRepository extends JpaAutoTxRepository<Classe, Classe\_Title, Classe\_Title> implements ClassRepository {  
  
 public JpaClassRepository(final TransactionalContext autoTx) {  
 super(autoTx, "title");  
 }  
  
 public JpaClassRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(), "title");  
 }  
  
 @Override  
 public List<Classe> findClassesByTeacher(String teacherAcronym) {  
 final TypedQuery<Classe> query = createQuery("SELECT c FROM Class c WHERE c.teacherAcronym = :acronym", Classe.class);  
 query.setParameter("acronym", teacherAcronym);  
 return query.getResultList();  
 }  
 @Override  
 public Iterable<Classe> findAll() {  
 return this.createQuery("SELECT c FROM Classe c", Classe.class).getResultList();  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaClassRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaClientUserRepository.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.clientusermanagement.domain.ClientUser;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import java.util.HashMap;  
import java.util.Map;  
import java.util.Optional;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
class JpaClientUserRepository  
 extends JpaAutoTxRepository<ClientUser, MechanographicNumber, MechanographicNumber>  
 implements ClientUserRepository {  
  
 public JpaClientUserRepository(final TransactionalContext autoTx) {  
 super(autoTx, "mecanographicNumber");  
 }  
  
 public JpaClientUserRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(),  
 "mecanographicNumber");  
 }  
  
 @Override  
 public Optional<ClientUser> findByUsername(final Username name) {  
 final Map<String, Object> params = new HashMap<>();  
 params.put("name", name);  
 return matchOne("e.systemUser.username=:name", params);  
 }  
  
 @Override  
 public Optional<ClientUser> findByMechanographicNumber(final MechanographicNumber number) {  
 final Map<String, Object> params = new HashMap<>();  
 params.put("number", number);  
 return matchOne("e.mecanographicNumber=:number", params);  
 }  
  
 @Override  
 public Iterable<ClientUser> findAllActive() {  
 return match("e.systemUser.active = true");  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaClientUserRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaCourseRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.Course.Domain.Course;  
import eapli.base.Course.Domain.Course\_ID;  
import eapli.base.Course.Domain.Course\_Name;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.TypedQuery;  
  
public class JpaCourseRepository extends JpaAutoTxRepository<Course, Long, Course\_Name> implements CourseRepository {  
  
 public JpaCourseRepository(final TransactionalContext autoTx) {  
 super(autoTx, "id");  
 }  
  
 public JpaCourseRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(),  
 "id");  
 }  
  
 @Override  
 public Iterable<Course> findCourseByName(String courseName) {  
 final TypedQuery<Course> q = createQuery(  
 "SELECT e FROM Course e WHERE e.courseName.value = :courseName", Course.class);  
 q.setParameter("courseName", courseName);  
 return q.getResultList();  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaCourseRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaExamRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.Course.Domain.Course;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.base.exammanagement.domain.Exam;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaBaseRepository;  
  
import javax.persistence.\*;  
import java.util.List;  
  
class JpaExamRepository extends JpaBaseRepository<Exam, Long, Long> implements ExamRepository {  
  
 public JpaExamRepository() {  
 super("id");  
 }  
  
 @Override  
 public Iterable<Exam> findByCourse(final Course course) {  
 final TypedQuery<Exam> q = entityManager().createQuery(  
 "SELECT e FROM Exam e WHERE e.examCourse.courseName.value = :course", Exam.class);  
 q.setParameter("course", course.getCourseName().getValue());  
 return q.getResultList();  
 }  
  
 @Override  
 public Iterable<Exam> findStudentFutureExams(final Student student) {  
 final TypedQuery<Exam> q = entityManager().createQuery(  
 "SELECT e FROM Exam e WHERE :student MEMBER OF e.examCourse.students AND e.examDate.examDate > CURRENT\_DATE", Exam.class);  
 q.setParameter("student", student);  
 return q.getResultList();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaExamRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaExtraClassRepository.java

package eapli.base.persistence.impl.jpa;  
import eapli.base.Application;  
import eapli.base.Classe.domain.Classe;  
import eapli.base.Classe.domain.Classe\_Title;  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.base.ExtraClasse.domain.ExtraClasse;  
import eapli.base.ExtraClasse.domain.ExtraClasse\_Title;  
import eapli.base.ExtraClasse.repository.ExtraClasseRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.TypedQuery;  
import java.util.List;  
import java.util.Spliterator;  
import java.util.function.Consumer;  
  
public class JpaExtraClassRepository extends JpaAutoTxRepository<ExtraClasse, ExtraClasse\_Title, ExtraClasse\_Title> implements ExtraClasseRepository {  
  
 public JpaExtraClassRepository(final TransactionalContext autoTx) {  
 super(autoTx, "title");  
 }  
  
 public JpaExtraClassRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(), "title");  
 }  
  
 @Override  
 public List<ExtraClasse> findExtraClassesByTeacher(String teacherAcronym) {  
 final TypedQuery<ExtraClasse> query = createQuery("SELECT c FROM ExtraClass c WHERE c.teacherAcronym = :acronym", ExtraClasse.class);  
 query.setParameter("acronym", teacherAcronym);  
 return query.getResultList();  
 }  
 @Override  
 public Iterable<ExtraClasse> findAll() {  
 return this.createQuery("SELECT c FROM ExtraClasse c", ExtraClasse.class).getResultList();  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaExtraClassRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaInviteRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.meetingmanagement.domain.Invite;  
import eapli.base.meetingmanagement.domain.InviteState;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.\*;  
import java.util.Optional;  
  
public class JpaInviteRepository extends JpaAutoTxRepository<Invite, Long, Long> implements InviteRepository {  
  
 public JpaInviteRepository(final TransactionalContext autoTx) {  
 super(autoTx, "id");  
 }  
  
 public JpaInviteRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(),  
 "id");  
 }  
  
// @Override  
// public Iterable<Invite> findInvitesByReceiverUsername(Username username) {  
// final TypedQuery<Invite> q = entityManager().createQuery(  
// "SELECT e FROM Invite e WHERE e.receiver.username.value = :username AND e.state= :state", Invite.class);  
// q.setParameter("username", username.toString());  
// q.setParameter("state", InviteState.PENDING);  
// return q.getResultList();  
// }  
  
 @Override  
 public Iterable<Invite> findInvitesByMeeting(Meeting meeting) {  
 final TypedQuery<Invite> q = entityManager().createQuery(  
 "SELECT e FROM Invite e WHERE e.meeting.meetingTitle.title = :meeting", Invite.class);  
 q.setParameter("meeting", meeting.getMeetingTitle().getTitle());  
 return q.getResultList();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaInviteRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaMeetingRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.Classe.domain.Classe;  
import eapli.base.meetingmanagement.domain.Meeting;  
import eapli.base.meetingmanagement.domain.MeetingTitle;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.TypedQuery;  
  
class JpaMeetingRepository extends JpaAutoTxRepository<Meeting, Long, Long> implements MeetingRepository {  
  
 public JpaMeetingRepository(final TransactionalContext autoTx) {  
 super(autoTx, "id");  
 }  
  
 public JpaMeetingRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(),  
 "id");  
 }  
  
 @Override  
 public Meeting findByMeetingById(MeetingTitle meetingTitle) {  
 final TypedQuery<Meeting> query = entityManager().createQuery("SELECT c FROM Meeting c WHERE c.meetingTitle.title = :title", Meeting.class);  
 query.setParameter("title", meetingTitle.getTitle());  
 return query.getSingleResult();  
 }  
  
 @Override  
 public Iterable<Meeting> findAll() {  
 return this.createQuery("SELECT m FROM Meeting m", Meeting.class).getResultList();  
 }  
  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaMeetingRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaRepositoryFactory.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.Classe.repository.ClassRepository;  
import eapli.base.Course.Repository.CourseRepository;  
import eapli.base.Enrollment.Repository.EnrollmentRepository;  
import eapli.base.ExtraClasse.aplication.ExtraClassController;  
import eapli.base.ExtraClasse.repository.ExtraClasseRepository;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.base.clientusermanagement.repositories.ClientUserRepository;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.base.exammanagement.repository.ExamRepository;  
import eapli.base.infrastructure.persistence.PersistenceContext;  
import eapli.base.infrastructure.persistence.RepositoryFactory;  
import eapli.base.meetingmanagement.repository.InviteRepository;  
import eapli.base.meetingmanagement.repository.MeetingRepository;  
import eapli.base.systemUserManagement.SystemUserRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.repositories.UserRepository;  
import eapli.framework.infrastructure.authz.repositories.impl.jpa.JpaAutoTxUserRepository;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
/\*\*  
 \*  
 \* Created by nuno on 21/03/16.  
 \*/  
public class JpaRepositoryFactory implements RepositoryFactory {  
  
  
 @Override  
 public UserRepository users(final TransactionalContext autoTx) {  
 return new JpaAutoTxUserRepository(autoTx);  
 }  
  
 @Override  
 public UserRepository users() {  
 return new JpaAutoTxUserRepository(Application.settings().getPersistenceUnitName(),  
 Application.settings().getExtendedPersistenceProperties());  
 }  
  
 @Override  
 public JpaClientUserRepository clientUsers(final TransactionalContext autoTx) {  
 return new JpaClientUserRepository(autoTx);  
 }  
  
 @Override  
 public JpaClientUserRepository clientUsers() {  
 return new JpaClientUserRepository(Application.settings().getPersistenceUnitName());  
 }  
  
 @Override  
 public SignupRequestRepository signupRequests(final TransactionalContext autoTx) {  
 return new JpaSignupRequestRepository(autoTx);  
 }  
  
 @Override  
 public SignupRequestRepository signupRequests() {  
 return new JpaSignupRequestRepository(Application.settings().getPersistenceUnitName());  
 }  
  
 @Override  
 public ExamRepository examRepository() {  
 return new JpaExamRepository();  
 }  
  
 @Override  
 public CourseRepository courseRepository() {  
 return new JpaCourseRepository(Application.settings().getPersistenceUnitName());  
 }  
  
 @Override  
 public MeetingRepository meetingRepository() {  
 return new JpaMeetingRepository(Application.settings().getPersistenceUnitName());  
 }  
  
 @Override  
 public InviteRepository inviteRepository() {  
 return new JpaInviteRepository(PersistenceContext.repositories()  
 .newTransactionalContext());  
 }  
  
 @Override  
 public ClassRepository classRepository() {  
 return new JpaClassRepository(PersistenceContext.repositories()  
 .newTransactionalContext());  
 }  
  
 @Override  
 public ExtraClasseRepository extraClassRepostory() {  
 return new JpaExtraClassRepository(PersistenceContext.repositories()  
 .newTransactionalContext());  
 }  
  
 @Override  
 public TeacherRepository teacherRepository() {  
 return new JpaTeacherRepository(PersistenceContext.repositories().newTransactionalContext());  
 }  
  
 @Override  
 public StudentRepository studentRepository() {  
 return new JpaStudentRepository(PersistenceContext.repositories().newTransactionalContext());  
 }  
  
 @Override  
 public ClientUserRepository clientUserRepository() {  
 return null;  
 }  
  
 @Override  
 public SharedBoardRepository sharedBoardRepository() {  
 return new JpaSharedBoardRepository(PersistenceContext.repositories().newTransactionalContext());  
 }  
  
 @Override  
 public EnrollmentRepository enrollmentRepository() {  
 return null;  
 }  
  
 @Override  
 public SystemUserRepository systemUserRepository() {  
 return new JpaSystemUserRepository(PersistenceContext.repositories().newTransactionalContext());  
 }  
  
 @Override  
 public TransactionalContext newTransactionalContext() {  
 return JpaAutoTxRepository.buildTransactionalContext(Application.settings().getPersistenceUnitName(),  
 Application.settings().getExtendedPersistenceProperties());  
 }  
  
  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaRepositoryFactory.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaSharedBoardRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.SharedBoard.aplication.SharedBoardArchiveController;  
import eapli.base.SharedBoard.domain.SharedBoard;  
import eapli.base.SharedBoard.domain.Shared\_Board\_Title;  
import eapli.base.SharedBoard.repository.SharedBoardRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.\*;  
  
class JpaSharedBoardRepository extends JpaAutoTxRepository<SharedBoard, Shared\_Board\_Title, Shared\_Board\_Title> implements SharedBoardRepository {  
  
 EntityManagerFactory emf = Persistence.createEntityManagerFactory(Application.settings().getPersistenceUnitName());  
 EntityManager em = emf.createEntityManager();  
  
 public JpaSharedBoardRepository(final TransactionalContext autoTx) {  
 super(autoTx, "id");  
 }  
  
 public JpaSharedBoardRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(), "id");  
 }  
  
 public Iterable<SharedBoard> findSharedBoardsByOwner(SystemUser owner) {  
 final TypedQuery<SharedBoard> q = entityManager().createQuery(  
 "SELECT sb FROM SharedBoard sb WHERE sb.owner = :owner", SharedBoard.class);  
 q.setParameter("owner", owner);  
 return q.getResultList();  
 }  
 public Iterable<SharedBoard> findByBoardId(Shared\_Board\_Title title){  
 final TypedQuery<SharedBoard> q = entityManager().createQuery(  
 "SELECT sb FROM SharedBoard sb WHERE sb.title = :title", SharedBoard.class);  
 q.setParameter("title", title);  
 return q.getResultList();  
 }  
  
 public Iterable<SharedBoard> findBoardsBySystemUser(SystemUser systemUser){  
 final TypedQuery<SharedBoard> q = entityManager().createQuery(  
 "SELECT e FROM SharedBoard e WHERE KEY(e.sharedUsers) = :systemUser", SharedBoard.class);  
 q.setParameter("systemUser", systemUser);  
 return q.getResultList();  
 }  
  
 @Override  
 public Iterable<SharedBoard> findAll() {  
 return this.createQuery("SELECT s FROM SharedBoard s", SharedBoard.class).getResultList();  
 }  
  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaSharedBoardRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaSignupRequestRepository.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.clientusermanagement.domain.SignupRequest;  
import eapli.base.clientusermanagement.repositories.SignupRequestRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
/\*\*  
 \*  
 \* @author Jorge Santos ajs@isep.ipp.pt 02/04/2016  
 \*/  
class JpaSignupRequestRepository extends JpaAutoTxRepository<SignupRequest, Username, Username>  
 implements SignupRequestRepository {  
  
 public JpaSignupRequestRepository(final TransactionalContext autoTx) {  
 super(autoTx, "username");  
 }  
  
 public JpaSignupRequestRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(), "username");  
 }  
  
 @Override  
 public Iterable<SignupRequest> pendingSignupRequests() {  
 return match(  
 "e.approvalStatus=eapli.base.clientusermanagement.domain"  
 + ".ApprovalStatus.PENDING");  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaSignupRequestRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaStudentRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.Student\_Teacher.Student.Repository.StudentRepository;  
import eapli.base.Student\_Teacher.Student.domain.MechanographicNumber;  
import eapli.base.Student\_Teacher.Student.domain.Student;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaBaseRepository;  
import org.hibernate.jpa.internal.util.PersistenceUtilHelper;  
  
import javax.persistence.TypedQuery;  
import java.util.HashMap;  
import java.util.Map;  
import java.util.Optional;  
  
public class JpaStudentRepository extends JpaAutoTxRepository<Student,MechanographicNumber, MechanographicNumber>  
 implements StudentRepository {  
  
 public JpaStudentRepository(final TransactionalContext autoTx) {  
 super(autoTx, "mechanographic number");  
 }  
  
 public JpaStudentRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(),  
 "mechanographic number");  
 }  
  
 @Override  
 public Iterable<Student> findStudentByMechanographicNumberReturnList(String mechanographicNumber) {  
 return null;  
 }  
  
 @Override  
 public Student findStudentByMechanographicNumber(String mechanographicNumber) {  
 final TypedQuery<Student> q = createQuery(  
 "SELECT e FROM Student e WHERE e.mechanographicNumber = :mechanographicNumber", Student.class);  
 q.setParameter("mechanographicNumber", mechanographicNumber);  
 return q.getSingleResult();  
 }  
  
 @Override  
 public Student findStudentBySystemUser(SystemUser systemUser) {  
 final TypedQuery<Student> q = createQuery(  
 "SELECT e FROM Student e WHERE e.systemUser.username.value = :systemUser", Student.class);  
 q.setParameter("systemUser", systemUser.username().toString());  
 return q.getSingleResult();  
 }  
  
 @Override  
 public Optional<Student> findByUsername(final Username name) {  
 final Map<String, Object> params = new HashMap<>();  
 params.put("name", name);  
 return matchOne("e.systemUser.username=:name", params);  
 }  
  
 @Override  
 public Optional<Student> findByMechanographicNumber(final MechanographicNumber number) {  
 final Map<String, Object> params = new HashMap<>();  
 params.put("number", number);  
 return matchOne("e.mechanographicNumber=:number", params);  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaStudentRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaSystemUserRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.systemUserManagement.SystemUserRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.\*;  
import java.util.List;  
  
public class JpaSystemUserRepository extends JpaAutoTxRepository<SystemUser, Username, Username> implements SystemUserRepository {  
  
 public JpaSystemUserRepository(final TransactionalContext autoTx) {  
 super(autoTx, "username");  
 }  
  
 public JpaSystemUserRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(),  
 "username");  
 }  
  
 @Override  
 public Iterable<SystemUser> findByUsername(Username name) {  
 final TypedQuery<SystemUser> q = entityManager().createQuery(  
 "SELECT e FROM SystemUser e WHERE e.username.value = :username", SystemUser.class);  
 q.setParameter("username", name.toString());  
 return q.getResultList();  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaSystemUserRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaTeacherRepository.java

package eapli.base.persistence.impl.jpa;  
  
import eapli.base.Application;  
import eapli.base.Student\_Teacher.Teacher.Domain.Acronym;  
import eapli.base.Student\_Teacher.Teacher.Domain.Teacher;  
import eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository;  
import eapli.framework.domain.repositories.TransactionalContext;  
import eapli.framework.infrastructure.authz.domain.model.SystemUser;  
import eapli.framework.infrastructure.authz.domain.model.Username;  
import eapli.framework.infrastructure.repositories.impl.jpa.JpaAutoTxRepository;  
  
import javax.persistence.TypedQuery;  
import java.util.List;  
import java.util.Optional;  
  
public class JpaTeacherRepository  
 extends JpaAutoTxRepository<Teacher, Long, Long>  
 implements TeacherRepository {  
  
 public JpaTeacherRepository(final TransactionalContext autoTx) {  
 super(autoTx, "id");  
 }  
  
 public JpaTeacherRepository(final String puname) {  
 super(puname, Application.settings().getExtendedPersistenceProperties(), "id");  
 }  
  
 @Override  
 public List<Teacher> findTeacherByAcronym(Acronym acronym) {  
 final TypedQuery<Teacher> query = entityManager().createQuery(  
 "SELECT c FROM Teacher c WHERE c.acronym = :acronym",  
 Teacher.class);  
 query.setParameter("acronym", acronym);  
 return query.getResultList();  
 }  
  
 @Override  
 public Optional<Teacher> ofIdentity(Username id) {  
 return Optional.empty();  
 }  
  
 @Override  
 public Teacher findTeacherBySystemUser(SystemUser systemUser) {  
 final TypedQuery<Teacher> q = entityManager().createQuery(  
 "SELECT e FROM Teacher e WHERE e.systemUser.username.value = :systemUser",  
 Teacher.class);  
 q.setParameter("systemUser", systemUser.username().toString());  
 return q.getSingleResult();  
 }  
  
 @Override  
 public void deleteOfIdentity(Username entityId) {  
  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\JpaTeacherRepository.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\PersistenceSettings.java

/\*  
 \* Copyright (c) 2013-2023 the original author or authors.  
 \*  
 \* MIT License  
 \*  
 \* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and  
 \* associated documentation files (the "Software"), to deal in the Software without restriction,  
 \* including without limitation the rights to use, copy, modify, merge, publish, distribute,  
 \* sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is  
 \* furnished to do so, subject to the following conditions:  
 \*  
 \* The above copyright notice and this permission notice shall be included in all copies or  
 \* substantial portions of the Software.  
 \*  
 \* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT  
 \* NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
 \* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
 \* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
 \* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
 \*/  
package eapli.base.persistence.impl.jpa;  
  
/\*\*  
 \*  
 \* Created by nuno on 21/03/16.  
 \*/  
class PersistenceSettings {  
  
 public static final String PERSISTENCE\_UNIT\_NAME = "eapli.base";  
  
 private PersistenceSettings() {  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\java\eapli\base\persistence\impl\jpa\PersistenceSettings.java

### [File Begins] sem4pi-22-23-61-master\base.persistence.impl\src\main\resources\META-INF\persistence.xml

<?xml version="1.0" encoding="UTF-8"?>  
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence">  
 <persistence-unit name="eapli.base">  
 <!--  
 <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>  
 -->  
 <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>  
  
 <class>eapli.framework.infrastructure.authz.domain.model.SystemUser</class>  
 <class>eapli.framework.infrastructure.authz.domain.model.Role</class>  
 <class>eapli.framework.infrastructure.authz.domain.model.RoleSet</class>  
 <class>eapli.framework.infrastructure.authz.domain.model.RoleAssignment</class>  
 <class>eapli.base.clientusermanagement.domain.ClientUser</class>  
 <class>eapli.base.clientusermanagement.domain.SignupRequest</class>  
 <class>eapli.base.Application</class>  
 <class>eapli.framework.infrastructure.authz.domain.repositories.UserRepository</class>  
 <class>eapli.base.core.systemUserManagement.SystemUserRepository</class>  
 <class>eapli.base.AppSettings</class>  
 <class>eapli.base.infrastructure.authz.AuthenticationCredentialHandler</class>  
 <class>eapli.base.infrastructure.authz.CredentialHandler</class>  
 <class>eapli.base.Classe.domain.Classe</class>  
 <class>eapli.base.Course.Domain.Course</class>  
 <class>eapli.base.Enrollment.Domain.Enrollment</class>  
 <class>eapli.base.ExtraClasse.domain.ExtraClasse</class>  
 <class>eapli.base.exammanagement.domain.Exam</class>  
 <class>eapli.base.Manager.Manager</class>  
 <class>eapli.base.meetingmanagement.domain.Invite</class>  
 <class>eapli.base.meetingmanagement.domain.Meeting</class>  
 <class>eapli.base.SharedBoard.domain.SharedBoard</class>  
 <class>eapli.base.SharedBoard.repository.SharedBoardRepository</class>  
 <class>eapli.base.Student\_Teacher.Student.domain.Student</class>  
 <class>eapli.base.Student\_Teacher.Teacher.Domain.Teacher</class>  
 <class>eapli.base.Student\_Teacher.Teacher.Repository.TeacherRepository</class>  
 <class>eapli.base.usermanagement.domain.BaseRoles</class>  
  
  
  
 <properties>  
 <!--<property name="javax.persistence.jdbc.url" value="jdbc:h2:tcp://localhost/~/base"/>-->  
 <property name="javax.persistence.jdbc.url" value="jdbc:h2:tcp://vsgate-s2.dei.isep.ipp.pt:10751/base;MV\_STORE=FALSE;AUTO\_SERVER=true;"/>  
  
 <property name="javax.persistence.jdbc.user" value="sa"/>  
 <property name="javax.persistence.jdbc.password" value="eapli"/>  
 <property name="javax.persistence.jdbc.driver" value="org.h2.Driver"/>  
 <!-- <property name="javax.persistence.schema-generation.database.action" value="drop-and-create"/> -->  
 <!-- <property name="eclipselink.logging.level" value="FINE"/> -->  
 <!-- <property name = "hibernate.show\_sql" value = "true" /> -->  
 <property name="hibernate.dialect" value="org.hibernate.dialect.H2Dialect"/>  
  
 </properties>  
 </persistence-unit>  
</persistence>

### [File Ends] sem4pi-22-23-61-master\base.persistence.impl\src\main\resources\META-INF\persistence.xml

### [File Begins] sem4pi-22-23-61-master\base.rcomp\Makefile

APPS\_FOLDERS=tcp-chat tcp-cli-srv udp-chat udp-cli-srv http-server-form-file-upload http-server-ajax-voting http-server-chat SSL  
  
all:   
 $(foreach AF,$(APPS\_FOLDERS),(cd $(AF); make );)  
  
clean:  
 $(foreach AF,$(APPS\_FOLDERS),(cd $(AF); make clean);)

### [File Ends] sem4pi-22-23-61-master\base.rcomp\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\Makefile

APPS\_FOLDERS=check-TLS https-server-ajax-voting tcp-cli-srv  
  
all:   
 $(foreach AF,$(APPS\_FOLDERS),(cd $(AF); make );)  
  
clean:  
 $(foreach AF,$(APPS\_FOLDERS),(cd $(AF); make clean);)

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\check-TLS\CheckServerTLS.java

import java.io.\*;  
import java.net.\*;  
  
import javax.net.ssl.\*;  
import javax.security.cert.X509Certificate;  
  
/\*\*  
 \*  
 \* @author asc@isep.ipp.pt  
 \*/  
public class CheckServerTLS {  
 static private SSLSocket sock;  
 static private InetAddress serverIP;  
 static private int serverPort;  
 static private DataOutputStream sOut;  
 static private DataInputStream sIn;  
  
 // THE NUMBER OF VOTES TO CAST ON THE FIRST CANDIDATE  
 static private final int VOTES\_TO\_CAST = 200;  
  
 public static void main(String args[]) throws Exception {  
  
 if(args.length<1||args.length>2) {  
 System.out.println("Usage: java CheckServerTLS {SERVER-ADDRESS} [{SERVER-PORT-NUMBER}]");  
 System.out.println(" The default {SERVER-PORT-NUMBER} is 443 (HHTPS)." );  
 System.exit(1);  
 }  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid SERVER-ADDRESS.");  
 System.exit(1);  
 }  
  
 if(args.length==2) {  
 try { serverPort = Integer.parseInt(args[1]); }  
 catch(NumberFormatException ex) {  
 System.out.println("Invalid SERVER-PORT.");  
 System.exit(1);  
 }  
 } else {  
 serverPort=443;  
 }  
   
 // Don't set the trust store, use the default: {JREHOME}/lib/security/cacerts  
 // In Linux, it's also: /etc/ssl/certs/java/cacerts  
 // System.setProperty("javax.net.ssl.trustStore", "cacerts.jks");  
 // System.setProperty("javax.net.ssl.trustStorePassword","public");  
   
 SSLSocketFactory sf = (SSLSocketFactory) SSLSocketFactory.getDefault();  
   
 try {  
 sock = (SSLSocket) sf.createSocket(serverIP,serverPort);  
 }  
 catch(IOException ex) {  
 System.out.println("Failed to connect to: " + args[0] + ":" + serverPort);  
 System.out.println("Application aborted.");  
 System.exit(1);  
 }  
  
 System.out.println("Connected to server: " + args[0] + ":" + serverPort);  
  
 try {  
 sock.startHandshake();  
 SSLSession ssl = sock.getSession();  
 System.out.println("------------------------------------------------------");  
 System.out.println("SSL/TLS version: " + ssl.getProtocol() +   
 " Cypher suite: " + ssl.getCipherSuite());  
  
 X509Certificate[] chain=ssl.getPeerCertificateChain();  
 System.out.println("------------------------------------------------------");  
 System.out.println("Certificate subject: " + chain[0].getSubjectDN());  
 System.out.println("------------------------------------------------------");  
 System.out.println("Certificate issuer: " + chain[0].getIssuerDN());  
 System.out.println("------------------------------------------------------");  
 System.out.println("Not before: " + chain[0].getNotBefore());  
 System.out.println("------------------------------------------------------");  
 System.out.println("Not after: " + chain[0].getNotAfter());  
 System.out.println("------------------------------------------------------");  
  
  
 }  
 catch (SSLException tlsE) {  
 System.out.println("SSL/TLS handshake has failed:\r\n" + tlsE.getCause() );  
 try { sock.close(); } catch(IOException ex2) { System.out.println("Error closing socket."); }  
 System.exit(1);  
 }  
 try { sock.close(); } catch(IOException ex2) { System.out.println("Error closing socket."); }  
  
 } // MAIN METHOD  
 } // CLASS

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\check-TLS\CheckServerTLS.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\check-TLS\Makefile

P1=CheckServerTLS  
  
all: $(P1).class  
  
$(P1).class: $(P1).java  
 javac $(P1).java  
  
clean:  
 rm -f \*.class \*.jks

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\check-TLS\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\HTTPmessage.java

import java.io.\*;  
  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
  
public class HTTPmessage {  
   
 private static final int CR=13;  
 private static final int LF=10;  
   
 private static final String VERSION="HTTP/1.1";  
   
 private static final String CONTENT\_TYPE="Content-type:";  
 private static final String CONTENT\_LENGTH="Content-length:";  
 private static final String CONNECTION="Connection:";  
   
 private static final String[][] knownFileExt = {  
 { ".pdf" , "application/pdf" } ,  
 { ".js" , "application/javascript" } ,  
 { ".txt" , "text/plain" } ,  
 { ".gif" , "image/gif" } ,  
 { ".png" , "image/png" }  
 };  
   
   
 static private String readHeaderLine(DataInputStream in) throws IOException {  
 String ret="";  
 int val;  
 do {  
 val=in.read();  
 if(val==-1) throw new IOException();  
 if(val!=CR) ret=ret+(char)val;  
 }  
 while(val!=CR);  
 val=in.read(); // read LF  
 if(val==-1) throw new IOException();  
 return ret;  
 }  
   
 static private void writeHeaderLine(DataOutputStream out, String line) throws IOException {  
 out.write(line.getBytes(), 0, line.length());  
 out.write(CR); out.write(LF);  
 }  
   
   
 //// NON-STATIC (INSTANCE) ELEMENTS  
   
 private boolean isRequest;  
 private String method;  
 private String uri;  
 private String status;  
   
 private String contentType;  
 private byte[] content;  
   
 /\*\*  
 \* Creates a new HTTPmessage by receiving it from an DataInputStream  
 \* @param in   
 \* @throws IOException  
 \*/  
 public HTTPmessage(DataInputStream in) throws IOException {  
 String firstLine=readHeaderLine(in);  
 isRequest= !firstLine.startsWith("HTTP/");  
 method=null;  
 uri=null;  
 content=null;  
 status=null;  
 contentType=null;  
  
 String[] firstLineComp=firstLine.split(" ");  
 if(isRequest) {  
 method=firstLineComp[0];  
 uri=firstLineComp[1];  
 }  
 else { // response  
 status=firstLineComp[1] + " " + firstLineComp[2];  
 }  
   
 String headerLine;  
   
 do {  
 headerLine=readHeaderLine(in);  
 if(headerLine.toUpperCase().startsWith(CONTENT\_TYPE.toUpperCase())) {  
 contentType=headerLine.substring(CONTENT\_TYPE.length()).trim();  
 }   
 else   
 if(headerLine.toUpperCase().startsWith(CONTENT\_LENGTH.toUpperCase())) {  
 String cLen=headerLine.substring(CONTENT\_LENGTH.length()).trim();  
 int len;  
 try { len=Integer.parseInt(cLen); }  
 catch(NumberFormatException ne) { throw new IOException(); }  
 content = new byte[len];  
 }   
 }  
 while(!headerLine.isEmpty());  
   
 // READ CONTENT  
 if(content!=null) in.readFully(content,0,content.length);  
 }  
   
 public HTTPmessage() {  
 isRequest=true;  
 method=null;  
 uri=null;  
 content=null;  
 status=null;  
 contentType=null;  
 }  
   
   
 public void setResponseStatus(String sT) {  
 isRequest=false;  
 status=sT;  
 }  
   
 public void setContent(String cnt, String cType) {   
 content=cnt.getBytes();  
 contentType=cType;  
 }  
  
 public void setRequestMethod(String m) {  
 isRequest=true;  
 method=m;  
 }  
   
   
 public boolean send(DataOutputStream out) throws IOException {  
 if(isRequest) {  
 if(method==null||uri==null) return false;  
 writeHeaderLine(out, method + " " + uri + " " + VERSION);  
 }  
 else {  
 if(status==null) return false;  
 writeHeaderLine(out,VERSION + " " + status);  
 }  
   
 if(content!=null) {  
 if(contentType!=null) writeHeaderLine(out,CONTENT\_TYPE + " " + contentType);  
 writeHeaderLine(out,CONTENT\_LENGTH + " " + content.length);  
 }  
 writeHeaderLine(out,CONNECTION + " close");  
 writeHeaderLine(out,"");  
 if(content!=null) {  
 out.write(content,0,content.length);  
 }  
 return true;  
 }  
   
   
 public String getMethod() { return method; }  
 public String getURI() { return uri; }  
 public String getStatus() { return status; }  
 public void setURI(String u) { uri=u; }  
   
  
  
 public boolean hasContent() { return (content!=null); }  
 public String getContentAsString() { return(new String(content)); }  
 public byte[] getContent() { return(content); }  
 public void setContentFromString(String c, String ct) {  
 content=c.getBytes(); contentType=ct;  
 }  
   
 public boolean setContentFromFile(String fname) {  
 File f=new File(fname);  
 contentType=null;  
 if(!f.exists()) {  
 content=null;   
 return false;  
 }   
 for (String[] k : knownFileExt) {  
 if (fname.endsWith(k[0]))  
 contentType = k[1];  
 }  
 if(contentType==null) contentType="text/html";  
  
 int cLen = (int) f.length();  
 if(cLen==0) {  
 content=null;  
 contentType=null;  
 return false;  
 }  
   
 content = new byte[cLen];  
   
 DataInputStream fr;  
 try {  
 fr = new DataInputStream(new FileInputStream(f));  
 try { fr.readFully(content,0,cLen); fr.close(); }  
 catch(IOException ex) {   
 System.out.println("Error reading file");   
 content=null;  
 contentType=null;   
 return false;  
 }  
 }  
 catch(FileNotFoundException ex) {   
 System.out.println("File Not Found");  
 content=null;  
 contentType=null;  
 return false;  
 }  
 return true;   
 }   
  
 } // CLASS END

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\HTTPmessage.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\HttpsServerAjaxVoting.java

import java.io.IOException;  
  
import javax.net.ssl.SSLServerSocket;  
import javax.net.ssl.SSLServerSocketFactory;  
import javax.net.ssl.SSLSocket;  
  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
public class HttpsServerAjaxVoting {  
 static private final String BASE\_FOLDER="www";  
 static private SSLServerSocket sock;  
  
 public static void main(String args[]) throws Exception {  
 SSLSocket cliSock;  
  
 if(args.length!=1) {  
 System.out.println("Local port number required at the command line.");  
 System.exit(1);  
 }  
  
 System.setProperty("javax.net.ssl.keyStore", "server.jks");  
 System.setProperty("javax.net.ssl.keyStorePassword", "forgotten");  
   
 accessesCounter=0;  
 for(int i=0;i<candidatesNumber; i++) {  
 candidateName[i] = "Candidate " + i;  
 candidateVotes[i] = 0;  
 }  
  
 try {  
 SSLServerSocketFactory sslF = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault();  
 sock = (SSLServerSocket) sslF.createServerSocket(Integer.parseInt(args[0]));  
 }  
 catch(IOException ex) {  
 System.out.println("Server failed to open local port " + args[0]);  
 System.exit(1);  
 }  
  
 while(true) {   
 cliSock= (SSLSocket) sock.accept();  
 httpAjaxVotingRequest req=new httpAjaxVotingRequest(cliSock, BASE\_FOLDER);  
 req.start();  
 incAccessesCounter();  
 }  
 }   
   
   
 // DATA ACCESSED BY THREADS - LOCKING REQUIRED  
   
 private static final int candidatesNumber = 4;  
 private static final String[] candidateName = new String[candidatesNumber];  
 private static final int[] candidateVotes = new int[candidatesNumber];  
 private static int accessesCounter;  
   
 private static synchronized void incAccessesCounter() { accessesCounter++; }  
   
 public static synchronized String getVotesStandingInHTML() {  
 String textHtml = "<hr><ul>";  
 for(int i=0; i<candidatesNumber; i++) {  
 textHtml = textHtml + "<li><button type=button onclick=voteFor(" + (i+1) +  
 ")>Vote for " + candidateName[i] + "</button> " +  
 " - " + candidateVotes[i] + " votes </li>";   
 }  
 textHtml = textHtml + "</ul><hr><p>HTTP server accesses counter: " + accessesCounter + "</p><hr>";  
 return textHtml;  
 }  
   
 public static synchronized void castVote(String i) {  
 int cN;  
 try { cN=Integer.parseInt(i); }  
 catch(NumberFormatException ne) { return; }  
 cN--;  
 if(cN >= 0 && cN < candidatesNumber) candidateVotes[cN]++;  
 }  
   
   
   
}

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\HttpsServerAjaxVoting.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\Makefile

P1=HttpsServerAjaxVoting  
C2=HTTPmessage  
C3=HttpAjaxVotingRequest  
  
all: $(P1).class  
  
$(P1).class: $(P1).java $(C2).class $(C3).class  
 javac $(P1).java  
  
$(C2).class: $(C2).java  
 javac $(C2).java  
  
$(C3).class: $(C3).java  
 javac $(C3).java  
cert:  
 ./make\_cert  
  
clean:  
 rm -f \*.class \*.jks

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\httpAjaxVotingRequest.java

import java.io.\*;  
  
//import java.net.Socket;  
import javax.net.ssl.SSLSocket;  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
public class httpAjaxVotingRequest extends Thread {  
 String baseFolder;  
 SSLSocket sock;  
 DataInputStream inS;  
 DataOutputStream outS;  
   
 public httpAjaxVotingRequest(SSLSocket s, String f) {  
 baseFolder=f; sock=s;  
 }   
   
 public void run() {  
 try {  
 outS = new DataOutputStream(sock.getOutputStream());  
 inS = new DataInputStream(sock.getInputStream());  
 }  
 catch(IOException ex) { System.out.println("Thread error on data streams creation"); }  
 try {  
 HTTPmessage request = new HTTPmessage(inS);  
 HTTPmessage response = new HTTPmessage();  
 // System.out.println(request.getURI());  
   
 if(request.getMethod().equals("GET")) {  
 if(request.getURI().equals("/votes")) {  
 response.setContentFromString(  
 HttpsServerAjaxVoting.getVotesStandingInHTML(), "text/html");  
 response.setResponseStatus("200 Ok");  
 }  
 else {  
 String fullname=baseFolder + "/";  
 if(request.getURI().equals("/")) fullname=fullname+"index.html";  
 else fullname=fullname+request.getURI();  
 if(response.setContentFromFile(fullname)) {  
 response.setResponseStatus("200 Ok");  
 }  
 else {  
 response.setContentFromString(  
 "<html><body><h1>404 File not found</h1></body></html>",  
 "text/html");  
 response.setResponseStatus("404 Not Found");   
 }  
 }  
 response.send(outS);   
 }  
 else { // NOT GET  
 if(request.getMethod().equals("PUT")   
 && request.getURI().startsWith("/votes/")) {  
 HttpsServerAjaxVoting.castVote(request.getURI().substring(7));  
 response.setResponseStatus("200 Ok");  
 }  
 else {  
 response.setContentFromString(  
 "<html><body><h1>ERROR: 405 Method Not Allowed</h1></body></html>",  
 "text/html");  
 response.setResponseStatus("405 Method Not Allowed");  
 }  
 response.send(outS);   
 }  
 }  
 catch(IOException ex) { }  
 try { sock.close();}  
 catch(IOException ex) { System.out.println("CLOSE IOException"); }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\httpAjaxVotingRequest.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\make\_cert

#!/bin/bash  
rm -f server.jks  
echo -e "$(hostname -f)\nDEI\nISEP\nPORTO\nPORTO\nPT\nyes" | keytool -genkey -v -alias server -keyalg RSA -keysize 2048 \  
 -validity 365 -keystore server.jks -storepass forgotten  
####  
#keytool -list -v -keystore server.jks -storepass forgotten  
####

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\make\_cert

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\www\index.html

<html><head><title>HTTP demo</title>  
<script src="rcomp-ajax.js"></script>  
</head>  
<body bgcolor=#C0C0C0 onload="refreshVotes()"><h1>HTTP server demo - Voting with AJAX</h1>  
<h3>Java version</h3>  
<hr>  
<center>  
<table width=60% border=1 cellpadding=20 cellspacing=20><tr>  
<td height="300" align=left width=50% valign="top">  
<big>  
<div id="votes">  
Please wait, loading voting results ...  
</div>  
</big>  
</td></tr></table>  
</center>  
<hr>  
<center><table border=0><tr><td align=center>Image contents are supported:<br><br><img src=http2.png><br>(http2.png)</td>  
<td align=center><img src=http.gif><br>(http.gif)</td></tr></table></center>  
</body></html>

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\www\index.html

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\www\rcomp-ajax.js

// IMPORTANT: notice the next request is scheduled only after the  
// previous request is fully processed either successfully  
// or not.  
  
function refreshVotes() {  
 var request = new XMLHttpRequest();  
 var vBoard=document.getElementById("votes");  
   
 request.onload = function() {  
 vBoard.innerHTML = this.responseText;  
 vBoard.style.color="black";  
 setTimeout(refreshVotes, 2000);  
 };  
   
 request.ontimeout = function() {  
 vBoard.innerHTML = "Server timeout, still trying ...";  
 vBoard.style.color="red";  
 setTimeout(refreshVotes, 100);   
 };  
   
 request.onerror = function() {   
 vBoard.innerHTML = "No server reply, still trying ...";  
 vBoard.style.color="red";  
 setTimeout(refreshVotes, 5000);   
 };  
   
 request.open("GET", "/votes", true);  
 request.timeout = 5000;  
 request.send();  
 }  
  
function voteFor(option) {  
 var request = new XMLHttpRequest();  
 request.open("PUT", "/votes/" + option , true);  
 request.send();  
 var vBoard=document.getElementById("votes");  
 vBoard.innerHTML = vBoard.innerHTML + "<p>Casting your vote ... Please wait.";  
  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\https-server-ajax-voting\www\rcomp-ajax.js

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\Makefile

P1=TcpCliSumTLS  
P2=TcpSrvSumTLS  
  
all: $(P1).class $(P2).class  
  
$(P1).class: $(P1).java  
 javac $(P1).java  
  
$(P2).class: $(P2).java  
 javac $(P2).java  
  
certs:  
 ./make\_certs  
  
clean:  
 rm -f \*.class \*.crt \*.key \*.pem \*.jks

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\TcpCliSumTLS.java

import java.io.\*;  
import java.net.\*;  
import java.security.KeyStore;  
import javax.net.\*;  
import javax.net.ssl.\*;  
import javax.security.cert.X509Certificate;  
  
/\*\*  
 \*  
 \* @author asc@isep.ipp.pt  
 \*/  
  
class TcpCliSumTLS {   
  
 static final int SERVER\_PORT=9999;  
 static final String KEYSTORE\_PASS="forgotten";  
  
 static InetAddress serverIP;  
 static SSLSocket sock;  
  
 public static void main(String args[]) throws Exception {  
 if(args.length!=2) {  
 System.out.println("Server IPv4/IPv6 address/DNS name is required as first argument");  
 System.out.println("Client name is required as second argument (a matching keystore must exist)");  
 System.exit(1); }  
   
 // Trust these certificates provided by servers  
 System.setProperty("javax.net.ssl.trustStore", args[1]+".jks");  
 System.setProperty("javax.net.ssl.trustStorePassword",KEYSTORE\_PASS);  
  
 // Use this certificate and private key for client certificate when requested by the server  
 System.setProperty("javax.net.ssl.keyStore",args[1]+".jks");  
 System.setProperty("javax.net.ssl.keyStorePassword",KEYSTORE\_PASS);  
  
 SSLSocketFactory sf = (SSLSocketFactory) SSLSocketFactory.getDefault();  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid server specified: " + args[0]);  
 System.exit(1); }  
  
  
 try {  
 sock = (SSLSocket) sf.createSocket(serverIP,SERVER\_PORT);  
 }  
 catch(IOException ex) {  
 System.out.println("Failed to connect to: " + args[0] + ":" + SERVER\_PORT);  
 System.out.println("Application aborted.");  
 System.exit(1);  
 }  
  
 System.out.println("Connected to: " + args[0] + ":" + SERVER\_PORT);  
  
  
 sock.startHandshake();  
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
 DataOutputStream sOut = new DataOutputStream(sock.getOutputStream());  
 DataInputStream sIn = new DataInputStream(sock.getInputStream());  
  
  
 String frase;  
 long f,i,n,num;  
 do {  
 do {  
 num=-1;  
 while(num<0) {  
 System.out.print("Enter a positive integer to SUM (zero to terminate): ");  
 frase = in.readLine();  
 try { num=Integer.parseInt(frase); }  
 catch(NumberFormatException ex) {num=-1;}  
 if(num<0) System.out.println("Invalid number");  
 }  
 n=num; for(i=0;i<4;i++) {sOut.write((byte)(n%256)); n=n/256; }  
 }  
 while(num!=0);  
 num=0; f=1;  
 for(i=0;i<4;i++) {num=num+f\*sIn.read(); f=f\*256;}  
 System.out.println("SUM RESULT = " + num);  
 }  
 while(num!=0);  
 sock.close();  
 }   
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\TcpCliSumTLS.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\TcpSrvSumTLS.java

import java.io.\*;   
import java.net.\*;   
  
import java.net.ServerSocket;  
import java.net.Socket;  
  
import javax.net.ssl.SSLServerSocket;  
import javax.net.ssl.SSLServerSocketFactory;  
import javax.net.ssl.SSLSocket;  
  
class TcpSrvSumTLS {   
  
 static final int SERVER\_PORT=9999;  
 static final String TRUSTED\_STORE="server\_J.jks";  
 static final String KEYSTORE\_PASS="forgotten";  
  
  
 public static void main(String args[]) throws Exception {   
 SSLServerSocket sock=null;  
 Socket cliSock;  
  
  
 // Trust these certificates provided by authorized clients  
 System.setProperty("javax.net.ssl.trustStore", TRUSTED\_STORE);  
 System.setProperty("javax.net.ssl.trustStorePassword",KEYSTORE\_PASS);  
  
 // Use this certificate and private key as server certificate  
 System.setProperty("javax.net.ssl.keyStore",TRUSTED\_STORE);  
 System.setProperty("javax.net.ssl.keyStorePassword",KEYSTORE\_PASS);  
  
 SSLServerSocketFactory sslF = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault();  
 try {  
 sock = (SSLServerSocket) sslF.createServerSocket(SERVER\_PORT);  
 sock.setNeedClientAuth(true);  
 }  
 catch(IOException ex) {  
 System.out.println("Server failed to open local port " + SERVER\_PORT);  
 System.exit(1);  
 }  
  
  
 while(true) {  
 cliSock=sock.accept();  
 new Thread(new TcpSrvSumTLSThread(cliSock)).start();  
 }  
 }   
 }  
  
  
  
class TcpSrvSumTLSThread implements Runnable {   
 private Socket s;  
 private DataOutputStream sOut;  
 private DataInputStream sIn;  
  
 public TcpSrvSumTLSThread(Socket cli\_s) { s=cli\_s;}  
  
 public void run() {  
 long f,i,num,sum;  
 InetAddress clientIP;  
  
 clientIP=s.getInetAddress();  
 System.out.println("New client connection from " + clientIP.getHostAddress() +   
 ", port number " + s.getPort());  
 try {  
 sOut = new DataOutputStream(s.getOutputStream());  
 sIn = new DataInputStream(s.getInputStream());  
 do {  
 sum=0;  
 do {  
 num=0; f=1; for(i=0;i<4;i++) {num=num+f\*sIn.read(); f=f\*256;}  
 sum=sum+num;  
 }  
 while(num>0);  
 num=sum; for(i=0;i<4;i++) {sOut.write((byte)(num%256)); num=num/256; }  
 }  
 while(sum>0);  
  
 System.out.println("Client " + clientIP.getHostAddress() + ", port number: " + s.getPort() +   
 " disconnected");  
 s.close();  
 }  
 catch(IOException ex) { System.out.println("IOException"); }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\TcpSrvSumTLS.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\make\_certs

#!/bin/bash  
STOREPASS="forgotten"  
for ENT in server\_J client1\_J client2\_J client3\_J client4\_J ; do  
 rm -f ${ENT}.jks ${ENT}.pem  
 echo -e "${ENT}\nDEI\nISEP\nPORTO\nPORTO\nPT\nyes" | keytool -genkey -v -alias ${ENT} -keyalg RSA -keysize 2048 \  
 -validity 365 -keystore ${ENT}.jks -storepass ${STOREPASS}  
 keytool -exportcert -alias ${ENT} -keystore ${ENT}.jks -storepass ${STOREPASS} -rfc -file ${ENT}.pem  
done  
####  
echo "Creating trust relations"  
### IMPORTING TRUSTED CERTIFICATES  
### (The server trusts all clients except for client4\_J)  
### (Every client trusts server\_J)  
for ENT in client1\_J client2\_J client3\_J; do  
 echo "yes"|keytool -import -alias ${ENT} -keystore server\_J.jks -file ${ENT}.pem -storepass ${STOREPASS}  
 echo "yes"|keytool -import -alias server\_J -keystore ${ENT}.jks -file server\_J.pem -storepass ${STOREPASS}  
done  
echo "yes"|keytool -import -alias server\_J -keystore client4\_J.jks -file server\_J.pem -storepass ${STOREPASS}  
echo "############################################################################"  
keytool -list -keystore server\_J.jks -storepass ${STOREPASS}  
echo "############################################################################"  
echo "WARNING: For testing, client4\_J is not added to the list of authorized clients"  
echo "############################################################################"  
#######

### [File Ends] sem4pi-22-23-61-master\base.rcomp\SSL\tcp-cli-srv\make\_certs

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\DemoConsumer.java

import java.io.\*;  
import java.net.\*;  
  
/\*\*  
 \*  
 \* @author asc@isep.ipp.pt  
 \*/  
public class DemoConsumer {  
 static private Socket sock;  
 static private InetAddress serverIP;  
 static private int serverPort;  
 static private DataOutputStream sOut;  
 static private DataInputStream sIn;  
  
 // THE NUMBER OF VOTES TO CAST ON THE FIRST CANDIDATE  
 static private final int VOTES\_TO\_CAST = 200;  
  
 public static void main(String args[]) throws Exception {  
  
 if(args.length!=2) {  
 System.out.println("Server address and port number required at command line.");  
 System.out.println("Usage: java DemoConsumer {SERVER-ADDRESS} {SERVER-PORT-NUMBER}");  
 System.exit(1);  
 }  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid SERVER-ADDRESS.");  
 System.exit(1);  
 }  
  
 try { serverPort = Integer.parseInt(args[1]); }  
 catch(NumberFormatException ex) {  
 System.out.println("Invalid SERVER-PORT.");  
 System.exit(1);  
 }  
   
 httpmessage request = new httpmessage();  
 request.setRequestMethod("PUT");  
 request.setURI("/votes/1");  
 System.out.println("Casting " + VOTES\_TO\_CAST + " votes on the first candidate ...");  
  
  
 for(int i=0; i<VOTES\_TO\_CAST; i++) {  
 System.out.println("Connecting to http://" + args[0] + ":" + serverPort + "/");  
 try { sock = new Socket(serverIP, serverPort); }  
 catch(IOException ex) {  
 System.out.println("Failed to connect to provided SERVER-ADDRESS and SERVER-PORT.");  
 System.out.println("Application aborted.");  
 System.exit(1);  
 }  
  
 try {  
 sOut = new DataOutputStream(sock.getOutputStream());  
 sIn = new DataInputStream(sock.getInputStream());  
 }  
 catch(IOException ex) {  
 System.out.println("Error accessing socket's streams. Aborted.");  
 try { sock.close(); } catch(IOException ex2) { System.out.println("Error closing socket."); }  
 System.out.println("Application aborted.");  
 System.exit(1);  
 }  
 System.out.println("Casting a vote");  
 request.send(sOut); // send HTTP request  
 httpmessage response = new httpmessage(sIn); // receive HTTP response  
 System.out.println("HTTP server response status: " + response.getStatus());  
 try { sock.close(); } catch(IOException ex2) { System.out.println("Error closing socket."); }  
 }  
  
 } // MAIN METHOD  
 } // CLASS

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\DemoConsumer.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\HttpAjaxVotingRequest.java

import java.io.\*;  
import java.net.Socket;  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
public class HttpAjaxVotingRequest extends Thread {  
 String baseFolder;  
 Socket sock;  
 DataInputStream inS;  
 DataOutputStream outS;  
   
 public HttpAjaxVotingRequest(Socket s, String f) {  
 baseFolder=f; sock=s;  
 }   
   
 public void run() {  
 try {  
 outS = new DataOutputStream(sock.getOutputStream());  
 inS = new DataInputStream(sock.getInputStream());  
 }  
 catch(IOException ex) { System.out.println("Thread error on data streams creation"); }  
 try {  
 httpmessage request = new httpmessage(inS);  
 httpmessage response = new httpmessage();  
 // System.out.println(request.getURI());  
   
 if(request.getMethod().equals("GET")) {  
 if(request.getURI().equals("/votes")) {  
 response.setContentFromString(  
 HttpServerAjaxVoting.getVotesStandingInHTML(), "text/html");  
 response.setResponseStatus("200 Ok");  
 }  
 else {  
 String fullname=baseFolder + "/";  
 if(request.getURI().equals("/")) fullname=fullname+"index.html";  
 else fullname=fullname+request.getURI();  
 if(response.setContentFromFile(fullname)) {  
 response.setResponseStatus("200 Ok");  
 }  
 else {  
 response.setContentFromString(  
 "<html><body><h1>404 File not found</h1></body></html>",  
 "text/html");  
 response.setResponseStatus("404 Not Found");   
 }  
 }  
 response.send(outS);   
 }  
 else { // NOT GET  
 if(request.getMethod().equals("PUT")   
 && request.getURI().startsWith("/votes/")) {  
 HttpServerAjaxVoting.castVote(request.getURI().substring(7));  
 response.setResponseStatus("200 Ok");  
 }  
 else {  
 response.setContentFromString(  
 "<html><body><h1>ERROR: 405 Method Not Allowed</h1></body></html>",  
 "text/html");  
 response.setResponseStatus("405 Method Not Allowed");  
 }  
 response.send(outS);   
 }  
 }  
 catch(IOException ex) { System.out.println("Thread error when reading request"); }  
 try { sock.close();}  
 catch(IOException ex) { System.out.println("CLOSE IOException"); }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\HttpAjaxVotingRequest.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\HttpServerAjaxVoting.java

import java.io.IOException;  
import java.net.ServerSocket;  
import java.net.Socket;  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
public class HttpServerAjaxVoting {  
 static private final String BASE\_FOLDER="www";  
 static private ServerSocket sock;  
  
 public static void main(String args[]) throws Exception {  
 Socket cliSock;  
  
 if(args.length!=1) {  
 System.out.println("Local port number required at the command line.");  
 System.exit(1);  
 }  
   
 accessesCounter=0;  
 for(int i=0;i<candidatesNumber; i++) {  
 candidateName[i] = "Candidate " + i;  
 candidateVotes[i] = 0;  
 }  
   
 try { sock = new ServerSocket(Integer.parseInt(args[0])); }  
 catch(IOException ex) {  
 System.out.println("Server failed to open local port " + args[0]);  
 System.exit(1);  
 }  
 while(true) {   
 cliSock=sock.accept();  
 HttpAjaxVotingRequest req=new HttpAjaxVotingRequest(cliSock, BASE\_FOLDER);  
 req.start();  
 incAccessesCounter();  
 }  
 }   
   
   
 // DATA ACCESSED BY THREADS - LOCKING REQUIRED  
   
 private static final int candidatesNumber = 4;  
 private static final String[] candidateName = new String[candidatesNumber];  
 private static final int[] candidateVotes = new int[candidatesNumber];  
 private static int accessesCounter;  
   
 private static synchronized void incAccessesCounter() { accessesCounter++; }  
   
 public static synchronized String getVotesStandingInHTML() {  
 String textHtml = "<hr><ul>";  
 for(int i=0; i<candidatesNumber; i++) {  
 textHtml = textHtml + "<li><button type=button onclick=voteFor(" + (i+1) +  
 ")>Vote for " + candidateName[i] + "</button> " +  
 " - " + candidateVotes[i] + " votes </li>";   
 }  
 textHtml = textHtml + "</ul><hr><p>HTTP server accesses counter: " + accessesCounter + "</p><hr>";  
 return textHtml;  
 }  
   
 public static synchronized void castVote(String i) {  
 int cN;  
 try { cN=Integer.parseInt(i); }  
 catch(NumberFormatException ne) { return; }  
 cN--;  
 if(cN >= 0 && cN < candidatesNumber) candidateVotes[cN]++;  
 }  
   
   
   
}

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\HttpServerAjaxVoting.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\Httpmessage.java

import java.io.\*;  
  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
  
public class Httpmessage {  
   
 private static final int CR=13;  
 private static final int LF=10;  
   
 private static final String VERSION="HTTP/1.1";  
   
 private static final String CONTENT\_TYPE="Content-type:";  
 private static final String CONTENT\_LENGTH="Content-length:";  
 private static final String CONNECTION="Connection:";  
   
 private static final String[][] knownFileExt = {  
 { ".pdf" , "application/pdf" } ,  
 { ".js" , "application/javascript" } ,  
 { ".txt" , "text/plain" } ,  
 { ".gif" , "image/gif" } ,  
 { ".png" , "image/png" }  
 };  
   
   
 static private String readHeaderLine(DataInputStream in) throws IOException {  
 String ret="";  
 int val;  
 do {  
 val=in.read();  
 if(val==-1) throw new IOException();  
 if(val!=CR) ret=ret+(char)val;  
 }  
 while(val!=CR);  
 val=in.read(); // read LF  
 if(val==-1) throw new IOException();  
 return ret;  
 }  
   
 static private void writeHeaderLine(DataOutputStream out, String line) throws IOException {  
 out.write(line.getBytes(), 0, line.length());  
 out.write(CR); out.write(LF);  
 }  
   
   
 //// NON-STATIC (INSTANCE) ELEMENTS  
   
 private boolean isRequest;  
 private String method;  
 private String uri;  
 private String status;  
   
 private String contentType;  
 private byte[] content;  
   
 /\*\*  
 \* Creates a new httpmessage by receiving it from an DataInputStream  
 \* @param in   
 \* @throws IOException  
 \*/  
 public Httpmessage(DataInputStream in) throws IOException {  
 String firstLine=readHeaderLine(in);  
 isRequest= !firstLine.startsWith("HTTP/");  
 method=null;  
 uri=null;  
 content=null;  
 status=null;  
 contentType=null;  
  
 String[] firstLineComp=firstLine.split(" ");  
 if(isRequest) {  
 method=firstLineComp[0];  
 uri=firstLineComp[1];  
 }  
 else { // response  
 status=firstLineComp[1] + " " + firstLineComp[2];  
 }  
   
 String headerLine;  
   
 do {  
 headerLine=readHeaderLine(in);  
 if(headerLine.toUpperCase().startsWith(CONTENT\_TYPE.toUpperCase())) {  
 contentType=headerLine.substring(CONTENT\_TYPE.length()).trim();  
 }   
 else   
 if(headerLine.toUpperCase().startsWith(CONTENT\_LENGTH.toUpperCase())) {  
 String cLen=headerLine.substring(CONTENT\_LENGTH.length()).trim();  
 int len;  
 try { len=Integer.parseInt(cLen); }  
 catch(NumberFormatException ne) { throw new IOException(); }  
 content = new byte[len];  
 }   
 }  
 while(!headerLine.isEmpty());  
   
 // READ CONTENT  
 if(content!=null) in.readFully(content,0,content.length);  
 }  
   
 public Httpmessage() {  
 isRequest=true;  
 method=null;  
 uri=null;  
 content=null;  
 status=null;  
 contentType=null;  
 }  
   
   
 public void setResponseStatus(String sT) {  
 isRequest=false;  
 status=sT;  
 }  
   
 public void setContent(String cnt, String cType) {   
 content=cnt.getBytes();  
 contentType=cType;  
 }  
  
 public void setRequestMethod(String m) {  
 isRequest=true;  
 method=m;  
 }  
   
   
 public boolean send(DataOutputStream out) throws IOException {  
 if(isRequest) {  
 if(method==null||uri==null) return false;  
 writeHeaderLine(out, method + " " + uri + " " + VERSION);  
 }  
 else {  
 if(status==null) return false;  
 writeHeaderLine(out,VERSION + " " + status);  
 }  
   
 if(content!=null) {  
 if(contentType!=null) writeHeaderLine(out,CONTENT\_TYPE + " " + contentType);  
 writeHeaderLine(out,CONTENT\_LENGTH + " " + content.length);  
 }  
 writeHeaderLine(out,CONNECTION + " close");  
 writeHeaderLine(out,"");  
 if(content!=null) {  
 out.write(content,0,content.length);  
 }  
 return true;  
 }  
   
   
 public String getMethod() { return method; }  
 public String getURI() { return uri; }  
 public String getStatus() { return status; }  
 public void setURI(String u) { uri=u; }  
   
  
  
 public boolean hasContent() { return (content!=null); }  
 public String getContentAsString() { return(new String(content)); }  
 public byte[] getContent() { return(content); }  
 public void setContentFromString(String c, String ct) {  
 content=c.getBytes(); contentType=ct;  
 }  
   
 public boolean setContentFromFile(String fname) {  
 File f=new File(fname);  
 contentType=null;  
 if(!f.exists()) {  
 content=null;   
 return false;  
 }   
 for (String[] k : knownFileExt) {  
 if (fname.endsWith(k[0]))  
 contentType = k[1];  
 }  
 if(contentType==null) contentType="text/html";  
  
 int cLen = (int) f.length();  
 if(cLen==0) {  
 content=null;  
 contentType=null;  
 return false;  
 }  
   
 content = new byte[cLen];  
   
 DataInputStream fr;  
 try {  
 fr = new DataInputStream(new FileInputStream(f));  
 try { fr.readFully(content,0,cLen); fr.close(); }  
 catch(IOException ex) {   
 System.out.println("Error reading file");   
 content=null;  
 contentType=null;   
 return false;  
 }  
 }  
 catch(FileNotFoundException ex) {   
 System.out.println("File Not Found");  
 content=null;  
 contentType=null;  
 return false;  
 }  
 return true;   
 }   
  
 } // CLASS END

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\Httpmessage.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\Makefile

P1=HttpServerAjaxVoting  
C2=HTTPmessage  
C3=HttpAjaxVotingRequest  
P2=DemoConsumer  
  
all: $(P1).class $(P2).class  
  
$(P1).class: $(P1).java $(C2).class $(C3).class  
 javac $(P1).java  
  
$(P2).class: $(P2).java $(C2).class  
 javac $(P2).java  
  
$(C2).class: $(C2).java  
 javac $(C2).java  
  
$(C3).class: $(C3).java  
 javac $(C3).java  
  
clean:  
 rm -f \*.class

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\www\index.html

<html><head><title>HTTP demo</title>  
<script src="rcomp-ajax.js"></script>  
</head>  
<body bgcolor=#C0C0C0 onload="refreshVotes()"><h1>HTTP server demo - Voting with AJAX</h1>  
<h3>Java version</h3>  
<hr>  
<center>  
<table width=60% border=1 cellpadding=20 cellspacing=20><tr>  
<td height="300" align=left width=50% valign="top">  
<big>  
<div id="votes">  
Please wait, loading voting results ...  
</div>  
</big>  
</td></tr></table>  
</center>  
<hr>  
<center><table border=0><tr><td align=center>Image contents are supported:<br><br><img src=http2.png><br>(http2.png)</td>  
<td align=center><img src=http.gif><br>(http.gif)</td></tr></table></center>  
</body></html>

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\www\index.html

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\www\rcomp-ajax.js

// IMPORTANT: notice the next request is scheduled only after the  
// previous request is fully processed either successfully  
// or not.  
  
function refreshVotes() {  
 var request = new XMLHttpRequest();  
 var vBoard=document.getElementById("votes");  
   
 request.onload = function() {  
 vBoard.innerHTML = this.responseText;  
 vBoard.style.color="black";  
 setTimeout(refreshVotes, 2000);  
 };  
   
 request.ontimeout = function() {  
 vBoard.innerHTML = "Server timeout, still trying ...";  
 vBoard.style.color="red";  
 setTimeout(refreshVotes, 100);   
 };  
   
 request.onerror = function() {   
 vBoard.innerHTML = "No server reply, still trying ...";  
 vBoard.style.color="red";  
 setTimeout(refreshVotes, 5000);   
 };  
   
 request.open("GET", "/votes", true);  
 request.timeout = 5000;  
 request.send();  
 }  
  
function voteFor(option) {  
 var request = new XMLHttpRequest();  
 request.open("PUT", "/votes/" + option , true);  
 request.send();  
 var vBoard=document.getElementById("votes");  
 vBoard.innerHTML = vBoard.innerHTML + "<p>Casting your vote ... Please wait.";  
  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-ajax-voting\www\rcomp-ajax.js

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\HttpChatConsumer.java

import java.io.\*;   
import java.net.\*;   
  
class HttpChatConsumer {   
 private static InetAddress serverIP;  
 private static int serverPort;  
 public static boolean userExit;  
   
 public static void main(String args[]) throws Exception {  
 String nickName, textLine;  
  
 if(args.length!=2) {  
 System.out.println("Server address and port number required at command line.");  
 System.out.println("Usage: java HttpChatConsumer {SERVER-ADDRESS} {SERVER-PORT-NUMBER}");  
 System.exit(1);  
 }  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid SERVER-ADDRESS: " + args[0]);  
 System.exit(1); }  
  
 try { serverPort = Integer.parseInt(args[1]); }  
 catch(NumberFormatException ex) {  
 System.out.println("Invalid SERVER-PORT-NUMBER: " + args[1]);  
 System.exit(1);  
 }  
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
  
 System.out.print("Enter your nickname: ");  
 nickName = in.readLine();  
  
 userExit=false;  
  
 System.out.println("Press ENTER to fetch all current messages from the server and start chatting.");  
 System.out.println("Once chatting, type exit to quit. NOW PRESS ENTER PLEASE.");  
 textLine = in.readLine();  
 // start a thread to fetch messages from the HTTP server (GET)  
 MessageFetcher msgF = new MessageFetcher(serverIP, serverPort);  
 msgF.start();  
  
 while(!userExit) { // read messages from the console and send them to the HTTP server (POST)  
 textLine=in.readLine();  
 if(textLine.equals("exit")) {  
 userExit=true;  
   
 }  
 else {  
 if(!postMessage("(" + nickName + ") " + textLine)) {  
 System.out.println("Sorry, server not responding, message not posted");  
 // msgF.abort();  
 }  
 }  
 }  
 msgF.abort();  
 msgF.join();  
 }   
  
 private static boolean postMessage(String msg) {  
 Socket TCPconn;  
 DataOutputStream sOut;  
 DataInputStream sIn;  
 try {  
 TCPconn = new Socket(serverIP, serverPort);  
 }  
 catch(IOException ex) {  
 return false;  
 }  
   
 try {  
 sOut = new DataOutputStream(TCPconn.getOutputStream());  
 sIn = new DataInputStream(TCPconn.getInputStream());  
 httpmessage request = new httpmessage();  
 request.setRequestMethod("POST");  
 request.setURI("/messages");  
 request.setContentFromString(msg, "text/plain");  
 request.send(sOut);  
 httpmessage response = new httpmessage(sIn);  
 if(!response.getStatus().startsWith("200")) {  
 throw new IOException();  
 }  
  
 }  
 catch(IOException ex) {  
 try { TCPconn.close(); } catch(IOException ex2) { }  
 return false;  
 }  
 try { TCPconn.close(); } catch(IOException ex2) { }  
 return true;  
 }  
  
  
} // HttpChatConsumer CLASS  
  
  
  
class MessageFetcher extends Thread {   
 private Socket TCPconn;  
 private InetAddress serverIP;  
 private int serverPort;  
 private int nextMessage;  
  
 public MessageFetcher(InetAddress ip, int port) { serverIP=ip; serverPort=port; }  
  
 public void run() {  
 nextMessage=0;  
 while(!HttpChatConsumer.userExit) {  
 String message = getNextMessage();  
 if(HttpChatConsumer.userExit) return;  
 if(message==null) {  
 System.out.println("Server not responding ...");  
 nextMessage=0;  
 try {Thread.sleep(5000); }  
 catch(InterruptedException ie) {  
 System.out.println("Thread Interrupted ...");  
 }  
 }  
 else {  
 System.out.println(message);  
 nextMessage++;  
 }  
 }  
 }  
   
  
 private String getNextMessage() {  
 DataOutputStream sOut;  
 DataInputStream sIn;  
 try {   
 synchronized (serverIP) { TCPconn = new Socket(serverIP, serverPort);}  
 }  
 catch(IOException ex) {  
 synchronized (serverIP) { TCPconn = null;}  
 return null;  
 }  
  
 httpmessage response;  
 try {  
 sOut = new DataOutputStream(TCPconn.getOutputStream());  
 sIn = new DataInputStream(TCPconn.getInputStream());  
 httpmessage request = new httpmessage();  
 request.setRequestMethod("GET");  
 request.setURI("/messages/" + nextMessage);  
 request.send(sOut);  
 response = new httpmessage(sIn); // the server may hold the response  
 if(!response.getStatus().startsWith("200")) {  
 throw new IOException();  
 }  
  
 }  
 catch(IOException ex) {  
 synchronized (serverIP) {   
 try { TCPconn.close(); } catch(IOException ex2) { }  
 TCPconn = null;  
 }  
 return null;  
 }  
 synchronized (serverIP) {   
 try { TCPconn.close(); } catch(IOException ex2) { }  
 TCPconn = null;  
 }  
 return(new String(response.getContent()));  
 }  
  
 public void abort() { // close the socket to force the thread's exit  
 synchronized (serverIP) {   
 if(TCPconn==null) return;  
 try { TCPconn.close(); } catch(IOException ex2) { }  
 }  
 }  
  
} // MessageFetcher CLASS

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\HttpChatConsumer.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\HttpChatRequest.java

import java.io.\*;  
import java.net.\*;  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
public class HttpChatRequest extends Thread {  
 String baseFolder;  
 Socket sock;  
 DataInputStream inS;  
 DataOutputStream outS;  
  
 public HttpChatRequest(Socket s, String f) {  
 baseFolder=f; sock=s;  
 }  
   
 @Override  
 public void run() {  
 try {  
 outS = new DataOutputStream(sock.getOutputStream());  
 inS = new DataInputStream(sock.getInputStream());  
 }  
 catch(IOException ex) { System.out.println("Thread error on data streams creation"); }  
 try {  
 httpmessage request = new httpmessage(inS);  
 httpmessage response = new httpmessage();  
 response.setResponseStatus("200 Ok");  
   
 if(request.getMethod().equals("GET")) {  
 if(request.getURI().startsWith("/messages/")) {  
 int msgNum;  
 try { msgNum=Integer.parseInt(request.getURI().substring(10)); }  
 catch(NumberFormatException ne) { msgNum=-1; }  
 if(msgNum<0) {  
 response.setContentFromString(  
 "<html><body><h1>ERROR: 405 Method Not Allowed</h1></body></html>",  
 "text/html");  
 response.setResponseStatus("405 Method Not Allowed");   
 }  
 else {  
 String msg=HttpServerChat.getMsg(msgNum);  
 // if msgNum doesn't yet exist, the getMsg() method waits  
 // until it does. So the HTTP request was received, but the  
 // HTTP response is sent only when there's a message  
 response.setContentFromString(msg, "text/plain");   
 }  
 }  
 else { // NOT GET /messages/ , THEN IT MUST BE A FILE  
 String fullname=baseFolder + "/";  
 if(request.getURI().equals("/")) fullname=fullname+"index.html";  
 else fullname=fullname+request.getURI();  
 if(!response.setContentFromFile(fullname)) {  
 response.setContentFromString(  
 "<html><body><h1>404 File not found</h1></body></html>",  
 "text/html");  
 response.setResponseStatus("404 Not Found");   
 }   
 }  
 }  
 else { // NOT GET, must be POST  
 if(request.getMethod().equals("POST")   
 && request.getURI().equals("/messages")) {  
 HttpServerChat.addMsg(request.getContentAsString());  
 response.setResponseStatus("200 Ok");   
 }  
 else {  
 response.setContentFromString(  
 "<html><body><h1>ERROR: 405 Method Not Allowed</h1></body></html>",  
 "text/html");  
 response.setResponseStatus("405 Method Not Allowed");  
 }  
 }  
 response.send(outS); // SEND THE HTTP RESPONSE  
 }  
 catch(IOException ex) { System.out.println("Thread I/O error on request/response"); }  
 try { sock.close();} catch(IOException ex) { System.out.println("CLOSE IOException"); }  
 }  
}

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\HttpChatRequest.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\HttpServerChat.java

import java.io.\*;  
import java.net.\*;  
import java.util.\*;  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
public class HttpServerChat {  
 static private final String BASE\_FOLDER="www";  
 static private ServerSocket sock;  
  
 public static void main(String args[]) throws Exception {  
 Socket cliSock;  
   
 if(args.length!=1) {  
 System.out.println("Local port number required at the command line.");  
 System.exit(1);  
 }  
 try { sock = new ServerSocket(Integer.parseInt(args[0])); }  
 catch(IOException ex) {  
 System.out.println("Server failed to open local port " + args[0]);  
 System.exit(1);  
 }  
 System.out.println("Server ready, listening on port number " + args[0]);  
 addMsg("HTTP Chat Server is ready ...");  
 while(true) {   
 cliSock=sock.accept();  
 HttpChatRequest req=new HttpChatRequest(cliSock, BASE\_FOLDER);  
 req.start();  
 }  
 }   
   
 // MESSAGES ARE ACCESSED BY THREADS - LOCKING REQUIRED  
 private static int nextMsgNum = 0;  
 private static final ArrayList<String> MSG\_LIST = new ArrayList<>();  
  
 public static String getMsg(int msgNumber) {  
 synchronized(MSG\_LIST) {  
 while(msgNumber>=nextMsgNum) {   
 try { MSG\_LIST.wait(); } // wait for a notification on MSG\_LIST's monitor  
 // while waiting MSG\_LIST's intr lock is released  
 catch(InterruptedException ex) {  
 System.out.println("Thread error: interrupted");  
 return null;  
 }  
 }  
 return MSG\_LIST.get(msgNumber);  
 }  
 }  
   
 public static void addMsg(String msg) {  
 synchronized(MSG\_LIST) {  
 MSG\_LIST.add(nextMsgNum, msg);  
 nextMsgNum++;  
 MSG\_LIST.notifyAll(); // notify all threads waiting on MSG\_LIST's monitor  
 }  
 }  
   
   
}

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\HttpServerChat.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\Makefile

P1=HttpServerChat  
P2=HttpChatConsumer  
C2=HTTPmessage  
C3=HttpChatRequest  
  
all: $(P1).class $(P2).class  
  
$(P1).class: $(P1).java $(C2).class $(C3).class  
 javac $(P1).java  
  
$(P2).class: $(P2).java $(C2).class  
 javac $(P2).java  
  
$(C2).class: $(C2).java  
 javac $(C2).java  
  
$(C3).class: $(C3).java  
 javac $(C3).java  
  
clean:  
 rm -f \*.class

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\README

JAVA/http-server-chat/HttpServerChat.java  
  
 \*\*\* HTTP Chat server \*\*\*  
  
This implementation addresses one classical issue with web services:  
  
 How to notify a consumer something has happened on the server's side?  
  
 (in this implementation, the event is: there's a new message to be fetched)  
  
  
The most basic approach is forcing consumers to periodically make a request to check  
if there's something new (polling). Though this is not very efficient, clients come  
to know about events with some delay and it's processing and networking intensive.  
  
Here another approach is used, in fact, HTTP doesn't establish a maximum time for a  
server to reply a client's request. Based on this, the server can hold the response  
until the event occurs, when it does the response is sent.  
  
This is ok as far as the client uses asynchronous HTTP requests (like with AJAX) and  
doesn't block waiting for the response.  
  
One drawback of this solution is the server will have one pending response for each  
client, this may present a problem for the server if there are too many clients.  
  
Regarding this specific implementation:  
  
- Chat messages are nummbered by the server, starting from zero, as they come in.  
  
- Clients send messages by requesting POST /messages  
  
- Clients retrieve messages from the servers by requesting GET /messages/N, they  
start with GET /messages/0, once the response is received, they request GET /messages/1,  
and so on.  
  
- If a client requests for a non existing message, the server holds the response until  
it exists.  
  
- The client only requests the next message once the response to the previous message  
is obtained. No timeout is defined, so while the server is holding the response, the  
client is idle, waiting for it.  
  
- On the server side, each request is processed by a thread, if the request is   
GET /messages/N and there's no message N yet, the thread is put into a wait state.  
Once a client posts a new message, all waiting threads are notified and awaked.

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\README

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\httpmessage.java

import java.io.\*;  
  
  
/\*\*  
 \*  
 \* @author ANDRE MOREIRA (asc@isep.ipp.pt)  
 \*/  
  
public class httpmessage {  
   
 private static final int CR=13;  
 private static final int LF=10;  
   
 private static final String VERSION="HTTP/1.1";  
   
 private static final String CONTENT\_TYPE="Content-type:";  
 private static final String CONTENT\_LENGTH="Content-length:";  
 private static final String CONNECTION="Connection:";  
   
 private static final String[][] knownFileExt = {  
 { ".pdf" , "application/pdf" } ,  
 { ".js" , "application/javascript" } ,  
 { ".txt" , "text/plain" } ,  
 { ".gif" , "image/gif" } ,  
 { ".png" , "image/png" }  
 };  
   
   
 static private String readHeaderLine(DataInputStream in) throws IOException {  
 String ret="";  
 int val;  
 do {  
 val=in.read();  
 if(val==-1) throw new IOException();  
 if(val!=CR) ret=ret+(char)val;  
 }  
 while(val!=CR);  
 val=in.read(); // read LF  
 if(val==-1) throw new IOException();  
 return ret;  
 }  
   
 static private void writeHeaderLine(DataOutputStream out, String line) throws IOException {  
 out.write(line.getBytes(), 0, line.length());  
 out.write(CR); out.write(LF);  
 }  
   
   
 //// NON-STATIC (INSTANCE) ELEMENTS  
   
 private boolean isRequest;  
 private String method;  
 private String uri;  
 private String status;  
   
 private String contentType;  
 private byte[] content;  
   
 /\*\*  
 \* Creates a new httpmessage by receiving it from an DataInputStream  
 \* @param in   
 \* @throws IOException  
 \*/  
 public httpmessage(DataInputStream in) throws IOException {  
 String firstLine=readHeaderLine(in);  
 isRequest= !firstLine.startsWith("HTTP/");  
 method=null;  
 uri=null;  
 content=null;  
 status=null;  
 contentType=null;  
  
 String[] firstLineComp=firstLine.split(" ");  
 if(isRequest) {  
 method=firstLineComp[0];  
 uri=firstLineComp[1];  
 }  
 else { // response  
 status=firstLineComp[1] + " " + firstLineComp[2];  
 }  
   
 String headerLine;  
   
 do {  
 headerLine=readHeaderLine(in);  
 if(headerLine.toUpperCase().startsWith(CONTENT\_TYPE.toUpperCase())) {  
 contentType=headerLine.substring(CONTENT\_TYPE.length()).trim();  
 }   
 else   
 if(headerLine.toUpperCase().startsWith(CONTENT\_LENGTH.toUpperCase())) {  
 String cLen=headerLine.substring(CONTENT\_LENGTH.length()).trim();  
 int len;  
 try { len=Integer.parseInt(cLen); }  
 catch(NumberFormatException ne) { throw new IOException(); }  
 content = new byte[len];  
 }   
 }  
 while(!headerLine.isEmpty());  
   
 // READ CONTENT  
 if(content!=null) in.readFully(content,0,content.length);  
 }  
   
 public httpmessage() {  
 isRequest=true;  
 method=null;  
 uri=null;  
 content=null;  
 status=null;  
 contentType=null;  
 }  
   
   
 public void setResponseStatus(String sT) {  
 isRequest=false;  
 status=sT;  
 }  
   
 public void setContent(String cnt, String cType) {   
 content=cnt.getBytes();  
 contentType=cType;  
 }  
  
 public void setRequestMethod(String m) {  
 isRequest=true;  
 method=m;  
 }  
   
   
 public boolean send(DataOutputStream out) throws IOException {  
 if(isRequest) {  
 if(method==null||uri==null) return false;  
 writeHeaderLine(out, method + " " + uri + " " + VERSION);  
 }  
 else {  
 if(status==null) return false;  
 writeHeaderLine(out,VERSION + " " + status);  
 }  
   
 if(content!=null) {  
 if(contentType!=null) writeHeaderLine(out,CONTENT\_TYPE + " " + contentType);  
 writeHeaderLine(out,CONTENT\_LENGTH + " " + content.length);  
 }  
 writeHeaderLine(out,CONNECTION + " close");  
 writeHeaderLine(out,"");  
 if(content!=null) {  
 out.write(content,0,content.length);  
 }  
 return true;  
 }  
   
   
 public String getMethod() { return method; }  
 public String getURI() { return uri; }  
 public String getStatus() { return status; }  
 public void setURI(String u) { uri=u; }  
   
  
  
 public boolean hasContent() { return (content!=null); }  
 public String getContentAsString() { return(new String(content)); }  
 public byte[] getContent() { return(content); }  
 public void setContentFromString(String c, String ct) {  
 content=c.getBytes(); contentType=ct;  
 }  
   
 public boolean setContentFromFile(String fname) {  
 File f=new File(fname);  
 contentType=null;  
 if(!f.exists()) {  
 content=null;   
 return false;  
 }   
 for (String[] k : knownFileExt) {  
 if (fname.endsWith(k[0]))  
 contentType = k[1];  
 }  
 if(contentType==null) contentType="text/html";  
  
 int cLen = (int) f.length();  
 if(cLen==0) {  
 content=null;  
 contentType=null;  
 return false;  
 }  
   
 content = new byte[cLen];  
   
 DataInputStream fr;  
 try {  
 fr = new DataInputStream(new FileInputStream(f));  
 try { fr.readFully(content,0,cLen); fr.close(); }  
 catch(IOException ex) {   
 System.out.println("Error reading file");   
 content=null;  
 contentType=null;   
 return false;  
 }  
 }  
 catch(FileNotFoundException ex) {   
 System.out.println("File Not Found");  
 content=null;  
 contentType=null;  
 return false;  
 }  
 return true;   
 }   
  
 } // CLASS END

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\httpmessage.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\www\index.html

<html><head><title>HTTP server demo - CHAT</title>  
<script src="rcomp-chat.js"></script>  
</head>  
<body bgcolor="#C0C0C0" onload="loadAndStart()">  
<table border="0" cellspacing="0" width="1100">  
<tr><td valign=top bgcolor="#ffffff">  
<h1>HTTP server demo - CHAT</h1>  
<h3>Java version</h3>  
</td><td align="center" bgcolor="#ffffff"><img src="chat.png" height="100" width="200"></td></tr>  
<tr><td colspan="2">  
<hr>  
<textarea id="messages" rows="20" cols="160" readonly>  
Please wait, loading messages ...  
</textarea>  
</td></tr>  
<tr><td>Nickname: <input type="text" id="nickname"></td><td><div id="hints"></div></td></tr>  
<tr><td colspan="2">Text message:   
<input type="text" id="message" size="100"><input type="button" value="SEND" onclick="postMessage()">  
<hr>  
</td></tr></table>  
</body></html>

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\www\index.html

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-chat\www\rcomp-chat.js

// global variables  
var nextMsg;  
var mArea,nicknameBox, messageBox, hints; // defined only after the document is loaded  
  
function loadAndStart() {  
 mArea=document.getElementById("messages");  
 nicknameBox=document.getElementById("nickname");  
 messageBox=document.getElementById("message");  
 hints=document.getElementById("hints");  
 nextMsg=0;  
 setTimeout(getNextMessage, 1000);  
 }  
  
function getNextMessage() {  
 var request = new XMLHttpRequest();  
   
 request.onload = function() {  
 if(nextMsg===0) {   
 mArea.value = "";  
 mArea.style.color="blue";  
 }  
 mArea.value = mArea.value + this.responseText + "\r\n";  
 mArea.scrollTop = mArea.scrollHeight; // scroll the textarea to make last lines visible  
 nextMsg=nextMsg+1;   
 setTimeout(getNextMessage, 100);  
 };  
  
 request.onerror = function() {   
 nextMsg=0;  
 mArea.value = "Server not responding.";  
 mArea.style.color="red";  
 setTimeout(getNextMessage, 1000);   
 };  
  
 request.ontimeout = function() {   
 nextMsg=0;  
 mArea.value = "Server not responding.";  
 mArea.style.color="red";  
 setTimeout(getNextMessage, 100);   
 };  
   
   
 request.open("GET", "/messages/" + nextMsg, true);  
 if(nextMsg===0) request.timeout = 1000;  
 // Message 0 is a server's greeting, it should always exist  
 // no timeout, for following messages, the server responds only when the requested  
 // message number exists  
 request.send();  
}  
  
function postMessage() {  
 hints.innerHTML="";  
 if(nicknameBox.value === "") {  
 hints.innerHTML="Settle a nickname before sending.";  
 return;  
 }  
 if(messageBox.value === "") {   
 hints.innerHTML="Not sending empty message.";  
 return;  
 }  
 var POSTrequest = new XMLHttpRequest();  
 nicknameBox.disabled=true;  
 POSTrequest.open("POST", "/messages", true);  
 POSTrequest.timeout = 5000;  
 POSTrequest.send("(" + nicknameBox.value + ") " + messageBox.value);  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-chat\www\rcomp-chat.js

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\HTTP.java

import java.io.\*;   
import java.net.\*;   
  
class HTTP {   
 static private final String HTTP\_VERSION ="HTTP/1.0";  
 static private final String HTTP\_CONNECTION\_CLOSE ="Connection: close";  
 static private final String HTTP\_CRLF ="\r\n";  
   
 static String readLineCRLF(DataInputStream sIn) {  
 byte[] line = new byte[300];  
 int p=0;  
 try {  
 for(;;) {  
 sIn.read(line,p,1);  
 if(line[p]=='\n') return(new String( line, 0, p));  
 else  
 if(line[p]!='\r') p++;  
 }  
 }  
 catch(IOException ex) { System.out.println("READ IOException"); }  
 return(null);  
 }  
  
 static void writeLineCRLF(DataOutputStream sOut, String line) {  
 try {  
 sOut.write(line.getBytes(),0,(byte)line.length());  
 sOut.write(HTTP\_CRLF.getBytes(),0,2);  
 }   
 catch(IOException ex) { System.out.println("WRITE IOException"); }  
 }   
  
  
 static void sendHttpResponseHeader(DataOutputStream sOut, String status, String contentType, int contentLength) {  
 writeLineCRLF(sOut, HTTP\_VERSION + " " + status);  
 writeLineCRLF(sOut, "Content-Type: " + contentType);  
 writeLineCRLF(sOut, "Content-Length: " + contentLength);  
 writeLineCRLF(sOut, HTTP\_CONNECTION\_CLOSE);  
 writeLineCRLF(sOut, "");  
 }  
  
 static void sendHttpResponse(DataOutputStream sOut, String status, String contentType,   
 byte[] content, int contentLength) {  
 sendHttpResponseHeader(sOut,status,contentType,contentLength);  
 try {  
 sOut.write(content,0,contentLength);  
 }  
 catch(IOException ex) { System.out.println("IOException"); }  
 }  
  
  
 static void sendHttpStringResponse(DataOutputStream sOut, String status, String contentType, String content) {  
 sendHttpResponse(sOut,status,contentType,content.getBytes(),content.length());  
 }  
  
  
 static void sendHttpFileResponse(DataOutputStream sOut, String status, String filePath) {  
 String responseStatus="200 Ok";  
 String contentType="text/html";  
 File f;  
  
 f=new File(filePath);  
 if(!f.exists()) {  
 sendHttpStringResponse(sOut, "404 Not Found", contentType,   
 "<html><body><h1>404 File not found</h1></body></html>");  
 return;  
 }  
 if(filePath.endsWith(".pdf")) contentType="application/pdf";  
 else  
 if(filePath.endsWith(".js")) contentType="application/javascript";  
 else  
 if(filePath.endsWith(".txt")) contentType="text/plain";  
 else  
 if(filePath.endsWith(".gif")) contentType="image/gif";  
 else  
 if(filePath.endsWith(".png")) contentType="image/png";  
  
 if(status!=null) responseStatus=status;  
  
 int len= (int) f.length();  
 sendHttpResponseHeader(sOut,responseStatus,contentType,len);  
 byte[] data = new byte[300];  
 int done, readNow;  
 InputStream fReader;  
 try {  
 fReader = new BufferedInputStream(new FileInputStream(f));  
 do {  
 if(len>300) readNow=300; else readNow=(int)len;  
 try {  
 done=fReader.read(data,0,readNow);  
 len=len-done;  
 sOut.write(data,0,done);  
 }  
 catch(IOException ex) { System.out.println("IOException"); }  
 }  
 while(len>0);  
  
 }  
 catch(FileNotFoundException ex) { System.out.println("FILE OPEN FileNotFoundException"); }  
 }  
  
  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\HTTP.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\HttpRequest.java

import java.io.\*;   
import java.net.\*;   
  
class HttpRequest implements Runnable {   
 private Socket s;  
 private DataOutputStream sOut;  
 private DataInputStream sIn;  
 private String baseFolder;  
  
 public HttpRequest(Socket cli\_s, String folder) { s=cli\_s; baseFolder=folder;}  
  
 public void run() {  
 try {  
 sOut = new DataOutputStream(s.getOutputStream());  
 sIn = new DataInputStream(s.getInputStream());  
 }  
 catch(IOException ex) { System.out.println("Data Stream IOException"); }  
 String request=HTTP.readLineCRLF(sIn);  
 // System.out.println("Request: " + request);  
 if(request.startsWith("POST /upload")) processPostUpload();  
 else  
 if(request.startsWith("POST /list")) processPostList();  
 else processGet(request);  
   
 try { s.close();}  
 catch(IOException ex) { System.out.println("CLOSE IOException"); }  
 }  
  
  
 void processGet(String req) {  
 String line, fileName, filePath;  
  
 do { // READ AND IGNORE THE HEADER LINES  
 line=HTTP.readLineCRLF(sIn);  
 }   
 while(line.length()>0);  
  
 fileName=req.split(" ")[1];  
 if(fileName.compareTo("/")==0) fileName="/index.html";  
 filePath=baseFolder+fileName;  
 HTTP.sendHttpFileResponse(sOut, null, filePath);  
 }  
  
  
 void processPostUpload() {  
 String line, boundary, filename, filePath;  
 int done,readNow,len;  
 String cDisp="Content-Disposition: form-data; name=\"filename\"; filename=\"";  
 File f;  
 FileOutputStream fOut;  
 byte[] data = new byte[300];  
  
 len=0;  
 boundary=null;  
 do { // FIRSH HEADER  
 line=HTTP.readLineCRLF(sIn);  
 if(line.startsWith("Content-Length: ")) {  
 len=Integer.parseInt(line.split(" ")[1]);  
 }  
 else  
 if(line.startsWith("Content-Type: multipart/form-data; boundary=")) {  
 boundary=line.split("=")[1];  
 }  
 }  
 while(line.length()>0);  
  
 if(len==0) {replyPostError("Content-Length: expected and not found"); return;}  
 if(boundary==null) {replyPostError("Content-Type: multipart/form-data; expected and not found"); return;}  
  
 line=HTTP.readLineCRLF(sIn);  
 if(!line.endsWith(boundary)) {replyPostError("Multipart separator expected and not found"); return;}  
 len=len-line.length()-2;  
 filename="";  
 do { // SECOND HEADER  
 line=HTTP.readLineCRLF(sIn);  
 len=len-line.length()-2;  
 if(line.startsWith(cDisp)) {  
 filename=line.split("=")[2];  
 filename=filename.substring(1,filename.length()-1);  
 }  
 }  
 while(line.length()>0);  
  
 try {  
 if(filename.length()==0) {  
 do { done=sIn.read(data,0,300);len=len-done;} while(len>0);  
 replyPostError("Content-Disposition: form-data; expected and not found (NO FILENAME)");  
 return;  
 }  
 filePath=baseFolder+"/"+filename;  
 f = new File(filePath);  
 fOut = new FileOutputStream(f);  
  
 len=len-boundary.length()-6;  
  
 do {  
 if(len>300) readNow=300; else readNow=len;  
 done=sIn.read(data,0,readNow);  
 fOut.write(data,0,done);  
 len=len-done;  
 }  
 while(len>0);  
 fOut.close();  
 line=HTTP.readLineCRLF(sIn);  
 }  
 catch(IOException ex) { System.out.println("IOException"); }  
 replyPostList();  
 }  
  
  
 void processPostList() {  
 String line;  
 do { // READ AND IGNORE THE HEADER LINES  
 line=HTTP.readLineCRLF(sIn);  
 }  
 while(line.length()>0);  
 replyPostList();  
 }  
  
  
 void replyPostList() {  
 String s1="<html><head><title>File List</title></head><body><h1>File List:</h1><big><ul>";  
 String s2="</ul></big><hr><p><a href=/>BACK</a></body></html>";  
 String list;  
 File d;  
  
 d= new File("www/");  
 list=s1;  
 File[] filesList = d.listFiles();  
 for(File f : filesList)  
 if(f.isFile())  
 list=list + "<li><a href=/" + f.getName() + ">" + f.getName() + "</a>";  
 list=list+s2;  
 HTTP.sendHttpStringResponse(sOut, "200 Ok", "text/html", list);  
 }  
  
  
 void replyPostError(String error) {  
 HTTP.sendHttpStringResponse(sOut, "500 Internal Server Error", "text/html",   
 "<html><head><title>Server Error</title></head><body><center><img src=500.png><br>(500.png)</center>"  
 + "<h1>Server error on POST</h1><p>ERROR: " + error +  
 "<hr><p><a href=/>BACK</a></body></html>");  
 }  
  
  
 } // END OF CLASS

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\HttpRequest.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\HttpServerFormFileUpload.java

import java.io.\*;   
import java.net.\*;   
  
class HttpServerFormFileUpload {   
 static private final String BASE\_FOLDER="www";  
 static private ServerSocket sock;  
  
 public static void main(String args[]) throws Exception {  
 Socket cliSock;  
  
 if(args.length!=1) {  
 System.out.println("Local port number required at the command line.");  
 System.exit(1);  
 }  
  
 try { sock = new ServerSocket(Integer.parseInt(args[0])); }  
 catch(IOException ex) {  
 System.out.println("Failed to open local port " + args[0]);  
 System.exit(1);  
 }  
 while(true) {   
 cliSock=sock.accept();  
 new Thread(new HttpRequest(cliSock, BASE\_FOLDER)).start();  
 }  
 }   
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\HttpServerFormFileUpload.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\Makefile

P1=HttpServerFormFileUpload  
C2=HTTP  
C3=HttpRequest  
  
all: $(P1).class   
  
$(P1).class: $(P1).java $(C2).class $(C3).class  
 javac $(P1).java  
  
$(C2).class: $(C2).java  
 javac $(C2).java  
  
$(C3).class: $(C3).java  
 javac $(C3).java  
  
clean:  
 rm -f \*.class

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\www\index.html

<html><head><title>Sample HTTP server</title></head>  
<body bgcolor=#C0C0C0><h1>Sample HTTP server - File upload with multipart/form-data</h1>  
<h3>Java version</h3>  
<hr>  
<center><img src=http.gif><br>(http.gif)</center>  
<hr>  
<center>  
<table width=60% border=1 cellpadding=20 cellspacing=20><tr><td>  
<form name=upload enctype="multipart/form-data" method=POST action=upload>  
1st: <input type=file name=filename><br><br>  
2th: <input type=submit value=UPLOAD>  
</form>  
</td><td align=center>  
<form name=list enctype="text/plain" method=POST action=list>  
<button type=submit>LIST UPLOADED FILES</button>  
</form>  
</td></tr></table>  
</center>  
<hr>  
<center><img src=http2.png><br>http2.png</center>  
</body></html>

### [File Ends] sem4pi-22-23-61-master\base.rcomp\http-server-form-file-upload\www\index.html

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-chat\Makefile

P1=TcpChatCli  
P2=TcpChatCliGui  
P3=TcpChatSrv  
P4=TcpChatSrvSingleThread  
  
all: $(P1).class $(P2).class $(P3).class $(P4).class  
  
$(P1).class: $(P1).java  
 javac $(P1).java  
  
$(P2).class: $(P2).java  
 javac $(P2).java  
  
$(P3).class: $(P3).java  
 javac $(P3).java  
  
$(P4).class: $(P4).java  
 javac $(P4).java  
  
  
clean:  
 rm -f \*.class

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-chat\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatCli.java

import java.io.\*;   
import java.net.\*;   
  
class TcpChatCli {   
 static InetAddress serverIP;  
 static Socket sock;  
  
  
 public static void main(String args[]) throws Exception {  
 String nick, frase;  
 byte[] data = new byte[300];  
  
 if(args.length!=1) {  
 System.out.println(  
 "Server IPv4/IPv6 address or DNS name is required as argument");  
 System.exit(1); }  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid server: " + args[0]);  
 System.exit(1); }  
  
 try { sock = new Socket(serverIP, 9999); }  
 catch(IOException ex) {  
 System.out.println("Failed to connect.");  
 System.exit(1); }  
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
 DataOutputStream sOut = new DataOutputStream(sock.getOutputStream());  
  
 System.out.println("Connected to server");  
 System.out.print("Enter your nickname: "); nick = in.readLine();  
  
 // start a thread to read incoming messages from the server  
 Thread serverConn = new Thread(new TcpChatCliConn(sock));  
 serverConn.start();  
  
  
 while(true) { // read messages from the console and send them to the server  
 frase=in.readLine();  
 if(frase.compareTo("exit")==0)   
 { sOut.write(0); break;}  
 frase="(" + nick + ") " + frase;  
 data = frase.getBytes();  
 sOut.write((byte)frase.length());  
 sOut.write(data,0,(byte)frase.length());  
 }  
  
 serverConn.join();  
 sock.close();  
 }   
 }  
  
  
  
class TcpChatCliConn implements Runnable {   
 private Socket s;  
 private DataInputStream sIn;  
  
 public TcpChatCliConn(Socket tcp\_s) { s=tcp\_s;}  
  
 public void run() {  
 int nChars;  
 byte[] data = new byte[300];  
 String frase;  
  
 try {  
 sIn = new DataInputStream(s.getInputStream());  
 while(true) {  
 nChars=sIn.read();  
 if(nChars==0) break;  
 sIn.read(data,0,nChars);  
 frase = new String(data, 0, nChars);  
 System.out.println(frase);  
 }  
 }  
 catch(IOException ex) { System.out.println("Client disconnected."); }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatCli.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatCliGui.java

import java.awt.GridBagConstraints;  
import java.awt.GridBagLayout;  
import java.awt.Insets;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.io.\*;   
import java.net.\*;   
import javax.swing.\*;  
  
  
class TcpChatCliGui extends JFrame implements ActionListener {   
  
 static private JTextField textToSend;  
 static private JTextArea textRec;  
 static private Socket sock;  
 static private InetAddress serverIP;  
 static String server, nick;  
 static private DataOutputStream sOut;  
  
 public TcpChatCliGui() { startGUI(); }  
  
  
  
 private void startGUI() {  
   
 JButton sendB = new JButton("SEND");  
 JButton exitB = new JButton("EXIT");  
 this.setTitle("TcpChatCliGui - " + nick + "@" + server);  
   
 GridBagLayout layout = new GridBagLayout();   
 GridBagConstraints c = new GridBagConstraints();   
 getContentPane().setLayout(layout);   
   
 textToSend = new JTextField(40);  
 textRec = new JTextArea(10,40);  
 textRec.setEditable(false);  
   
   
 c.gridy = 0; // top line   
 c.gridx = 0; // left column  
 c.gridwidth=4; c.insets=new Insets(5,5,5,5); c.fill=GridBagConstraints.BOTH;  
 layout.setConstraints(textRec, c); getContentPane().add(textRec);  
   
 c.gridy=1;  
 layout.setConstraints(textToSend, c); getContentPane().add(textToSend);  
   
 c.gridy=2;c.gridx=1;c.gridwidth=1;  
 layout.setConstraints(sendB, c); getContentPane().add(sendB);  
   
 c.gridx=3;  
 layout.setConstraints(exitB, c); getContentPane().add(exitB);  
   
 sendB.addActionListener(this);  
 exitB.addActionListener(this);  
 textToSend.addActionListener(this);  
   
 //setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);   
 setDefaultCloseOperation(JFrame.DO\_NOTHING\_ON\_CLOSE);  
 pack();   
 textToSend.requestFocusInWindow();  
 setVisible(true);   
 }  
   
  
 @Override  
 public void actionPerformed(ActionEvent event) {   
 byte[] data;  
 String label, frase;  
 try {  
 label=event.getActionCommand();  
 if(label.contentEquals("EXIT")) {sOut.write((byte)0);return;}  
 frase=textToSend.getText();  
 if(frase.isEmpty()) return;  
 textToSend.setText("");  
 frase="(" + nick + ") " + frase;  
 data = frase.getBytes();  
 sOut.write((byte)frase.length());  
 sOut.write(data,0,(byte)frase.length());  
 }  
 catch(IOException ex) {  
 System.out.println("NETWORK I/O ERROR");  
 }  
   
 }   
   
  
  
 public static void main(String[] args) throws Exception {  
 String frase;  
  
 DataInputStream sIn;  
 byte[] data = new byte[300];  
 int nChars;  
  
 server=(String)JOptionPane.showInputDialog(null,   
 "Enter the server's IP address or DNS name",  
 "127.0.0.1");  
 if(server==null) {System.exit(0);}  
  
 if(server.length()<1) {  
 JOptionPane.showMessageDialog(null,  
 "Empty server's IP address or DNS name", "ERROR", JOptionPane.WARNING\_MESSAGE);  
 System.exit(0);}  
  
 try { serverIP = InetAddress.getByName(server); }  
 catch(UnknownHostException ex) {  
 JOptionPane.showMessageDialog(null,  
 "The provided server (" + server + ") is invalid", "ERROR",  
 JOptionPane.WARNING\_MESSAGE);System.exit(0); }  
   
  
 try {  
 sock = new Socket(serverIP, 9999);  
 }  
 catch(IOException ex) {  
 JOptionPane.showMessageDialog(null,  
 "TCP connection with server " + server + " failed", "ERROR",  
 JOptionPane.WARNING\_MESSAGE);System.exit(0); }  
   
  
 do {  
 nick=(String)JOptionPane.showInputDialog(null,  
 "Enter your nickname to be used in server " + server,  
 "ENTER YOUR NICKNAME", JOptionPane.PLAIN\_MESSAGE);  
 }  
 while(nick==null || nick.length()<2);  
  
 sOut = new DataOutputStream(sock.getOutputStream());  
 sIn=new DataInputStream(sock.getInputStream());  
  
 TcpChatCliGui appWin = new TcpChatCliGui();  
  
 while(true) {  
 nChars=sIn.read();  
 if(nChars==0) break;  
 sIn.read(data,0,nChars);  
 frase = new String( data, 0, nChars);  
 textRec.append("\n" + frase);  
 }  
  
 System.out.println("DONE");  
 sock.close();  
 System.exit(0);  
 }   
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatCliGui.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatSrv.java

import java.io.\*;   
import java.net.\*;   
import java.util.concurrent.\*;  
import java.util.HashMap;  
  
class TcpChatSrv {  
  
 private static HashMap<Socket,DataOutputStream> cliList = new HashMap<>();  
  
 public static synchronized void sendToAll(int len, byte[] data) throws Exception {  
 for(DataOutputStream cOut: cliList.values()) {  
 cOut.write(len);  
 cOut.write(data,0,len);  
 }  
  
 }  
   
 public static synchronized void addCli(Socket s) throws Exception {  
 cliList.put(s,new DataOutputStream(s.getOutputStream()));  
 }  
  
 public static synchronized void remCli(Socket s) throws Exception {  
 cliList.get(s).write(0);  
 cliList.remove(s);  
 s.close();  
 }  
  
  
  
 private static ServerSocket sock;  
  
 public static void main(String args[]) throws Exception {  
 int i;  
  
 try { sock = new ServerSocket(9999); }  
 catch(IOException ex) {  
 System.out.println("Local port number not available.");  
 System.exit(1); }  
  
 while(true) {  
 Socket s=sock.accept(); // wait for a new client connection request  
 addCli(s);  
 Thread cli = new TcpChatSrvClient(s);  
 cli.start();  
 }  
 }   
 }  
  
  
  
class TcpChatSrvClient extends Thread {  
 private Socket myS;  
 private DataInputStream sIn;  
  
 public TcpChatSrvClient(Socket s) { myS=s;}  
   
 public void run() {  
 int nChars;  
 byte[] data = new byte[300];  
  
 try {  
 sIn = new DataInputStream(myS.getInputStream());  
 while(true) {  
 nChars=sIn.read();  
 if(nChars==0) break; // empty line means client wants to exit  
 sIn.read(data,0,nChars);  
 TcpChatSrv.sendToAll(nChars,data);  
 }  
 // the client wants to exit  
 TcpChatSrv.remCli(myS);  
 }  
 catch(Exception ex) { System.out.println("Error"); }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatSrv.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatSrvSingleThread.java

import java.io.\*;   
import java.net.\*;   
import java.util.HashMap;  
  
class TcpChatSrvSingleThread {   
  
 private static final int SO\_TIMEOUT = 100;  
  
 private static HashMap<Socket,DataOutputStream> cliListOut = new HashMap<>();  
 private static HashMap<Socket,DataInputStream> cliListIn = new HashMap<>();  
  
  
 private static ServerSocket sock;  
  
 public static void main(String args[]) throws Exception {   
 int nChars;  
 byte[] data = new byte[300];  
 Socket cliS;  
  
 try { sock = new ServerSocket(9999); }  
 catch(IOException ex) {  
 System.out.println("Local port number not available.");  
 System.exit(1); }  
  
 sock.setSoTimeout(SO\_TIMEOUT); // set the socket timeout  
  
 while(true) {  
 try { // check for new client connection requests  
 cliS=sock.accept();  
 cliS.setSoTimeout(SO\_TIMEOUT); // set the connected socket timeout  
 cliListOut.put(cliS,new DataOutputStream(cliS.getOutputStream()));  
 cliListIn.put(cliS,new DataInputStream(cliS.getInputStream()));  
 }  
 catch(SocketTimeoutException to) {} // no new connections  
  
  
 for(Socket s: cliListIn.keySet()) { // all connected clients  
 DataInputStream sIn = cliListIn.get(s);  
 try { // try reading the line size  
 nChars=sIn.read();  
 if(nChars==0) { // empty line - client wants to exit  
 DataOutputStream sOut = cliListOut.get(s);  
 sOut.write(nChars); // send back an empty line  
 cliListIn.remove(s);  
 cliListOut.remove(s);  
 s.close();  
 }   
 else {  
 sIn.read(data,0,nChars); // read the line  
 for(DataOutputStream sOut: cliListOut.values()) {  
 sOut.write(nChars);  
 sOut.write(data,0,nChars);  
 }  
 }  
 }  
 catch(SocketTimeoutException to) {} // no text line from client  
 }  
  
 }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-chat\TcpChatSrvSingleThread.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-cli-srv\Makefile

P1=TcpCliSum  
P2=TcpSrvSum  
  
all: $(P1).class $(P2).class  
  
$(P1).class: $(P1).java  
 javac $(P1).java  
  
$(P2).class: $(P2).java  
 javac $(P2).java  
  
clean:  
 rm -f \*.class

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-cli-srv\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-cli-srv\TcpCliSum.java

import java.io.\*;   
import java.net.\*;   
  
class TcpCliSum {   
 static InetAddress serverIP;  
 static Socket sock;  
  
 public static void main(String args[]) throws Exception {  
 if(args.length!=1) {  
 System.out.println("Server IPv4/IPv6 address or DNS name is required as argument");  
 System.exit(1); }  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid server specified: " + args[0]);  
 System.exit(1); }  
  
 try { sock = new Socket(serverIP, 9999); }  
 catch(IOException ex) {  
 System.out.println("Failed to establish TCP connection");  
 System.exit(1); }  
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
 DataOutputStream sOut = new DataOutputStream(sock.getOutputStream());  
 DataInputStream sIn = new DataInputStream(sock.getInputStream());  
  
  
 String frase;  
 long f,i,n,num;  
 do {  
 do {  
 num=-1;  
 while(num<0) {  
 System.out.print("Enter a positive integer to SUM (zero to terminate): ");  
 frase = in.readLine();  
 try { num=Integer.parseInt(frase); }  
 catch(NumberFormatException ex) {num=-1;}  
 if(num<0) System.out.println("Invalid number");  
 }  
 n=num; for(i=0;i<4;i++) {sOut.write((byte)(n%256)); n=n/256; }  
 }  
 while(num!=0);  
 num=0; f=1;  
 for(i=0;i<4;i++) {num=num+f\*sIn.read(); f=f\*256;}  
 System.out.println("SUM RESULT = " + num);  
 }  
 while(num!=0);  
 sock.close();  
 }   
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-cli-srv\TcpCliSum.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\tcp-cli-srv\TcpSrvSum.java

import java.io.\*;   
import java.net.\*;   
  
class TcpSrvSum {   
 static ServerSocket sock;  
  
 public static void main(String args[]) throws Exception {   
 Socket cliSock;  
  
 try { sock = new ServerSocket(9999); }  
 catch(IOException ex) {  
 System.out.println("Failed to open server socket");  
 System.exit(1);  
 }  
  
 while(true) {  
 cliSock=sock.accept();  
 new Thread(new TcpSrvSumThread(cliSock)).start();  
 }  
 }   
 }  
  
  
  
class TcpSrvSumThread implements Runnable {   
 private Socket s;  
 private DataOutputStream sOut;  
 private DataInputStream sIn;  
  
 public TcpSrvSumThread(Socket cli\_s) { s=cli\_s;}  
  
 public void run() {  
 long f,i,num,sum;  
 InetAddress clientIP;  
  
 clientIP=s.getInetAddress();  
 System.out.println("New client connection from " + clientIP.getHostAddress() +   
 ", port number " + s.getPort());  
 try {  
 sOut = new DataOutputStream(s.getOutputStream());  
 sIn = new DataInputStream(s.getInputStream());  
 do {  
 sum=0;  
 do {  
 num=0; f=1; for(i=0;i<4;i++) {num=num+f\*sIn.read(); f=f\*256;}  
 sum=sum+num;  
 }  
 while(num>0);  
 num=sum; for(i=0;i<4;i++) {sOut.write((byte)(num%256)); num=num/256; }  
 }  
 while(sum>0);  
  
 System.out.println("Client " + clientIP.getHostAddress() + ", port number: " + s.getPort() +   
 " disconnected");  
 s.close();  
 }  
 catch(IOException ex) { System.out.println("IOException"); }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\tcp-cli-srv\TcpSrvSum.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-chat\Makefile

P1=UdpChat  
  
all: $(P1).class  
  
$(P1).class: $(P1).java  
 javac $(P1).java  
  
clean:  
 rm -f \*.class

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-chat\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-chat\UdpChat.java

import java.io.\*;   
import java.net.\*;   
import java.util.HashSet;  
  
class UdpChat {   
  
 private static final String BCAST\_ADDR = "255.255.255.255";  
 private static final int SERVICE\_PORT = 9999;  
  
 private static HashSet<InetAddress> peersList = new HashSet<>();  
  
 public static synchronized void addIP(InetAddress ip) { peersList.add(ip);}  
  
 public static synchronized void remIP(InetAddress ip) { peersList.remove(ip);}  
  
 public static synchronized void printIPs() {   
 for(InetAddress ip: peersList) {  
 System.out.print(" " + ip.getHostAddress());  
 }  
 }  
  
 public static synchronized void sendToAll(DatagramSocket s, DatagramPacket p) throws Exception {   
 for(InetAddress ip: peersList) {  
 p.setAddress(ip);  
 s.send(p);  
 }  
 }  
  
 static InetAddress bcastAddress;  
 static DatagramSocket sock;  
  
 public static void main(String args[]) throws Exception {  
 String nick, frase;  
 byte[] data = new byte[300];  
 byte[] fraseData;  
 int i;  
 DatagramPacket udpPacket;  
  
 try { sock = new DatagramSocket(SERVICE\_PORT); }  
 catch(IOException ex) {  
 System.out.println("Failed to open local port");  
 System.exit(1); }  
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
  
 System.out.print("Nickname: "); nick = in.readLine();  
  
 bcastAddress=InetAddress.getByName(BCAST\_ADDR);  
 sock.setBroadcast(true);  
 data[0]=1;  
 udpPacket = new DatagramPacket(data, 1, bcastAddress, SERVICE\_PORT);  
 sock.send(udpPacket);  
  
 Thread udpReceiver = new Thread(new UdpChatReceive(sock));  
 udpReceiver.start();  
  
 while(true) { // handle user inputs  
 frase=in.readLine();  
 if(frase.compareTo("EXIT")==0) break;  
 if(frase.compareTo("LIST")==0) {  
 System.out.print("Active peers:");  
 printIPs();  
 System.out.println("");  
 }  
 else {  
 frase="(" + nick + ") " + frase;  
 fraseData = frase.getBytes();  
 udpPacket.setData(fraseData);  
 udpPacket.setLength(frase.length());  
 sendToAll(sock,udpPacket);  
 }  
 }  
 data[0]=0; // announce I'm leaving  
 udpPacket.setData(data);  
 udpPacket.setLength(1);  
 sendToAll(sock,udpPacket);  
 sock.close();  
 udpReceiver.join(); // wait for the thread to end  
 }   
 }  
  
  
  
class UdpChatReceive implements Runnable {   
 private DatagramSocket s;  
  
 public UdpChatReceive(DatagramSocket udp\_s) { s=udp\_s;}  
  
 public void run() {  
 int i;  
 byte[] data = new byte[300];  
 String frase;  
 DatagramPacket p;  
 InetAddress currPeerAddress;  
  
 p=new DatagramPacket(data, data.length);  
  
 while(true) {  
 p.setLength(data.length);  
 try { s.receive(p); }  
 catch(IOException ex) { return; }  
 currPeerAddress=p.getAddress();  
   
 if(data[0]==1) { // peer start  
 UdpChat.addIP(p.getAddress());  
 try { s.send(p); }   
 catch(IOException ex) { return; }  
 }  
 else  
 if(data[0]==0) { // peer exit  
 UdpChat.remIP(p.getAddress());  
 }  
 else { // chat message  
 frase = new String( p.getData(), 0, p.getLength());  
 System.out.println(frase);  
 }  
 }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-chat\UdpChat.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\Makefile

CC=javac  
  
P1=UdpCliBcast  
P2=UdpCli  
P3=UdpCliTo  
P4=UdpSrv  
P5=UdpSrvMport  
  
all: $(P1).class $(P2).class $(P3).class $(P3).class $(P4).class $(P5).class  
  
$(P1).class: $(P1).java  
 $(CC) $(P1).java  
  
$(P2).class: $(P2).java  
 $(CC) $(P2).java  
  
$(P3).class: $(P3).java  
 $(CC) $(P3).java  
  
$(P4).class: $(P4).java  
 $(CC) $(P4).java  
  
$(P5).class: $(P5).java  
 $(CC) $(P5).java  
  
clean:  
 rm -f \*.class

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\Makefile

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpCli.java

import java.io.\*;   
import java.net.\*;   
  
class UdpCli {   
 static InetAddress serverIP;  
  
 public static void main(String args[]) throws Exception {   
 byte[] data = new byte[300];  
 String frase;  
  
 if(args.length!=1) {  
 System.out.println("Server IPv4/IPv6 address or DNS name is required as argument");  
 System.exit(1);  
 }  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid server address supplied: " + args[0]);  
 System.exit(1);  
 }   
  
 DatagramSocket sock = new DatagramSocket();  
 DatagramPacket udpPacket = new DatagramPacket(data, data.length, serverIP, 9999);  
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
  
 while(true) {  
 System.out.print("Sentence to send (\"exit\" to quit): ");  
 frase = in.readLine();  
 if(frase.compareTo("exit")==0) break;  
 udpPacket.setData(frase.getBytes());  
 udpPacket.setLength(frase.length());  
 sock.send(udpPacket);  
 udpPacket.setData(data);  
 udpPacket.setLength(data.length);  
 sock.receive(udpPacket);  
 frase = new String( udpPacket.getData(), 0, udpPacket.getLength());  
 System.out.println("Received reply: " + frase);  
 }  
 sock.close();  
 }   
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpCli.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpCliBcast.java

import java.io.\*;  
import java.net.\*;  
  
class UdpCliBcast {  
 static InetAddress targetIP;  
  
 public static void main(String args[]) throws Exception {  
 byte[] data = new byte[300];  
 String frase;  
 targetIP=InetAddress.getByName("255.255.255.255");  
  
 DatagramSocket sock = new DatagramSocket();  
 sock.setBroadcast(true);  
 DatagramPacket udpPacket = new DatagramPacket(data, data.length, targetIP, 9999);  
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
  
 while(true) {  
 System.out.print("Request sentence to send (\"exit\" to quit): ");  
 frase = in.readLine();  
 if(frase.compareTo("exit")==0) break;  
 udpPacket.setData(frase.getBytes());  
 udpPacket.setLength(frase.length());  
 sock.send(udpPacket);  
 udpPacket.setData(data);  
 udpPacket.setLength(data.length);  
 sock.receive(udpPacket);  
 frase = new String( udpPacket.getData(), 0, udpPacket.getLength());  
 System.out.println("Received reply: " + frase);  
 }  
 sock.close();  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpCliBcast.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpCliTo.java

import java.io.\*;   
import java.net.\*;   
  
class UdpCliTo {   
 static InetAddress serverIP;  
  
 private static final int TIMEOUT=3; // seconds  
  
 public static void main(String args[]) throws Exception {   
 byte[] data = new byte[300];  
 String frase;  
  
 if(args.length!=1) {  
 System.out.println("Server IPv4/IPv6 address or DNS name is required as argument");  
 System.exit(1);  
 }  
  
 try { serverIP = InetAddress.getByName(args[0]); }  
 catch(UnknownHostException ex) {  
 System.out.println("Invalid server address supplied: " + args[0]);  
 System.exit(1);  
 }   
  
 BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
 DatagramSocket sock = new DatagramSocket();  
 sock.setSoTimeout(1000\*TIMEOUT); // set the socket timeout  
  
 DatagramPacket udpPacket = new DatagramPacket(data, data.length, serverIP, 9999);  
  
 while(true) {  
 System.out.print("Request sentence to send (\"exit\" to quit): ");  
 frase = in.readLine();  
 if(frase.compareTo("exit")==0) break;  
 udpPacket.setData(frase.getBytes());  
 udpPacket.setLength(frase.length());  
 sock.send(udpPacket);  
 udpPacket.setData(data);  
 udpPacket.setLength(data.length);  
 try {  
 sock.receive(udpPacket);  
 frase = new String( udpPacket.getData(), 0, udpPacket.getLength());  
 System.out.println("Received reply: " + frase);  
 }  
 catch(SocketTimeoutException ex) {  
 System.out.println("No reply from server");}  
 }  
 sock.close();  
 }   
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpCliTo.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpSrv.java

import java.io.\*;  
import java.net.\*;  
  
class UdpSrv {  
 static DatagramSocket sock;  
 public static void main(String args[]) throws Exception {  
 byte[] data = new byte[300];  
 byte[] data1 = new byte[300];  
 int i, len;  
  
 try { sock = new DatagramSocket(9999); }  
 catch(BindException ex) {  
 System.out.println("Bind to local port failed");  
 System.exit(1);  
 }  
  
 DatagramPacket udpPacket= new DatagramPacket(data, data.length);  
  
 System.out.println("Listening for UDP requests (IPv6/IPv4). Use CTRL+C to terminate the server");  
 while(true) {  
 udpPacket.setData(data);  
 udpPacket.setLength(data.length);  
 sock.receive(udpPacket);  
 len=udpPacket.getLength();  
 System.out.println("Request from: " + udpPacket.getAddress().getHostAddress() +  
 " port: " + udpPacket.getPort());  
 for(i=0;i<len;i++) data1[len-1-i]=data[i];  
 udpPacket.setData(data1);  
 udpPacket.setLength(len);  
 sock.send(udpPacket);  
 }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpSrv.java

### [File Begins] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpSrvMport.java

import java.io.\*;   
import java.net.\*;   
  
class UdpSrvMport {   
 private static final int SERVER\_PORT\_BASE=9009;  
 static DatagramSocket sock[];  
 public static void main(String args[]) throws Exception {  
 int i;  
 sock = new DatagramSocket[6];  
  
 for(i=0;i<6;i++) {  
 try { sock[i] = new DatagramSocket(SERVER\_PORT\_BASE+100\*i); }  
 catch(BindException ex) {  
 System.out.println("Failed to bind to port number " + SERVER\_PORT\_BASE+100\*i);  
 do { sock[i].close(); i--; } while(i>-1);  
 System.exit(1);  
 }  
 }  
  
  
 System.out.println("Listening for UDP requests (IPv6/IPv4). Use CTRL+C to terminate the server");  
  
 for(i=0;i<6;i++) // start one thread for each socket  
 new Thread(new UdpSrvMportThread(sock[i])).start();  
  
 }  
 }  
  
  
class UdpSrvMportThread implements Runnable {  
 private DatagramSocket sock;  
  
  
 public UdpSrvMportThread(DatagramSocket s) { sock=s;}  
  
 public void run() {  
 byte[] data = new byte[300];  
 byte[] data1 = new byte[300];  
 String frase;  
 int len, i;  
  
 DatagramPacket udpPacket = new DatagramPacket(data, data.length);  
  
 try {  
 while(true) {  
 udpPacket.setData(data);  
 udpPacket.setLength(data.length);  
 sock.receive(udpPacket);  
 len=udpPacket.getLength();  
 System.out.println("Request from: " + udpPacket.getAddress().getHostAddress() +  
 " port: " + udpPacket.getPort());  
 for(i=0;i<len;i++) data1[len-1-i]=data[i];  
 udpPacket.setData(data1);  
 udpPacket.setLength(len);  
 sock.send(udpPacket);  
 }  
 }  
 catch(IOException ex) { System.out.println("IOException"); }  
 }  
 }

### [File Ends] sem4pi-22-23-61-master\base.rcomp\udp-cli-srv\UdpSrvMport.java

### [File Begins] sem4pi-22-23-61-master\bitbucket-pipelines.yml

# Template maven-build  
  
# This template allows you to test and build your Java project with Maven.  
# The workflow allows running tests, code checkstyle and security scans on the default branch.  
  
# Prerequisites: pom.xml and appropriate project structure should exist in the repository.  
  
image: maven:3.6.3  
  
pipelines:  
 default:  
 - step:   
 runs-on:  
 - 'self.hosted'  
 - 'linux'   
 name: Build and Test  
# caches:  
# - maven  
 clone:  
 depth: full # SonarCloud scanner needs the full history to assign issues properly   
 script:  
 - mvn -B verify org.sonarsource.scanner.maven:sonar-maven-plugin:sonar   
 - step:  
 runs-on:  
 - 'self.hosted'  
 - 'linux'   
 name: Chekstyle  
# caches:  
# - maven  
 clone:  
 depth: full # SonarCloud scanner needs the full history to assign issues properly   
 script:  
 - mvn checkstyle:checkstyle   
 after-script: # Collect checkstyle results, if any, and convert to Bitbucket Code Insights.  
 - pipe: atlassian/checkstyle-report:0.3.0  
 - step:  
 runs-on:  
 - 'self.hosted'  
 - 'linux'   
 name: Gitinspector  
# caches:  
# - maven  
 clone:  
 depth: full   
 services:   
 - docker  
 script:  
 - docker version  
 - pwd   
 - ls -al  
 - docker run --rm -v /opt/atlassian/pipelines/agent/build:/repo felix/gitinspector:0.4.4 --format=html --grading > gitinspector.html  
 artifacts:  
 - gitinspector.html

### [File Ends] sem4pi-22-23-61-master\bitbucket-pipelines.yml

### [File Begins] sem4pi-22-23-61-master\docs\Divisao\_Tarefas\Divisao\_Tarefas

Divisão das US Sprint C  
  
LAPR4 - Daniel, João Cruz, João Maria, Miguel, José  
EAPLI - Daniel, João Cruz, João Maria, Miguel, José  
SCOMP - Daniel, Miguel, João Maria  
RCOMP - João Cruz, José  
LPROG - João Cruz, João Maria, Miguel, José  
  
US3004 - Miguel  
US3005 - Todos de SCOMP + Todos de RCOMP  
US3006 - Daniel  
US3007 - Daniel  
US3008 - João Maria  
US3009 - João Maria  
US3010 - Miguel  
  
US2004 - João Cruz, João Maria  
US2009 - Miguel, José  
  
US2005 - João Cruz  
US2006 - José  
  
US4003 - Miguel  
US4003 - Daniel  
US4004 - João Maria  
  
US5002 - Todos de LAPR4

### [File Ends] sem4pi-22-23-61-master\docs\Divisao\_Tarefas\Divisao\_Tarefas

### [File Begins] sem4pi-22-23-61-master\docs\Domain Model\US-G002\domain-model.puml

@startuml  
 hide methods  
 top to bottom direction  
 skinparam linetype ortho  
  
  
 package Exam{  
 class Exam << (E,#FF7700) Entity >> {  
 }  
 class Open\_Date<< (V,#008B8B) Value\_Object >>{  
 }  
 class Close\_Date<< (V,#008B8B) Value\_Object >>{  
 }  
 class Exam\_Result << (E,#FF7700) Entity >>{  
 }  
 class DashBoard << (E,#FF7700) Entity >>{  
 }  
 class Grammar << (V,#008B8B) Value\_Object >>{  
 }  
 Exam "1" --> "1" Grammar : has  
 Exam "1" --> "1" Open\_Date : has  
 Exam "1" --> "1" Close\_Date : has  
 Exam "1" --> "1" Exam\_Result : has  
 Exam\_Result "\*" --> "1" DashBoard : constitutes  
 }  
  
 package Post\_It{  
 class Post\_It << (E,#FF7700) Entity >>{  
 }  
 class Column << (V,#008B8B) Value\_Object >> {  
 }  
 class Row<< (V,#008B8B) Value\_Object >> {  
 }  
 class Content<< (V,#008B8B) Value\_Object >> {  
 }  
 Post\_It "1" --> "1" Content : has  
 Post\_It "1" --> "1" Row : has  
 Post\_It "1" --> "1" Column : has  
 }  
  
 package Class{  
 class Class << (E,#FF7700) Entity >>{  
 }  
 class Class\_Title << (V,#008B8B) Unique\_Value\_Object >> {  
 }  
 class Class\_Start\_Date << (V,#008B8B) Value\_Object >> {  
 }  
 class Class\_Finish\_Date << (V,#008B8B) Value\_Object >> {  
 }  
 class Class\_Start\_Time << (V,#008B8B) Value\_Object >> {  
 }  
 class Class\_Finish\_Time << (V,#008B8B) Value\_Object >> {  
 }  
 class Class\_Day\_Of\_Week << (V,#008B8B) Value\_Object >> {  
 }  
 Class "1" --> "1" Class\_Title : has  
 Class "1" --> "1" Class\_Day\_Of\_Week : has  
 Class "1" --> "1" Class\_Start\_Date : has  
 Class "1" --> "1" Class\_Finish\_Date : has  
 Class "1" --> "1" Class\_Start\_Time : has  
 Class "1" --> "1" Class\_Finish\_Time : has  
 }  
  
 package Course {  
 class Course << (E,#FF7700) Entity >> {  
 }  
 class Minimum\_Number\_Of\_Students<< (V,#008B8B) Value\_Object >>{  
 }  
 class Maximum\_Number\_Of\_Students<< (V,#008B8B) Value\_Object >>{  
 }  
 class Small\_Textual\_Description<< (V,#008B8B) Value\_Object >>{  
 }  
 class Course\_ID<< (V,#008B8B) Unique\_Value\_Object >>{  
 }  
 class Course\_Name<< (V,#008B8B) Value\_Object >>{  
 }  
 Course "1" --> "1" Minimum\_Number\_Of\_Students : has  
 Course "1" --> "1" Maximum\_Number\_Of\_Students : has  
 Course "1" --> "1" Small\_Textual\_Description : has  
 Course "1" --> "1" Course\_Name : has  
 Course "1" --> "1" Course\_ID : has  
 }  
  
 package Invite{  
 class Invite << (E,#FF7700) Entity >> {  
 }  
 }  
  
 package Meeting{  
 class Meeting << (E,#FF7700) Entity >> {  
 }  
 class Meeting\_Duration << (V,#008B8B) Value\_Object >> {  
 }  
 class Meeting\_Date << (V,#008B8B) Value\_Object >> {  
 }  
 class Meeting\_Time << (V,#008B8B) Value\_Object >> {  
 }  
 class Meeting\_ID << (V,#008B8B) Unique\_Value\_Object >> {  
 }  
 Meeting "1" --> "\*" Invite : sends  
 Meeting "1" --> "1" Meeting\_Duration : has  
 Meeting "1" --> "1" Meeting\_Date : has  
 Meeting "1" --> "1" Meeting\_Time : has  
 Meeting "1" --> "1" Meeting\_ID : has  
 }  
  
 package Shared\_Board{  
 class Shared\_Board << (E,#FF7700) Entity >> {  
 }  
 class Number\_of\_Columns << (V,#008B8B) Value\_Object >> {  
 }  
 class Number\_of\_Rows << (V,#008B8B) Value\_Object >> {  
 }  
 class Shared\_Board\_Title << (V,#008B8B) Value\_Object >> {  
 }  
 class Shared\_Board\_ID << (V,#008B8B) Value\_Object >> {  
 }  
 class sharedBoardHistory<< (E,#FF7700) Entity >> {  
 }  
 Shared\_Board "1" --> "1" Number\_of\_Columns : has  
 Shared\_Board "1" --> "1" Number\_of\_Rows : has  
 Shared\_Board "1" --> "1" Shared\_Board\_Title : has  
 Shared\_Board "1" --> "1" Shared\_Board\_ID : has  
 Shared\_Board "1" --> "1" sharedBoardHistory : has  
 }  
  
 package Person {  
 class Person << (E,#FF7700) Entity >> {  
 }  
 class Email << (V,#008B8B) Value\_Object >> {  
 }  
 class Password << (V,#008B8B) Value\_Object >> {  
 }  
 class Full\_Name << (V,#008B8B) Value\_Object >> {  
 }  
 class Short\_Name << (V,#008B8B) Value\_Object >> {  
 }  
 Person "1" --> "1" Email : has  
 Person "1" --> "1" Password : has  
 Person "1" --> "1" Full\_Name : has  
 Person "1" --> "1" Short\_Name : has  
 }  
  
 package Enrollment{  
 class Enrollment << (E,#FF7700) Entity >> {  
 }  
 }  
  
 package Teacher\_Students {  
 class Date\_of\_Birth << (V,#FF7700) Value\_Object >> {  
 }  
 class Tax\_Payer\_Number << (V,#FF7700) Value\_Object >> {  
 }  
  
 package Student {  
 class Student << (E,#FF7700) Entity >> {}  
 class Mechanographic\_Number << (V,#FF7700) Value\_Object >> {}  
 Student "1" --> "1" Mechanographic\_Number : has  
 Student "1" --> "1" Date\_of\_Birth : has  
 Student "1" --> "1" Tax\_Payer\_Number : has  
 }  
  
 package Teacher {  
 class Teacher << (E,#FF7700) Entity >> {}  
 class Acronym << (V,#FF7700) Value\_Object >> {}  
 Teacher "1" --> "1" Acronym : has  
 Teacher "1" --> "1" Date\_of\_Birth : has  
 Teacher "1" --> "1" Tax\_Payer\_Number : has  
 Teacher "1" --> "\*" DashBoard : checks  
 Teacher "1" --> "\*" Exam : creates  
 Course "1" --> "\*" Teacher : has  
 Teacher "1" --> "\*" Class : Schedules  
 Teacher "1" --> "\*" Extra\_Class : Schedules  
 }  
 }  
  
 package Extra\_Class{  
 class Extra\_Class << (E,#FF7700) Entity >> {  
 }  
 class Extra\_Class\_Day<< (V,#008B8B) Value\_Object >>{  
 }  
 class Extra\_Class\_Title<< (V,#008B8B) Value\_Object >>{  
 }  
 class Extra\_Class\_Start\_Time<< (V,#008B8B) Value\_Object >>{  
 }  
 class Extra\_Class\_Finish\_Time<< (V,#008B8B) Value\_Object >>{  
 }  
 Extra\_Class "1" --> "1" Extra\_Class\_Title : has  
 Extra\_Class "1" --> "1" Extra\_Class\_Day : has  
 Extra\_Class "1" --> "1" Extra\_Class\_Start\_Time : has  
 Extra\_Class "1" --> "1" Extra\_Class\_Finish\_Time : has  
 }  
  
 Student "1" --> "\*" DashBoard : checks  
 Student "1" --> "\*" Exam : takes  
 Student "1" --> "\*" Class : takes  
 Student "1" --> "\*" Extra\_Class : takes  
 Course "1" --> "\*" Exam : has  
 Person "1" --> "\*" Enrollment : requests  
 Person "1" --> "\*" Meeting : schedules  
 Person "1" --> "1" Invite : accepts  
 Person "1" --> "\*" Post\_It : creates  
 Person "1" --> "\*" Shared\_Board : creates  
 Person "1" --> "\*" Shared\_Board : edits  
 Person "1" --> "1" sharedBoardHistory : checks  
 Person <|-- Student  
 Person <|-- Teacher  
  
 @enduml

### [File Ends] sem4pi-22-23-61-master\docs\Domain Model\US-G002\domain-model.puml

### [File Begins] sem4pi-22-23-61-master\docs\Domain Model\US-G002\readme.md

# US 1001  
  
\*This is an example template\*  
  
## 1. Context  
  
\*Explain the context for this task. It is the first time the task is assigned to be developed or this tasks was incomplete in a previous sprint and is to be completed in this sprint? Are we fixing some bug?\*  
  
## 2. Requirements  
  
\*In this section you should present the functionality that is being developed, how do you understand it, as well as possible correlations to other requirements (i.e., dependencies).\*  
  
\*Example\*  
  
\*\*US G002\*\* As {Ator} I Want...  
  
- G002.1. Blá Blá Blá ...  
  
- G002.2. Blá Blá Blá ...  
  
\*Regarding this requirement we understand that it relates to...\*  
  
## 3. Analysis  
  
\*In this section, the team should report the study/analysis/comparison that was done in order to take the best design decisions for the requirement. This section should also include supporting diagrams/artifacts (such as domain model; use case diagrams, etc.),\*  
  
## 4. Design  
  
\*In this sections, the team should present the solution design that was adopted to solve the requirement. This should include, at least, a diagram of the realization of the functionality (e.g., sequence diagram), a class diagram (presenting the classes that support the functionality), the identification and rational behind the applied design patterns and the specification of the main tests used to validade the functionality.\*  
  
### 4.1. Realization  
  
### 4.2. Class Diagram  
  
![a class diagram](domain-model.svg "A Class Diagram")  
  
### 4.3. Applied Patterns  
  
### 4.4. Tests  
  
\*\*Test 1:\*\* \*Verifies that it is not possible to create an instance of the Example class with null values.\*  
  
```  
@Test(expected = IllegalArgumentException.class)  
public void ensureNullIsNotAllowed() {  
 Example instance = new Example(null, null);  
}  
````  
  
## 5. Implementation  
  
\*In this section the team should present, if necessary, some evidencies that the implementation is according to the design. It should also describe and explain other important artifacts necessary to fully understand the implementation like, for instance, configuration files.\*  
  
\*It is also a best practice to include a listing (with a brief summary) of the major commits regarding this requirement.\*  
  
## 6. Integration/Demonstration  
  
\*In this section the team should describe the efforts realized in order to integrate this functionality with the other parts/components of the system\*  
  
\*It is also important to explain any scripts or instructions required to execute an demonstrate this functionality\*  
  
## 7. Observations  
  
\*This section should be used to include any content that does not fit any of the previous sections.\*  
  
\*The team should present here, for instance, a critical prespective on the developed work including the analysis of alternative solutioons or related works\*  
  
\*The team should include in this section statements/references regarding third party works that were used in the development this work.\*

### [File Ends] sem4pi-22-23-61-master\docs\Domain Model\US-G002\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\Group\_Elements\1200614\readme.md

### [File Ends] sem4pi-22-23-61-master\docs\Group\_Elements\1200614\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\Group\_Elements\1200801\readme.md

### [File Ends] sem4pi-22-23-61-master\docs\Group\_Elements\1200801\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\Group\_Elements\1200874\readme.md

### [File Ends] sem4pi-22-23-61-master\docs\Group\_Elements\1200874\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\Group\_Elements\1201718\readme.md

# Student: \*Miguel Oliveira - 1200874\*  
  
## Developed Tasks  
  
  
| Sprint | Task |  
|--------|-------------------------------------------|  
| \*\*B\*\* | [US1010] |  
| \*\*B\*\* | [US1011] |  
| \*\*B\*\* | [US1012] |  
| \*\*B\*\* | [US2001] |  
| \*\*B\*\* | [US2007] |  
| \*\*B\*\* | [US2008] |  
| \*\*C\*\* | [US3004] |  
| \*\*C\*\* | [US3010] |  
| \*\*C\*\* | [US4002](../../US4002/US4002-ANALYSIS.md) |

### [File Ends] sem4pi-22-23-61-master\docs\Group\_Elements\1201718\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\Group\_Elements\1210965\readme.md

# Student: \*José Teixeira - 1210965\*  
  
## Developed Tasks  
  
  
| Sprint | Task|  
|--------|--------------------|  
| \*\*B\*\* | [US1007](../../US1007/readme.md) |  
| \*\*B\*\* | [US1008](../../US1008/readme.md) |  
| \*\*B\*\* | [US1009](../../US1009/readme.md) |

### [File Ends] sem4pi-22-23-61-master\docs\Group\_Elements\1210965\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\US1007\US1007-ANALYSIS.md

US1007:  
As Manager, I want to enroll students in bulk by importing their data using a csv file.  
  
  
US1007 BUSINESS RULES:  
1. Todas as regras de negócio da US1008, acerca de um enrollment, aplicam-se á US1007.  
2. O usuário tem de apresentar o caminho a um ficheiro csv com dados válidos.

### [File Ends] sem4pi-22-23-61-master\docs\US1007\US1007-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US1007\US1007-CD.puml

@startuml  
  
class EnrollStudentsBulkUI{  
}  
  
class EnrollStudentsBulkController{  
}  
  
class Enrollment{  
}  
  
class CSVReader{  
}  
  
class Student {  
  
}  
  
class EnrollmentRepository{  
}  
  
EnrollStudentsBulkUI ..> EnrollStudentsBulkController  
EnrollStudentsBulkController ..>Enrollment  
EnrollStudentsBulkController ..> CSVReader  
EnrollStudentsBulkController ..> Student  
EnrollStudentsBulkController ..> StudentRepository  
Enrollment ..> EnrollmentRepository  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1007\US1007-CD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1007\US1007-DM.puml

@startuml  
 hide methods  
  
  
 package Manager{  
 class Manager << (E,#FF7700) Entity >> {  
 }  
 }  
  
  
 package Enrollment{  
 class Enrollment << (E,#FF7700) Entity >> {  
 }  
 }  
  
  
 Enrollment "\*" -- "1" Manager : is accepted by  
  
 @enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1007\US1007-DM.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1007\US1007-SD.puml

@startuml US09-SD  
autonumber  
actor "Manager" as MAN  
participant "Controller" as CTRL  
participant "Store" as STR  
participant "Repository" as REP  
  
activate MAN  
MAN -> ":System" : Asks to enroll students in bulk  
activate ":System"  
":System" --> MAN : Requests the csv file containing the data  
deactivate ":System"  
MAN -> ":System" : Inserts csv file  
activate ":System"  
":System" --> MAN : Confirms/Declines operation  
deactivate ":System"  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1007\US1007-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1007\US1007-SSD.puml

@startuml US1007-SSD  
  
autonumber  
actor "Manager" as MAN  
  
  
activate MAN  
MAN -> ":System" : Asks to enroll students in bulk  
activate ":System"  
":System" --> MAN : Requests the csv file containing the data  
deactivate ":System"  
MAN -> ":System" : Inserts csv file  
activate ":System"  
":System" --> MAN : Confirms/Declines operation  
deactivate ":System"  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1007\US1007-SSD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1007\readme.md

# US 1007  
  
  
## 1. Context  
  
This is the beginning of the US 1007  
  
## 2. Requirements  
  
\*In this section you should present the functionality that is being developed, how do you understand it, as well as possible correlations to other requirements (i.e., dependencies).\*  
  
\*Example\*  
  
\*\*US1007\*\* As Manager, I want to enroll students in bulk by importing their data using a csv file  
  
This US has no dependencies. It is only required a csv file with data to import  
  
  
## 3. Analysis  
Regras de negócio:  
Testes unitários:  
## 4. Design  
  
### 4.1. System Sequence Diagram  
[US1007-SSD](../US1007/US1007-SSD.puml)  
### 4.2. Domain Model  
[US1007-DM](../US1007/US1007-DM.puml)  
### 4.3. Sequence Diagram  
[US1007-SD](../US1007/US1007-SD.puml)  
### 4.4. Class Diagram  
[US1007-CD](../US1007/US1007-CD.puml)  
  
  
  
  
  
  
  
  
### 4.4. Tests  
  
\*\*Test 1:\*\* \*Verifies that it is not possible to create an instance of the Example class with null values.\*  
  
````  
@Test(expected = IllegalArgumentException.class)  
public void ensureNullIsNotAllowed() {  
 Example instance = new Example(null, null);  
}  
````  
  
## 5. Implementation  
  
\*In this section the team should present, if necessary, some evidencies that the implementation is according to the design. It should also describe and explain other important artifacts necessary to fully understand the implementation like, for instance, configuration files.\*  
  
\*It is also a best practice to include a listing (with a brief summary) of the major commits regarding this requirement.\*  
  
## 6. Integration/Demonstration  
  
\*In this section the team should describe the efforts realized in order to integrate this functionality with the other parts/components of the system\*  
  
\*It is also important to explain any scripts or instructions required to execute an demonstrate this functionality\*  
  
## 7. Observations  
  
\*This section should be used to include any content that does not fit any of the previous sections.\*  
  
\*The team should present here, for instance, a critical prespective on the developed work including the analysis of alternative solutioons or related works\*  
  
\*The team should include in this section statements/references regarding third party works that were used in the development this work.\*

### [File Ends] sem4pi-22-23-61-master\docs\US1007\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\US1008\US-1008-ANALYSIS.md

US1008:  
As Student, I want to request my enrollment in a course  
  
  
US1008 BUSINESS RULES:  
1. O usuário tem de fazer log-in como estudante.  
2. Um enrollment tem um id.  
3. Um enrollment é pedido por um estudante.  
4. Um enrollment tem um Curso associado, escolhido pelo estudante.  
5. Um enrollment tem uma descrição.  
6. Um enrollment tem um estado(Pendente, aceite ou rejeitado).  
  
  
US1008 PRE-REQUISITES:  
1. Tem de existir pelo menos um curso no sistema.

### [File Ends] sem4pi-22-23-61-master\docs\US1008\US-1008-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US1008\US1008-CD.puml

@startuml  
'https://plantuml.com/sequence-diagram  
  
autonumber  
  
Alice -> Bob: Authentication Request  
Bob --> Alice: Authentication Response  
  
Alice -> Bob: Another authentication Request  
Alice <-- Bob: another authentication Response  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1008\US1008-CD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1008\US1008-DM.puml

@startuml  
 hide methods  
  
 package Enrollment{  
 class Enrollment << (E,#FF7700) Entity >> {  
 }  
 }  
  
  
  
  
 package Student {  
 class Date\_of\_Birth << (V,#FF7700) Value\_Object >> {}  
 class Tax\_Payer\_Number << (V,#FF7700) Value\_Object >> {}  
 class Student << (E,#FF7700) Entity >> {}  
 class Mechanographic\_Number << (V,#FF7700) Value\_Object >> {}  
 Student "1" -- "1" Mechanographic\_Number  
 Student "1" -- "1" Date\_of\_Birth  
 Student "1" -- "1" Tax\_Payer\_Number  
 }  
  
  
  
  
  
  
  
  
  
 Student "1" -- "\*" Enrollment : requests  
  
  
  
 @enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1008\US1008-DM.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1008\US1008-SD.puml

@startuml US1007-SD  
autonumber  
actor "Student" as STUD  
  
participant ":StudentEnrollmentUI" as UI  
participant ":StudentEnrollmentController" as CTRL  
participant ":StudentEnrollmentFactory" as FACT  
participant "StudentEnrollment" as ENRO  
participant "TrasactionalContext" as CONT  
participant "StudentEnrollmentRepository" as REPO  
  
  
activate STUD  
STUD -> UI : asks to enroll in a course  
activate UI  
  
UI --> STUD : show list of courses and asks to choose one  
deactivate UI  
STUD --> UI : chooses one course to enroll  
activate UI  
UI --> STUD : asks for the required data(x,y,z)  
deactivate UI  
STUD --> UI : inserts the required data(x,y,z)  
create CTRL  
activate UI  
UI --> CTRL : initializes controller  
UI --> CTRL : creates enrollment(x,y,z)  
deactivate UI  
activate CTRL  
CTRL --> FACT : creates enrollment(x,y,z)  
deactivate CTRL  
activate FACT  
create ENRO  
FACT --> ENRO : creates  
deactivate FACT  
CTRL --> CONT : begin  
activate CTRL  
activate CONT  
CTRL --> CONT : save  
CTRL --> REPO : save  
activate REPO  
deactivate REPO  
CTRL --> CONT : commit  
deactivate CONT  
deactivate CTRL  
  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1008\US1008-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1008\US1008-SSD.puml

@startuml US1008-SSD  
  
  
autonumber  
actor "Student" as STU  
  
  
activate STU  
STU -> ":System" : Asks to enroll in a course  
activate ":System"  
":System" --> STU : Shows a list of courses and asks to choose one  
deactivate ":System"  
STU -> ":System" : Chooses one course to enroll  
activate ":System"  
":System" --> STU : Asks for the required data  
deactivate ":System"  
STU -> ":System" : Inserts the required data(x,y,z)  
activate ":System"  
":System" --> STU : Confirms/Declines operation(x,y,z)  
deactivate ":System"  
  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1008\US1008-SSD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1008\readme.md

### [File Ends] sem4pi-22-23-61-master\docs\US1008\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\US1009\US1009-ANALYSIS.md

US1009:  
As Manager, I want to approve or reject students applications to courses.  
  
  
US1009 BUSINESS RULES:  
1. O usuário tem de fazer log-in como manager  
  
US1009 PRE-REQUISITES:  
1. Tem de existir pelo menos um enrollment pendente no sistema.

### [File Ends] sem4pi-22-23-61-master\docs\US1009\US1009-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US1009\US1009-CD.puml

@startuml  
  
class ApproveRejectApplicationsUI{}  
  
class ApproveRejectApplicationsController{  
+createVaccinationCenter(name,address,phone number,e-mail address,\n fax number, website address,opening and closing hours,slot duration,\n coordinator, maximum number of vaccines per slot)  
+saveVaccinationCenter()  
}  
  
  
class VaccinationCenterStore{  
+craeteVaccinationCenter(name,address,phone number,e-mail address,\n fax number, website address,opening and closing hours,slot duration,\n coordinator, maximum number of vaccines per slot)  
+validateVaccinationCenter()  
+saveVaccinationCenter()  
+addVaccinationCenter()  
}  
  
class Enrollment{  
- name  
- address  
- phoneNumber  
- e-mailAddress  
- faxNumber  
- websiteAddress  
- openingHour  
- closingHour  
- slotDuration  
- coordinator  
- maximumNumberOfVaccinesPerSlot  
  
}  
  
ApproveRejectApplicationsUI ..> ApproveRejectApplicationsController  
  
ApproveRejectApplicationsController ..> VaccinationCenterStore  
ApproveRejectApplicationsController ..> Enrollment  
  
VaccinationCenterStore "1" -> "\*" Enrollment : knows >  
  
  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1009\US1009-CD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1009\US1009-DM.puml

@startuml  
  
 hide methods  
  
  
 package Manager{  
 class Manager << (E,#FF7700) Entity >> {  
 }  
 }  
  
  
 package Enrollment{  
 class Enrollment << (E,#FF7700) Entity >> {  
 }  
 }  
  
  
 Enrollment "\*" -- "1" Manager : is accepted by  
  
 @enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1009\US1009-DM.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1009\US1009-SD.puml

@startuml  
'https://plantuml.com/sequence-diagram  
  
autonumber  
  
Alice -> Bob: Authentication Request  
Bob --> Alice: Authentication Response  
  
Alice -> Bob: Another authentication Request  
Alice <-- Bob: another authentication Response  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1009\US1009-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1009\US1009-SSD.puml

@startuml US1009-SSD  
  
autonumber  
actor "Manager" as MAN  
  
  
activate MAN  
MAN -> ":System" : Asks to approve/reject students' applications to courses  
activate ":System"  
":System" --> MAN : Shows a list of students' applications and asks Manager to choose one  
deactivate ":System"  
MAN -> ":System" : chooses one application  
activate ":System"  
":System" --> MAN : Asks to accept/reject the application  
deactivate ":System"  
MAN -> ":System" : accepts/declines the chosen application  
activate ":System"  
":System" --> MAN : Confirms operation  
deactivate ":System"  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1009\US1009-SSD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1009\readme.md

### [File Ends] sem4pi-22-23-61-master\docs\US1009\readme.md

### [File Begins] sem4pi-22-23-61-master\docs\US1010\ANALYSIS.txt

-----------------  
BUSINESS RULES  
-----------------  
  
1. A class schedule has a unique class ID.  
2. A class schedule must have a class title.  
3. A class schedule must have a start time and an end time.  
4. A class schedule must have a start date and an end date.  
5. A class schedule must have a day of the week assigned.  
6. A class schedule must be assigned to a teacher.  
7. The teacher assigned to a class schedule must exist in the system.  
8. A class schedule cannot have overlapping time periods with existing scheduled classes.  
9. A class schedule can optionally have a description.  
10. The class ID of a class schedule must be unique among all scheduled classes.

### [File Ends] sem4pi-22-23-61-master\docs\US1010\ANALYSIS.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1010\DESIGN.txt

-----------------  
DESIGN  
-----------------  
  
To design the class scheduling feature, we can utilize the standard layered architecture of the base application. This use case shares similarities with the "RegisterDishType" use case.  
  
Domain Class: ClassSchedule  
Controller: ScheduleClassController  
Repository: ClassScheduleRepository  
  
The ClassSchedule class will represent the domain entity for a scheduled class, containing attributes such as class ID, title, start time, end time, start date, end date, day of the week, and the associated teacher.  
  
The ScheduleClassController will handle the business logic for scheduling classes. It will interact with the ClassScheduleRepository to persist and retrieve class schedule data.  
  
Following the layered architecture, the presentation layer (e.g., console or GUI) will interact with the ScheduleClassController to collect input from the user and display relevant information.

### [File Ends] sem4pi-22-23-61-master\docs\US1010\DESIGN.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1010\US1010 - TEXT PLAN.txt

TEST PLAN #10 : SHEDULE NEW CLASS  
  
  
  
1 Prompt the user to enter the class ID.  
2 Prompt the user to enter the class title.  
3 Prompt the user to enter the start time of the class in "HH:mm" format.  
4 Prompt the user to enter the end time of the class in "HH:mm" format.  
5 Prompt the user to enter the start date of the class in "YYYY-MM-DD" format.  
6 Prompt the user to enter the end date of the class in "YYYY-MM-DD" format.  
7 Prompt the user to enter the day of the week for the class (1-7).  
8 Prompt the user to enter the acronym of the teacher responsible for the class.  
9 Check if there is a teacher with the provided acronym. Otherwise, display an error message.  
10 Check if there is any time conflict with another existing class. If so, display an error message. Otherwise, proceed to the next step.  
11 Schedule the class by calling the scheduleClass method of ClassController.  
12 Display a success message indicating that the class has been scheduled successfully.

### [File Ends] sem4pi-22-23-61-master\docs\US1010\US1010 - TEXT PLAN.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1010\US1010-SD.puml

@startuml  
'skinparam settings  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor Teacher as Actor  
participant UpdateClassUI as UI  
participant ScheduleClassController as Controller  
participant ClassRepository as Repo  
  
UI -> Actor: Request data (e.g., ID, title, start time, finish time, start date, finish date, day of the week)  
activate UI  
Actor -> UI: Provide requested data  
  
UI -> Controller: scheduleClass( id, title,startTime, finishTime, start\_date, finishDate, day, teacherAcronym)  
activate Controller  
Controller -> Service: scheduleClass(Classe classe)  
activate Service  
Service -> Repo : findAll()  
activate Repo  
deactivate Repo  
Service -> TeacherRepo : findTeacherByAcronym()  
activate TeacherRepo  
deactivate TeacherRepo  
deactivate Service  
alt Does not have conflict  
Controller -> Controller : checkClassConflict()  
Controller -> Controller : isTimeOverlapUserInput()  
Controller -> Repo : save(Classe classe)  
activate Repo  
deactivate Repo  
Controller -> UI :  
deactivate Controller  
  
UI -> Actor : System types : "Class has been scheduled"  
  
else Has conflict  
  
UI -> Actor : System types : "Conflict between Classes"  
  
end  
  
  
  
activate TeacherRepo  
  
deactivate TeacherRepo  
  
deactivate Service  
  
deactivate Controller  
  
  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1010\US1010-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1011\ANALYSIS.txt

-----------------  
BUSINESS RULES  
-----------------  
  
1. A Extraclass schedule has a unique class ID.  
2. A Extraclass schedule must have a class title.  
3. A Extraclass schedule must have a start time and an end time.  
4. A Extraclass schedule must have a start date and an end date.  
5. A Extraclass schedule must have a day of the week assigned.  
6. A Extraclass schedule must be assigned to a teacher.  
7. The teacher assigned to a class schedule must exist in the system.  
8. A Extraclass schedule cannot have overlapping time periods with existing scheduled classes.  
9. A Extraclass schedule can optionally have a description.  
10. The Extraclass ID of a class schedule must be unique among all scheduled classes.

### [File Ends] sem4pi-22-23-61-master\docs\US1011\ANALYSIS.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1011\DESIGN.txt

-----------------  
DESIGN  
-----------------  
  
To design the class scheduling feature, we can utilize the standard layered architecture of the base application. This use case shares similarities with the "RegisterDishType" use case.  
  
Domain Class: Classe  
Controller: ScheduleClassController  
Repository: ClassScheduleRepository  
  
The ClassSchedule class will represent the domain entity for a scheduled class, containing attributes such as class ID, title, start time, end time, start date, end date, day of the week, and the associated teacher.  
  
The ScheduleClassController will handle the business logic for scheduling classes. It will interact with the ClassScheduleRepository to persist and retrieve class schedule data.  
  
Following the layered architecture, the presentation layer (e.g., console or GUI) will interact with the ScheduleClassController to collect input from the user and display relevant information.

### [File Ends] sem4pi-22-23-61-master\docs\US1011\DESIGN.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1011\US1011- TEXT PLAN.txt

TEST PLAN #10: Update Class Schedule  
  
1 Log in as teacher/teacherA1  
2 Select the option Class > Update Class Schedule  
3 List the available classes using the provided option  
4 Input the ID of the class you want to update  
5 Input the new class start time in the format HH:mm  
6 Input the new class finish time in the format HH:mm  
7 Input the new class start date in the format yyyy/mm/dd  
8 Input the new class finish date in the format yyyy/mm/dd  
9 Input the new class teacher

### [File Ends] sem4pi-22-23-61-master\docs\US1011\US1011- TEXT PLAN.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1011\US1011-SD.puml

@startuml  
'skinparam settings  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor Teacher as Actor  
participant UpdateScheduleClassUI as UI  
participant UpdateScheduleClassController as Controller  
participant UpdateScheduleClassService as Service  
participant ClassRepository as Repo  
participant TeacherRepository as TeacherRepo  
  
UI -> Actor: Update the schedule of the class  
activate UI  
Actor -> UI: Provide requested data (id, newStartTime, newFinishTime)  
  
UI -> Controller: updateClassSchedule(id, newStartTime, newFinishTime)  
activate Controller  
Controller -> Service: checkClassSchedule(id, newStartTime, newFinishTime)  
activate Service  
Service -> Repo : findClassById(id)  
activate Repo  
deactivate Repo  
Service -> TeacherRepo : findTeacherByClassId(id)  
activate TeacherRepo  
deactivate TeacherRepo  
deactivate Service  
alt No schedule conflict  
Controller -> Repo : updateClassSchedule(id, newStartTime, newFinishTime)  
activate Repo  
deactivate Repo  
Controller -> UI : Show success message  
UI -> Actor : System types : "Class schedule has been updated"  
  
else Schedule conflict exists  
Controller -> UI : Show error message  
UI -> Actor : System types : "Conflict between class schedules"  
end  
deactivate Controller  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US1011\US1011-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US1012\ANALYSIS.txt

-----------------  
BUSINESS RULES  
-----------------  
  
1. A class schedule has a unique class ID.  
2. A class schedule must have a class title.  
3. A class schedule must have a start time and an end time.  
4. A class schedule must have a start date and an end date.  
5. A class schedule must have a day of the week assigned.  
6. A class schedule must be assigned to a teacher.  
7. The teacher assigned to a class schedule must exist in the system.  
8. A class schedule cannot have overlapping time periods with existing scheduled classes.  
9. A class schedule can optionally have a description.  
10. The class ID of a class schedule must be unique among all scheduled classes.

### [File Ends] sem4pi-22-23-61-master\docs\US1012\ANALYSIS.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1012\DESIGN.txt

-----------------  
DESIGN  
-----------------  
  
To design the class scheduling feature, we can utilize the standard layered architecture of the base application. This use case shares similarities with the "RegisterDishType" use case.  
  
Domain Class: ClassSchedule  
Controller: ScheduleClassController  
Repository: ClassScheduleRepository  
  
The ClassSchedule class will represent the domain entity for a scheduled class, containing attributes such as class ID, title, start time, end time, start date, end date, day of the week, and the associated teacher.  
  
The ScheduleClassController will handle the business logic for scheduling classes. It will interact with the ClassScheduleRepository to persist and retrieve class schedule data.  
  
Following the layered architecture, the presentation layer (e.g., console or GUI) will interact with the ScheduleClassController to collect input from the user and display relevant information.

### [File Ends] sem4pi-22-23-61-master\docs\US1012\DESIGN.txt

### [File Begins] sem4pi-22-23-61-master\docs\US1012\US1012 -TEXT PLAN.txt

TEST PLAN #12 : SHEDULE A NEW EXTRA CLASS  
  
1 Prompt the user to enter the class ID.  
2 Prompt the user to enter the class title.  
3 Prompt the user to enter the start time of the class in "HH:mm" format.  
4 Prompt the user to enter the end time of the class in "HH:mm" format.  
5 Prompt the user to enter the day of the week for the class (1-7).  
6 Prompt the user to enter the acronym of the teacher responsible for the class.  
7 Check if there is a teacher with the provided acronym. Otherwise, display an error message.  
8 Check if there is any time conflict with another existing class. If so, display an error message. Otherwise, proceed to the next step.  
9 Schedule the extra class by calling the scheduleExtraClass method of ExtraClassController.  
10 Display a success message indicating that the class has been scheduled successfully.

### [File Ends] sem4pi-22-23-61-master\docs\US1012\US1012 -TEXT PLAN.txt

### [File Begins] sem4pi-22-23-61-master\docs\US2001\Analysis.txt

Title: Create / Update Exam  
  
  
Description:  
As Teacher, I want to create/update an exam  
  
  
Acceptance Criteria:  
1. The system should allow the teacher to provide the exam title.  
2. The system should allow the teacher to specify the start and end date and time of the exam.  
3. The system should allow the teacher to associate the exam with an existing course.  
4. The system should save the created exam.  
5. The system should display a confirmation message after successful creation of the exam.  
  
  
Analysis:  
The User Story US2001 describes the functionality to create exams in the system. Teachers have the ability to create and edit exams by providing information such as the exam title, start date, end date, and associating the exam with a specific course. This information is essential for scheduling and organizing exams.  
  
The implementation of this User Story would involve creating a user interface (UI) that allows teachers to enter exam details such as title, date, and time. Additionally, it would be necessary to integrate the functionality to associate the exam with an existing course, possibly by presenting a list of available courses for selection.  
  
Internally, the system would require a controller layer, such as the 'CreateExamController', which would be responsible for receiving the data provided by the UI and executing the necessary business logic. This would include creating an 'Exam' object based on the provided data and calling the appropriate repository, such as the 'ExamRepository', to save the exam.  
  
The US2001 User Story enables teachers to efficiently create exams, properly schedule them, and associate them with the appropriate courses. This functionality is crucial for effective exam management and contributes to the organization and efficiency of the assessment process in educational institutions.  
  
This analysis provides an overview of the US2001 User Story and can serve as a basis for planning and implementing this functionality within the system.

### [File Ends] sem4pi-22-23-61-master\docs\US2001\Analysis.txt

### [File Begins] sem4pi-22-23-61-master\docs\US2001\US2001-SD.puml

@startuml  
  
actor Teacher #DeepSkyBlue  
  
Teacher -> CreateExamUI : createExam()  
CreateExamUI --> Teacher : Prompt for exam details  
Teacher -> CreateExamUI : provide exam details  
CreateExamUI -> Course : getCourses()  
Course --> CreateExamUI : Return list of courses  
CreateExamUI -> CreateExamController : createExam()  
CreateExamController -> Exam : Exam(title, date, openTime, closeTime, course)  
CreateExamController -> ExamRepository : save(exam)  
ExamRepository --> CreateExamController : Exam saved  
CreateExamController --> CreateExamUI : Exam created successfully  
CreateExamUI --> Teacher : Display exam created successfully  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US2001\US2001-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US2002\US2002-ANALYSIS.md

\*\*Análise User Story 2002 - Listar futuros exames do estudante registado\*\*  
   
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
 \* O estudante tem de estar inscrito no Course que o exame está associado  
 \* A data do exame tem de ser posterior à data atual  
   
  
\* \*\*TESTES UNITÁRIOS\*\*  
  
 \* EnsurePastExamDoesntShow

### [File Ends] sem4pi-22-23-61-master\docs\US2002\US2002-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US2002\US2002-SD.puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor Student as Actor  
participant ListFutureExamsUI as UI  
participant ListFutureExamsController as Controller  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant ListFutureExamsService as Service  
participant Student as Student  
participant ExamRepository as Repo  
  
-> UI: Show future exams  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> Controller: create(authz, examRepository)  
 activate Controller  
 Controller -> Service : create(authz, examRepository)  
 activate Service  
 deactivate Service  
 deactivate Controller  
 UI -> Student : getMecanographicNumber()  
 activate Student  
 deactivate Student  
 UI -> Controller: listFutureExams(mecanographicNumber)  
  
 activate Controller  
 Controller -> Service: futureExams(mecanographicNumber)  
  
 activate Service  
 Service -> Repo: findFutureExamsBy\nMecanographicNumber(mecanographicNumber)  
 activate Repo  
 Repo --> Service : exams  
 deactivate Repo  
 Service --> Controller : exams  
 deactivate Service  
  
 Controller --> UI: exams  
 deactivate Controller  
  
 UI-->Actor: lists exams  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US2002\US2002-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US2003\US2003-ANALYSIS.md

\*\*Análise User Story 2003 - Listar exames de um curso\*\*  
  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
 \* O professor seleciona o curso desejado  
 \* São apenas apresentados os exames associados a esse curso  
  
  
\* \*\*TESTES UNITÁRIOS\*\*  
  
 \* EnsureOnlyCourseExamsShow

### [File Ends] sem4pi-22-23-61-master\docs\US2003\US2003-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US2003\US2003-SD.puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor Teacher as Actor  
participant ListCourseExamsUI as UI  
participant ListCourseExamsController as Controller  
participant ListCourseExamsService as Service  
participant CourseRepository as CourseRepo  
participant ExamRepository as Repo  
  
-> UI: Show course exams  
activate UI  
 UI -> Controller: create(authz, examRepository, courseRepository)  
 activate Controller  
 Controller -> Service : create(authz, examRepository, courseRepository)  
 activate Service  
 deactivate Service  
 deactivate Controller  
 UI -> Controller : listCourses()  
 activate Controller  
 Controller -> Service : courses()  
 activate Service  
 Service -> CourseRepo : findAll()  
 activate CourseRepo  
 CourseRepo --> Service : courses  
 deactivate CourseRepo  
 Service --> Controller : courses  
 deactivate Service  
 Controller --> UI : courses  
 deactivate Controller  
  
 UI --> Actor : show courses  
  
 Actor -> UI : selects course  
  
 UI -> Controller : listCourseExams(course)  
 activate Controller  
 Controller -> Service : courseExams(course)  
 activate Service  
 Service -> Repo : findByCourse(course)  
 activate Repo  
 Repo --> Service : exams  
 deactivate Repo  
 Service --> Controller : exams  
 deactivate Service  
 Controller --> UI : exams  
 deactivate Controller  
  
 UI-->Actor: lists exams  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US2003\US2003-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US3004\US3004-ANALYSIS.md

Análise User Story 3004 - Como usuário quero partilhar uma board  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
  
Só o owner pode partilhar a board.  
Ao compartilhar uma board, o usuário deve fornecer o nome dos destinatários.  
O compartilhamento de uma board permite que os destinatários visualizem e editem o conteúdo da board.  
  
  
\* \*\*Perguntas ao cliente\*\*  
  
  
  
\* \*\*TESTES UNITÁRIOS\*\*

### [File Ends] sem4pi-22-23-61-master\docs\US3004\US3004-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US3004\US3004-CD.puml

@startuml  
  
  
class User {  
 +username: String  
 +findByUsername(username: String): User  
}  
  
class SharedBoard {  
 +owner: User  
 +usersWithAccess: Set<User>  
 +accessType: AccessType  
}  
  
enum AccessType {  
 READ  
 WRITE  
}  
  
class ShareBoardUI {  
 +shareBoard(ownerUsername: String, userToShareUsername: String, accessType: String)  
 +showSuccessMessage(message: String)  
 +showErrorMessage(message: String)  
}  
  
class ShareBoardController {  
 -shareBoard(ownerUsername: String, userToShareUsername: String, accessType: String)  
}  
  
class ShareBoardService {  
 -shareBoard(ownerUsername: String, userToShareUsername: String, accessType: String)  
}  
  
class SharedBoardRepository {  
 +findSharedBoardsByOwner(owner: User): Set<SharedBoard>  
}  
  
class SystemUserRepository {  
 +findByUsername(username: String): User  
}  
  
User "1" \*-- "1" SystemUserRepository: Uses  
ShareBoardUI "1" \*-- "1" ShareBoardController: Uses  
ShareBoardController "1" \*-- "1" ShareBoardService: Uses  
ShareBoardService "1" \*-- "1" SharedBoardRepository: Uses  
ShareBoardService "1" \*-- "1" SystemUserRepository: Uses  
ShareBoardController "1" \*-- "1" ShareBoardUI: Sends  
  
User "1" \*-- "\*" SharedBoard: Owns  
SharedBoard "1" \*-- "\*" User: Has access  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3004\US3004-CD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US3004\US3004-SD.puml

@startuml  
'skinparam settings  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor User as Actor  
participant ShareBoardUI as UI  
participant ShareBoardController as Controller  
participant ShareBoardService as Service  
participant SharedBoardRepository as Repo  
participant SystemUserRepository as UserRepo  
  
activate Actor  
Actor -> UI: wants to share a board  
activate UI  
Actor -> UI: Provide requested data (ownerUsername, userToShareUsername, accessType)  
  
UI -> Controller: shareBoard(ownerUsername, userToShareUsername, SharedBoard.AccessType.valueOf(accessType.toUpperCase()))  
activate Controller  
Controller -> Service: shareBoard(ownerUsername, userToShareUsername, accessType)  
activate Service  
Service -> UserRepo : findByUsername(ownerUsername)  
activate UserRepo  
deactivate UserRepo  
Service -> UserRepo : findByUsername(userToShareUsername)  
activate UserRepo  
deactivate UserRepo  
Service -> Repo: findSharedBoardsByOwner(owner)  
activate Repo  
deactivate Repo  
alt SharedBoard and users are found  
 Service -> Controller : shareBoard  
 Controller -> UI : Show success message  
 UI -> Actor : System types : "Shared board with user: " + userToShareUsername + " with " + accessType + " access."  
else Board not found or User not found  
 Controller -> UI : Show error message  
 UI -> Actor : System types : "Failed to share board: " + e.getMessage()  
end  
deactivate Service  
deactivate Controller  
deactivate UI  
deactivate Actor  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3004\US3004-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US3006\US3006(ONLY\_READ\_PERMISSION)-SD.puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant CreateBoardPostItUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant CreateBoardPostItController as Controller  
participant CreateBoardPostItService as Service  
participant SharedBoardRepository as Repo  
participant "theSharedBoard:SharedBoard" as SharedBoard  
participant PostIt as PostIt  
participant CreateBoardPostItThread as Thread  
participant Board as Board  
  
-> UI: Create Board post-it  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> Controller: create(sharedBoardRepository)  
 activate Controller  
 Controller -> Service : create(sharedBoardRepository)  
 activate Service  
 deactivate Service  
 deactivate Controller  
 UI -> Controller : findBoardsBySystemUser(myUser)  
 activate Controller  
 Controller -> Service : findBoardsBySystemUser(myUser)  
 activate Service  
 Service -> Repo : findBoardsBySystemUser(myUser)  
 activate Repo  
 deactivate Repo  
 deactivate Service  
 deactivate Controller  
 UI --> Actor : ask user shared \nboard selection  
 Actor --> UI : select shared board  
  
 UI -> Controller : getBoardAccessType(theSharedBoard, myUser)  
 activate Controller  
 Controller -> Service : getBoardAccessType(sharedBoard, myUser)  
 activate Service  
 Service -> SharedBoard : getAccessType(systemUser)  
 activate SharedBoard  
 deactivate SharedBoard  
 deactivate Service  
 deactivate Controller  
  
 UI --> Actor : informs the user doesn't \n have write permissions  
  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3006\US3006(ONLY\_READ\_PERMISSION)-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US3006\US3006-ANALYSIS.md

\*\*Análise User Story 3006 - Criar um post-it na board\*\*  
  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
 \* O conteúdo do post-it pode ser texto ou uma imagem  
 \* Apenas utilizadores com permissão de write podem fazer alterações na board  
 \* Apenas um utilizador de cada vez pode fazer alterações na board  
  
  
  
\* \*\*Perguntas ao cliente:\*\*  
  
 \* \*\*Pergunta 1\*\*: As a client, do you want us to persist post-its in the database or they can be available only while the server is running (deleted if the server stops)?  
 \* \*\*Resposta\*\*: For me the information regarding the shared boards may be available only during the shared board server execution.  
  
 \* \*\*Pergunta 2\*\*: Uma célula pode ter mais do que um post it?  
 \* \*\*Resposta\*\*: Neste momento (no âmbito deste projeto) isso não será necessário. A ser possível (uma célula com mais do que 1 post-it) isso iria dificultar algumas funcionalidades, como a que permite mudar um post-it.  
  
 \* \*\*Pergunta 3\*\*: Quando um User cria um post-it deve passar um link da imagem por exemplo:  
 "https://www.isep.ipp.pt/img/logo\_20230106.png"  
 Ou devemos anexar uma imagem que está no nosso computador?  
 \* \*\*Resposta\*\*: Para o cliente é um pouco indiferente o mecanismo que usam para fazer o "post" de imagens (assim como o(s) formato(s) suportado(s)).  
  
  
  
\* \*\*TESTES UNITÁRIOS\*\*  
  
 \* EnsureUserWithReadPermissionCantWrite  
 \* EnsurePostItIsNotAddedIfTheDesiredPositionIsOccupied  
 \* EnsurePostItIsAddedIfTheDesiredPositionIsFree

### [File Ends] sem4pi-22-23-61-master\docs\US3006\US3006-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US3006\US3006-SD(WRITE\_PERMISSION).puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant CreateBoardPostItUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant CreateBoardPostItController as Controller  
participant CreateBoardPostItService as Service  
participant SharedBoardRepository as Repo  
participant "theSharedBoard:SharedBoard" as SharedBoard  
participant PostIt as PostIt  
participant CreateBoardPostItThread as Thread  
participant Board as Board  
  
-> UI: Create Board post-it  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> Controller: create(sharedBoardRepository)  
 activate Controller  
 Controller -> Service : create(sharedBoardRepository)  
 activate Service  
 deactivate Service  
 deactivate Controller  
 UI -> Controller : findBoardsBySystemUser(myUser)  
 activate Controller  
 Controller -> Service : findBoardsBySystemUser(myUser)  
 activate Service  
 Service -> Repo : findBoardsBySystemUser(myUser)  
 activate Repo  
 deactivate Repo  
 deactivate Service  
 deactivate Controller  
 UI --> Actor : ask user shared \nboard selection  
 Actor --> UI : select shared board  
  
 UI -> Controller : getBoardAccessType(theSharedBoard, myUser)  
 activate Controller  
 Controller -> Service : getBoardAccessType(sharedBoard, myUser)  
 activate Service  
 Service -> SharedBoard : getAccessType(systemUser)  
 activate SharedBoard  
 deactivate SharedBoard  
 deactivate Service  
 deactivate Controller  
 UI --> Actor : ask content of\nthe post-it  
 Actor --> UI : content  
  
 UI -> Controller : createPostIt(content, myUser)  
 activate Controller  
 Controller -> Service : createPostIt(content, myUser)  
 activate Service  
 Service -> PostIt : postIt = create(content, myUser)  
 deactivate Service  
 deactivate Controller  
  
 UI --> Actor : ask user row and\ncolumn to add the post-it  
 Actor --> UI : row and column  
  
 UI -> Controller : addPostIt(theSharedBoard, row, column)  
 activate Controller  
 Controller -> Service : addPostIt(theSharedBoard, row, column)  
 activate Service  
 Service -> Thread : create(theSharedBoard.getBoard(), row, column, postIt)  
 activate Thread  
 deactivate Thread  
 Service -> Thread : start()  
 activate Thread  
 Thread -> Board : insertCell(row, column, postIt)  
 deactivate Thread  
 deactivate Service  
 deactivate Controller  
  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3006\US3006-SD(WRITE\_PERMISSION).puml

### [File Begins] sem4pi-22-23-61-master\docs\US3007\US3007-ANALYSIS.md

\*\*Análise User Story 3007 - Alterar um post-it na board\*\*  
  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
 \* Apenas utilizadores com permissão de write podem fazer alterações na board  
 \* O utilizador pode apenas alterar os post-it que criou na board  
 \* O utilizador pode alterar o conteúdo do post-it ou mover para uma célula livre  
 \* Apenas um utilizador de cada vez pode fazer alterações na board  
  
  
\* \*\*TESTES UNITÁRIOS\*\*  
  
 \* EnsureUserWithReadPermissionCantWrite  
 \* EnsurePostItIsNotUpdatedIfTheOriginalPositionIsNull  
 \* EnsurePostItIsNotUpdatedIfUserIsNotTheAuthorOfTheOriginalPostIt  
 \* EnsurePostItIsUpdatedIfAllConditionsAreMet  
 \* EnsurePostItIsNotMovedIfTheOriginalPositionIsNull  
 \* EnsurePostItIsNotMovedIfTheNewPositionIsOccupied  
 \* EnsurePostItIsNotMovedIfUserIsNotTheAuthorOfTheOriginalPostIt  
 \* EnsurePostItIsMovedIfAllConditionsAreMet

### [File Ends] sem4pi-22-23-61-master\docs\US3007\US3007-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US3007\US3007-SD(MOVE\_POST-IT).puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant UpdateBoardPostItUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant UpdateBoardPostItController as Controller  
participant UpdateBoardPostItService as Service  
participant SharedBoardRepository as Repo  
participant "theSharedBoard:SharedBoard" as SharedBoard  
participant PostIt as PostIt  
participant MoveBoardPostItThread as Thread  
participant Board as Board  
  
-> UI: Move a post-it in a board  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> Controller: create(sharedBoardRepository)  
 activate Controller  
 Controller -> Service : create(sharedBoardRepository)  
 activate Service  
 deactivate Service  
 deactivate Controller  
 UI -> Controller : findBoardsBySystemUser(myUser)  
 activate Controller  
 Controller -> Service : findBoardsBySystemUser(myUser)  
 activate Service  
 Service -> Repo : findBoardsBySystemUser(myUser)  
 activate Repo  
 deactivate Repo  
 deactivate Service  
 deactivate Controller  
 UI --> Actor : ask user shared \nboard selection  
 Actor --> UI : select shared board  
  
 UI -> Controller : getBoardAccessType(theSharedBoard, myUser)  
 activate Controller  
 Controller -> Service : getBoardAccessType(sharedBoard, myUser)  
 activate Service  
 Service -> SharedBoard : getAccessType(systemUser)  
 activate SharedBoard  
 deactivate SharedBoard  
 deactivate Service  
 deactivate Controller  
 UI --> Actor : ask row, column, new row and\nnew column to move the post-it  
 Actor --> UI : row, column, new row, new column  
  
 UI -> Controller : movePostIt(theSharedBoard, row, column, newRow, newColumn, myUser)  
 activate Controller  
 Controller -> Service : movePostIt(theSharedBoard, row, column, newRow, newColumn, myUser)  
 activate Service  
 Service -> Thread : create(theSharedBoard.getBoard(), row, column, newRow, newColumn, myUser)  
 activate Thread  
 deactivate Thread  
 Service -> Thread : start()  
 activate Thread  
 Thread -> Board : moveCell(row, column, newRow, newColumn, myUser)  
 activate Board  
 deactivate Board  
 deactivate Thread  
 deactivate Service  
 deactivate Controller  
  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3007\US3007-SD(MOVE\_POST-IT).puml

### [File Begins] sem4pi-22-23-61-master\docs\US3007\US3007-SD(UPDATE\_CREATED\_POST-IT).puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant UpdateBoardPostItUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant UpdateBoardPostItController as Controller  
participant UpdateBoardPostItService as Service  
participant SharedBoardRepository as Repo  
participant "theSharedBoard:SharedBoard" as SharedBoard  
participant PostIt as PostIt  
participant UpdateBoardPostItThread as Thread  
participant Board as Board  
  
-> UI: Update a post-it in a board  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> Controller: create(sharedBoardRepository)  
 activate Controller  
 Controller -> Service : create(sharedBoardRepository)  
 activate Service  
 deactivate Service  
 deactivate Controller  
 UI -> Controller : findBoardsBySystemUser(myUser)  
 activate Controller  
 Controller -> Service : findBoardsBySystemUser(myUser)  
 activate Service  
 Service -> Repo : findBoardsBySystemUser(myUser)  
 activate Repo  
 deactivate Repo  
 deactivate Service  
 deactivate Controller  
 UI --> Actor : ask user shared \nboard selection  
 Actor --> UI : select shared board  
  
 UI -> Controller : getBoardAccessType(theSharedBoard, myUser)  
 activate Controller  
 Controller -> Service : getBoardAccessType(sharedBoard, myUser)  
 activate Service  
 Service -> SharedBoard : getAccessType(systemUser)  
 activate SharedBoard  
 deactivate SharedBoard  
 deactivate Service  
 deactivate Controller  
 UI --> Actor : ask row, column, content\nof the post-it  
 Actor --> UI : row, column, content  
  
 UI -> Controller : createPostIt(content, myUser)  
 activate Controller  
 Controller -> Service : createPostIt(content, myUser)  
 activate Service  
 Service -> PostIt : postIt = create(content, myUser)  
 activate PostIt  
 deactivate PostIt  
 deactivate Service  
 deactivate Controller  
  
 UI -> Controller : updatePostIt(theSharedBoard, row, column)  
 activate Controller  
 Controller -> Service : updatePostIt(theSharedBoard, row, column)  
 activate Service  
 Service -> Thread : create(theSharedBoard.getBoard(), row, column, postIt)  
 activate Thread  
 deactivate Thread  
 Service -> Thread : start()  
 activate Thread  
 Thread -> Board : updateCell(row, column, postIt)  
 activate Board  
 deactivate Board  
 deactivate Thread  
 deactivate Service  
 deactivate Controller  
  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3007\US3007-SD(UPDATE\_CREATED\_POST-IT).puml

### [File Begins] sem4pi-22-23-61-master\docs\US3010\US3010-ANALYSIS.md

REGRAS DE NEGÓCIO  
  
Only the Admin of the board can archive it.  
When a board is archived, all shared users' access types should be changed to "READ," except for the owner who retains "WRITE" access.  
Archiving a board sets the isArchived flag to true.  
  
  
  
Perguntas ao cliente  
  
  
  
  
  
TESTES UNITÁRIOS

### [File Ends] sem4pi-22-23-61-master\docs\US3010\US3010-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US3010\US3010-CD.puml

@startuml  
skinparam handwritten true  
class SystemUser {  
}  
  
class SharedBoardArchiveUI {  
 +archiveBoard(title: string, currentUser: User)  
 +displayMessage(message: string)  
}  
  
class AuthzRegistry {  
 +authorizationService(): AuthorizationService  
}  
  
class AuthorizationService {  
 +session(): UserSession  
}  
  
class UserSession {  
 +authenticatedUser(): User  
}  
  
class SharedBoardArchiveController {  
 +archiveBoard(title: string, currentUser: User)  
}  
  
class SharedBoardArchiveService {  
 +archiveBoard(title: string, currentUser: User)  
}  
  
class SharedBoardRepository {  
 +findByBoardId(title: string): SharedBoard  
 +save(board: SharedBoard)  
}  
  
class SharedBoard {  
 -isArchived: boolean  
 +setIsArchived(isArchived: boolean)  
 +updateAccessTypeForSharedUsers()  
}  
  
class User {  
}  
  
SystemUser --> SharedBoardArchiveUI  
SharedBoardArchiveUI --> AuthzRegistry  
SharedBoardArchiveUI --> AuthorizationService  
SharedBoardArchiveUI --> UserSession  
SharedBoardArchiveUI --> SharedBoardArchiveController  
SharedBoardArchiveUI --> SystemUser  
  
AuthzRegistry --> AuthorizationService  
  
AuthorizationService --> UserSession  
  
SharedBoardArchiveController --> SharedBoardArchiveService  
  
SharedBoardArchiveService --> SharedBoardRepository  
  
SharedBoardArchiveController --> SharedBoard  
SharedBoardArchiveService --> SharedBoard  
SharedBoard --> SharedBoardRepository  
  
UserSession --> User  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3010\US3010-CD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US3010\US3010-SD.puml

@startuml  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant SharedBoardArchiveUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant SharedBoardArchiveController as Controller  
participant SharedBoardArchiveService as Service  
participant SharedBoardRepository as Repo  
participant "theSharedBoard:SharedBoard" as SharedBoard  
  
activate Actor  
Actor -> UI : wants to archive a board  
activate UI  
UI -> AuthzRes : authz=authorizationService()  
activate AuthzRes  
deactivate AuthzRes  
  
UI -> AuthzSvc : s=session()  
activate AuthzSvc  
deactivate AuthzSvc  
  
UI -> UserS : myUser=authenticatedUser()  
activate UserS  
deactivate UserS  
  
UI -> Controller: archiveBoard(title, currentUser)  
activate Controller  
Controller -> Service : archiveBoard(title, currentUser)  
activate Service  
Service -> Repo : findByBoardId(title)  
activate Repo  
deactivate Repo  
Service -> SharedBoard : archiveBoard()  
activate SharedBoard  
SharedBoard -> SharedBoard : set isArchived=true  
SharedBoard -> SharedBoard : Update access type for shared users  
SharedBoard -> Repo : save(board)  
activate Repo  
deactivate Repo  
deactivate SharedBoard  
deactivate Service  
deactivate Controller  
UI -> Actor : Display Message "Board has been Archived"  
deactivate Actor  
deactivate UI  
  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US3010\US3010-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US4001\US4001-ANALYSIS.md

\*\*Análise User Story 4001 - Agendar um meeting\*\*  
  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
 \* Qualquer utilizador pode criar um meeting  
 \* Qualquer utilizador pode ser convidado para o meeting  
 \* Se algum utilizador não estiver disponível, o meeting não deve ser criado  
  
  
  
\* \*\*Perguntas ao cliente:\*\*  
  
 \* \*\*Pergunta 1\*\*: Can any user of the system invite any other user? For example, can a student invite another student who is in a different course, or can a manager can create a meeting with any group of teachers.  
 \* \*\*Resposta\*\*: When in the document specification the term "User" is used it usually means "any user" of the system. Therefore, any user of the system can schedule a meeting and be a participant in a meeting.  
  
 \* \*\*Pergunta 2\*\*: Can a meeting and class overlap? If so, should the system notify that there is an overlap and for which user(s)?  
 \* \*\*Resposta\*\*: In the case of Meetings they should not be created if the participants are not available (i.e., they may have classes or other meetings at the same time).  
  
  
  
\* \*\*TESTES UNITÁRIOS\*\*  
  
 \* CancelMeeting\_MeetingStatusChangedToCanceled

### [File Ends] sem4pi-22-23-61-master\docs\US4001\US4001-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US4001\US4001-SD.puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant ScheduleMeetingUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant ListUsersController as UsersController  
participant ScheduleMeetingController as Controller  
participant UserManagementService as Usersvc  
participant MeetingRepository as Repo  
participant InviteRepository as InviteRepo  
  
-> UI: Schedule meeting  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> UsersController: create()  
 activate UsersController  
 deactivate UsersController  
 UI -> Controller: create(authz, meetingRepository, InviteRepository)  
 activate Controller  
 deactivate Controller  
 UI -> Controller: saveMeeting(meetingDuration, meetingDate, meetingTime)  
 activate Controller  
 Controller -> Repo : save(meetingDuration, meetingDate, meetingTime)  
 activate Repo  
 deactivate Repo  
 deactivate Controller  
 UI -> UsersController: allUsers()  
 activate UsersController  
 UsersController -> Usersvc : allUsers()  
 activate Usersvc  
 Usersvc --> UsersController : system users  
 deactivate Usersvc  
 UsersController --> UI : system users  
 deactivate UsersController  
 loop for each selected user  
 UI --> Actor : shows system users  
  
 Actor -> UI : selects User  
  
 UI -> Controller : saveInvite(myUser, systemUser, meeting)  
 activate Controller  
 Controller -> InviteRepo : save(myUser, systemUser, meeting)  
 activate InviteRepo  
 deactivate InviteRepo  
 deactivate Controller  
  
 UI --> Actor : invite sent  
 end  
  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US4001\US4001-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US4002\US4002-ANALYSIS.md

Análise User Story 4002 - Como usuário, quero cancelar uma reunião  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
Apenas um administrador têm permissão para cancelar uma reunião.  
A reunião cancelada vai ter um status, SCHEDULED, COMPLETED, CANCELED, quando cancelada vai mudar o seu status para CANCELED  
  
\* \*\*Perguntas ao cliente\*\*  
  
Há alguma ação adicional que precisa ser executada após o cancelamento de uma reunião?  
  
\* \*\*TESTES UNITÁRIOS\*\*  
   
\* Verificar se o criador da reunião ou um administrador pode cancelar a reunião com sucesso.  
\* Verificar se a reunião cancelada altera seu status para CANCELED.  
\* Verificar se uma reunião que já está concluída (status COMPLETED) ou já cancelada (status CANCELED) pode ser cancelada novamente - isso deve falhar.

### [File Ends] sem4pi-22-23-61-master\docs\US4002\US4002-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US4002\US4002-CD.puml

@startuml  
  
class CancelMeetingUI{}  
  
  
class CancelMeetingController{  
findById(meetingId)  
updateStatus(meeting, "Canceled")  
validateMeetingOwnership(meeting, myUser)  
}  
  
class AuthzRegistry{  
authorizationService()  
}  
  
class AuthorizationService{  
session()  
}  
class UserSession{  
+authenticatedUser()  
}  
  
class Meeting {  
 +invite  
 +inviteState  
 +meetingDate  
 +meetingDuration  
 +meetingStatus  
 +meetingTime  
 +cancelMeeting(): void  
}  
  
enum MeetingStatus(){  
SCHEDULED  
 COMPLETED  
 CANCELED  
}  
  
class SystemUser {  
 +username  
 +username  
 +email  
 +password  
 +name  
}  
  
class MeetingRepository {  
 +save(meeting: Meeting): void  
 +update(meeting: Meeting): void  
 +delete(meeting: Meeting): void  
 +findById(meetingId: int): Meeting  
}  
  
  
Meeting "1" --> "\*" SystemUser : participants  
Meeting "1" -- "1" SystemUser : createdBy  
Meeting --> MeetingRepository  
CancelMeetingUI --> CancelMeetingController  
CancelMeetingUI --> AuthzRegistry  
CancelMeetingUI --> UserSession  
CancelMeetingUI --> AuthorizationService  
CancelMeetingController --> Meeting  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US4002\US4002-CD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US4002\US4002-DM.puml

@startuml  
  
class Meeting{  
meetingTime  
title  
meetingDuration  
meetingDate  
List<SystemUser>  
MeetingStatus  
}  
  
  
class SystemUser{  
version  
Username  
Password  
Name  
EmailAddress  
}  
class Administrator{}  
  
  
Administrator "1" --> "\*" Meeting : Cancels  
Administrator "1" --> "1" SystemUser : is a  
  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US4002\US4002-DM.puml

### [File Begins] sem4pi-22-23-61-master\docs\US4002\US4002-SD.puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant CancelMeetingUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant MeetingRepository as Repo  
participant CancelMeetingController as Controller  
  
-> UI: Cancelar reunião  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> Controller: create(authz, meetingRepository)  
 activate Controller  
 deactivate Controller  
 UI -> Controller: cancelMeeting(meetingId)  
 activate Controller  
 Controller -> Repo : findById(meetingId)  
 activate Repo  
 Repo --> Controller : meeting  
 deactivate Repo  
 Controller -> Controller : validateMeetingOwnership(meeting, myUser)  
 Controller -> Repo : updateStatus(meeting, "Canceled")  
 activate Repo  
 deactivate Repo  
 deactivate Controller  
  
 UI --> Actor : Estado da reunião alterado para "Cancelada"  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US4002\US4002-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\US4003\US4003-ANALYSIS.md

\*\*Análise User Story 4003 - Aceitar/Rejeitar convites para um meeting\*\*  
  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
 \* O utilizador tem a opção de aceitar ou rejeitar o convite  
 \* O convite tem 3 estados: pendente, aceite, rejeitado  
  
  
  
\* \*\*Perguntas ao cliente:\*\*  
  
 \* \*\*Pergunta 1\*\*: When mentioning "accept or reject meeting", I though to myself about an invite being sent to the user and 2 options for the response:  
 \* -The sent invite has already as a response "Rejected", so that it can either be changed to "Accepted" or stays as it is, seeing that if the user doesn't accept it, he will be rejecting it.  
 \* -The sent invite has a response being "No answer" and, at a certain time near the begining of the meeting, the answer would change to "Rejected". The answer can be changed before it at any time to "Accepted" or "Rejected"  
   
 \* \*\*Resposta\*\*: In FRM04, the status of someone that did not answer should be "no answer" or "unknown".  
  
  
  
\* \*\*TESTES UNITÁRIOS\*\*  
  
 \* EnsureInviteStateChangesToAcceptedWhenUserAccepts  
 \* EnsureMeetingContainsUserWhenInviteIsAccepted  
 \* EnsureInviteStateChangesToRefusedWhenUserRefuses  
 \* EnsureMeetingDoesNotContainsUserWhenInviteIsRefused

### [File Ends] sem4pi-22-23-61-master\docs\US4003\US4003-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US4003\US4003-SD(ACCEPT\_INVITE).puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant AcceptRefuseInvitesUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant AcceptRefuseInvitesController as InvitesController  
participant AcceptRefuseInvitesService as Invitesvc  
participant InviteRepository as Repo  
participant "theInvite:Invite" as Invite  
participant Meeting as Meeting  
  
  
-> UI: Accept/Refuse meeting invites  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> InvitesController: create(InviteRepository)  
 activate InvitesController  
 deactivate InvitesController  
  
 loop for each selected invite  
 UI -> InvitesController: invitesReceived(myUser.username())  
 activate InvitesController  
 InvitesController -> Invitesvc : invitesReceived(username)  
 activate Invitesvc  
 Invitesvc -> Repo : findInvitesByReceiverUsername(username)  
 activate Repo  
 Repo --> Invitesvc  
 deactivate Repo  
 Invitesvc --> InvitesController  
 deactivate Invitesvc  
 InvitesController --> UI  
 deactivate InvitesController  
 UI --> Actor : ask user invite \nselection  
 Actor -> UI : select and \naccept invite  
  
 UI -> InvitesController : acceptInvite(theInvite, myUser)  
 activate InvitesController  
 InvitesController -> Invitesvc : acceptInvite(theInvite, myUser)  
 activate Invitesvc  
 Invitesvc -> Invite : addMeetingParticipants(myUser)  
 activate Invite  
 Invite -> Meeting : addParticipants(myUser)  
 activate Meeting  
 deactivate Meeting  
 deactivate Invite  
 Invitesvc -> Invite : setState(InviteState.ACCEPTED)  
 activate Invite  
 deactivate Invite  
 deactivate Invitesvc  
 deactivate InvitesController  
  
  
  
  
  
  
deactivate UI  
end  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US4003\US4003-SD(ACCEPT\_INVITE).puml

### [File Begins] sem4pi-22-23-61-master\docs\US4003\US4003-SD(REFUSE\_INVITE).puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant AcceptRefuseInvitesUI as UI  
participant AuthzRegistry as AuthzRes  
participant AuthorizationService as AuthzSvc  
participant UserSession as UserS  
participant AcceptRefuseInvitesController as InvitesController  
participant AcceptRefuseInvitesService as Invitesvc  
participant InviteRepository as Repo  
participant "theInvite:Invite" as Invite  
  
  
-> UI: Accept/Refuse meeting invites  
activate UI  
 UI -> AuthzRes : authz=authorizationService()  
 activate AuthzRes  
 deactivate AuthzRes  
 UI -> AuthzSvc : s=session()  
 activate AuthzSvc  
 deactivate AuthzSvc  
 UI -> UserS : myUser=authenticatedUser()  
 activate UserS  
 deactivate UserS  
  
 UI -> InvitesController: create(InviteRepository)  
 activate InvitesController  
 deactivate InvitesController  
  
 loop for each selected invite  
 UI -> InvitesController: invitesReceived(myUser.username())  
 activate InvitesController  
 InvitesController -> Invitesvc : invitesReceived(username)  
 activate Invitesvc  
 Invitesvc -> Repo : findInvitesByReceiverUsername(username)  
 activate Repo  
 Repo --> Invitesvc  
 deactivate Repo  
 Invitesvc --> InvitesController  
 deactivate Invitesvc  
 InvitesController --> UI  
 deactivate InvitesController  
 UI --> Actor : ask user invite \nselection  
 Actor -> UI : select and \nreject invite  
  
 UI -> InvitesController : refuseInvite(theInvite)  
 activate InvitesController  
 InvitesController -> Invitesvc : refuseInvite(theInvite)  
 activate Invitesvc  
 Invitesvc -> Invite : setState(InviteState.REJECTED)  
 activate Invite  
 deactivate Invite  
 deactivate Invitesvc  
 deactivate InvitesController  
  
deactivate UI  
end  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US4003\US4003-SD(REFUSE\_INVITE).puml

### [File Begins] sem4pi-22-23-61-master\docs\US4004\US4004-ANALYSIS.md

\*\*Análise User Story 4004 - Como usuário, visualizar uma lista de participantes na reunião e seu status\*\*  
  
  
\* \*\*REGRAS DE NEGÓCIO\*\*  
  
 \* O utilizador tem uma opção para visualizar a lista  
 \* Na lista existem 3 estados: aceite, rejeitado  
  
  
  
\* \*\*Perguntas ao cliente:\*\*  
  
 \* \*\*Pergunta 1\*\*: Isso significa que só devem ser apresentados os participantes que possuam status "accept" or "reject" (portanto não seriam mostrados os participantes com status "pending" ou "owner" por exemplo), ou serve apenas como exemplo para se entender o que refere quando se fala no status?  
  
 \* \*\*Resposta\*\*: A ideia é que apareçam todos os participantes e o seu estado. Suponho que quem ainda não respondeu não terá nada no estado ou "pending" ou "unknown".  
  
  
  
\* \*\*TESTES UNITÁRIOS\*\*

### [File Ends] sem4pi-22-23-61-master\docs\US4004\US4004-ANALYSIS.md

### [File Begins] sem4pi-22-23-61-master\docs\US4004\US4004-SD.puml

@startuml  
'http://plantuml.com/skinparam.html  
skinparam handwritten true  
skinparam monochrome true  
skinparam packageStyle rect  
skinparam defaultFontName FG Virgil  
skinparam shadowing false  
  
actor SystemUser as Actor  
participant ListParticipantUI as UI  
participant ListParticipantController as Controller  
participant ListParticipantService as Service  
participant MeetingRepository as MeetingRepo  
participant InviteRepository as InviteRepo  
participant Meeting as Meeting  
participant Invite as Invite  
participant SystemUser as User  
  
-> UI: View participants of a meeting  
activate UI  
 UI -> Controller: getMeetingById(meetingTitle)  
 activate Controller  
 Controller -> Service: findByMeetingById(meetingTitle)  
 activate Service  
 Service -> MeetingRepo: findByMeetingById(meetingTitle)  
 activate MeetingRepo  
 MeetingRepo --> Service  
 deactivate MeetingRepo  
 Service --> Controller  
 deactivate Service  
 Controller -> UI: meeting  
 deactivate Controller  
 alt meeting is null  
 UI -> Actor: display "No meeting found with the given ID."  
 else  
 UI -> Controller: findInvitesByMeeting(meeting)  
 activate Controller  
 Controller -> Service: findInvitesByMeeting(meeting)  
 activate Service  
 Service -> InviteRepo: findInvitesByMeeting(meeting)  
 activate InviteRepo  
 InviteRepo --> Service  
 deactivate InviteRepo  
 Service --> Controller  
 deactivate Service  
 Controller -> UI: participantsStatus  
 deactivate Controller  
 UI -> Actor: display participants list  
 end  
deactivate UI  
@enduml

### [File Ends] sem4pi-22-23-61-master\docs\US4004\US4004-SD.puml

### [File Begins] sem4pi-22-23-61-master\docs\readme.md

# Project Planning and Technical Documentation  
  
## 1. Group Members  
  
The members of the group:  
  
| Student Nr. | Name |  
|--------------|------------------------------|  
| \*\*[1210965](Group\_Elements/1210965/readme.md)\*\* | José Teixeira |  
| \*\*[1200614](Group\_Elements/1200614/readme.md)\*\* | João Miranda|  
| \*\*[1201718](Group\_Elements/1201718/readme.md)\*\* | João Cruz|  
| \*\*[1200874](Group\_Elements/1200874/readme.md)\*\* | Miguel Oliveira|  
| \*\*[1200801](Group\_Elements/1200801/readme.md)\*\* | Daniel Braga|  
  
## 2. Task Assignment  
  
The assignment of tasks (requirements/user stories/use cases) during the project.  
  
| Student Nr. | Sprint A | Sprint B | Sprint C |  
|------------|----------|----------|----------|  
| [1170000](Group\_Elements/1210965/readme.md)| [US G002](us\_g002(base)(apagar depois)/readme.md)| [US 1001](us\_1001(Base)(Apagar depois)/readme.md)| [US 3004](us\_3004(base)(apagar depois)/readme.md) |  
| | | | |  
| | | | |  
| | | | |  
| | | | |  
| | | | |  
| | | | |

### [File Ends] sem4pi-22-23-61-master\docs\readme.md

### [File Begins] sem4pi-22-23-61-master\libs\ModeloConceptual (1).drawio.html

<!--[if IE]><meta http-equiv="X-UA-Compatible" content="IE=5,IE=9" ><![endif]-->  
<!DOCTYPE html>  
<html>  
<head>  
<title>ModeloConceptual (1)</title>  
<meta charset="utf-8"/>  
</head>  
<body><div class="mxgraph" style="max-width:100%;border:1px solid transparent;" data-mxgraph="{&quot;highlight&quot;:&quot;#0000ff&quot;,&quot;nav&quot;:true,&quot;resize&quot;:true,&quot;toolbar&quot;:&quot;zoom layers tags lightbox&quot;,&quot;edit&quot;:&quot;\_blank&quot;,&quot;xml&quot;:&quot;&lt;mxfile host=\&quot;app.diagrams.net\&quot; modified=\&quot;2023-03-31T14:23:59.192Z\&quot; agent=\&quot;Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/111.0.0.0 Safari/537.36\&quot; etag=\&quot;ao-w5NrHMs82-NgX3OQf\&quot; version=\&quot;21.1.2\&quot; type=\&quot;device\&quot;&gt;\n &lt;diagram id=\&quot;C5RBs43oDa-KdzZeNtuy\&quot; name=\&quot;Modelo Conceitual\&quot;&gt;\n &lt;mxGraphModel dx=\&quot;3452\&quot; dy=\&quot;2637\&quot; grid=\&quot;1\&quot; gridSize=\&quot;10\&quot; guides=\&quot;1\&quot; tooltips=\&quot;1\&quot; connect=\&quot;1\&quot; arrows=\&quot;1\&quot; fold=\&quot;1\&quot; page=\&quot;1\&quot; pageScale=\&quot;1\&quot; pageWidth=\&quot;827\&quot; pageHeight=\&quot;1169\&quot; math=\&quot;0\&quot; shadow=\&quot;0\&quot;&gt;\n &lt;root&gt;\n &lt;mxCell id=\&quot;WIyWlLk6GJQsqaUBKTNV-0\&quot; /&gt;\n &lt;mxCell id=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-0\&quot; /&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-26\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Course Name\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1494.25\&quot; y=\&quot;995\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Shared Board\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2820\&quot; y=\&quot;-9.999999999999957\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-28\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Class\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;605\&quot; y=\&quot;-350\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-29\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;DashBoard\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-240\&quot; y=\&quot;360\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Course\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1086.5\&quot; y=\&quot;750\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Person\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2050\&quot; y=\&quot;100\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-115\&quot; value=\&quot;is checked by\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-29\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.3608\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;630\&quot; y=\&quot;-730\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;-180\&quot; y=\&quot;-810\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint x=\&quot;-1\&quot; as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Student\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1300\&quot; y=\&quot;-840\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Exam\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;170\&quot; y=\&quot;360\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-34\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Manager\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1410\&quot; y=\&quot;545\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-112\&quot; value=\&quot;create\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;exitX=0.167;exitY=0.999;exitDx=0;exitDy=0;exitPerimeter=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.1625\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1770\&quot; y=\&quot;-740\&quot; /&gt;\n &lt;mxPoint x=\&quot;1770\&quot; y=\&quot;190\&quot; /&gt;\n &lt;mxPoint x=\&quot;1070\&quot; y=\&quot;190\&quot; /&gt;\n &lt;mxPoint x=\&quot;1070\&quot; y=\&quot;380\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint x=\&quot;1740\&quot; y=\&quot;-740\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-4\&quot; value=\&quot;is checked by\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;exitX=0.75;exitY=0;exitDx=0;exitDy=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-29\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;-170\&quot; y=\&quot;-80\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1720\&quot; y=\&quot;-458.2050969050481\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;-150\&quot; y=\&quot;-70\&quot; /&gt;\n &lt;mxPoint x=\&quot;1040\&quot; y=\&quot;-70\&quot; /&gt;\n &lt;mxPoint x=\&quot;1040\&quot; y=\&quot;-200\&quot; /&gt;\n &lt;mxPoint x=\&quot;1760\&quot; y=\&quot;-200\&quot; /&gt;\n &lt;/Array&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-5\&quot; value=\&quot;check\&quot; style=\&quot;edgeLabel;html=1;align=center;verticalAlign=middle;resizable=0;points=[];\&quot; parent=\&quot;cFPUC0em5CPohN2MoM2B-4\&quot; vertex=\&quot;1\&quot; connectable=\&quot;0\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.0888\&quot; y=\&quot;-2\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Teacher\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1740\&quot; y=\&quot;-800\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-36\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Acronym\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1850\&quot; y=\&quot;-630\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-37\&quot; value=\&quot;&amp;amp;lt;&amp;amp;gt;&amp;lt;br&amp;gt;Class Duration\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;472.5\&quot; y=\&quot;-200\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-39\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Extra Class\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;592.5\&quot; y=\&quot;-650\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-40\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Open Date\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;221.5\&quot; y=\&quot;555\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-49\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Meeting\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1950\&quot; y=\&quot;440\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-50\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;SharedBoardID\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;3090\&quot; y=\&quot;150\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-51\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Shared Board Title\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;3090\&quot; y=\&quot;70\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-52\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Number of Columns\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;3090\&quot; y=\&quot;-90\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-54\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Meeting ID\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2140\&quot; y=\&quot;550\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-55\&quot; value=\&quot;&amp;amp;lt;ValueObject&amp;amp;gt;&amp;lt;br&amp;gt;Close Date\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;351.5\&quot; y=\&quot;555\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-56\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Section\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;170\&quot; y=\&quot;230\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-57\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Question\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;170\&quot; y=\&quot;130\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-58\&quot; value=\&quot;&amp;amp;lt;&amp;amp;gt;&amp;lt;br&amp;gt;Class Title\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;607.5\&quot; y=\&quot;-200\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-59\&quot; value=\&quot;&amp;amp;lt;&amp;amp;gt;&amp;lt;br&amp;gt;Class Date\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;737.5\&quot; y=\&quot;-200\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-60\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Password\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1960\&quot; y=\&quot;-90\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-61\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Email\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1820\&quot; y=\&quot;-90\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-63\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Exam Title\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;91.5\&quot; y=\&quot;555\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-64\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Mechanographic Number\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1220\&quot; y=\&quot;-670\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-71\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-60\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;969\&quot; y=\&quot;-29.999999999999943\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;980\&quot; y=\&quot;-110\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-72\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-61\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;930\&quot; y=\&quot;-29.999999999999943\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;990\&quot; y=\&quot;-100\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-76\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-64\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.3362\&quot; y=\&quot;3\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;820\&quot; y=\&quot;270\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;870\&quot; y=\&quot;220\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-77\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-36\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.3107\&quot; y=\&quot;4\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1197\&quot; y=\&quot;-449.7\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1197\&quot; y=\&quot;-379.7\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;-1\&quot; as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-83\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-26\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1196\&quot; y=\&quot;810\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1099\&quot; y=\&quot;880\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-85\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-56\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-57\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;440\&quot; y=\&quot;360\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;300\&quot; y=\&quot;360\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-86\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-63\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;450\&quot; y=\&quot;410\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;310\&quot; y=\&quot;410\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-87\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-56\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;460\&quot; y=\&quot;420\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;320\&quot; y=\&quot;420\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-88\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-40\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;470\&quot; y=\&quot;430\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;330\&quot; y=\&quot;430\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-89\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-55\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;480\&quot; y=\&quot;440\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;340\&quot; y=\&quot;440\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-92\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-51\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.4244\&quot; y=\&quot;1\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2465.3846153846152\&quot; y=\&quot;732.44\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2520\&quot; y=\&quot;752.44\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-93\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-50\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2435.3846153846152\&quot; y=\&quot;732.44\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2510\&quot; y=\&quot;812.44\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-94\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Exam Result\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-30\&quot; y=\&quot;360\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-95\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-94\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;210\&quot; y=\&quot;430\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;110\&quot; y=\&quot;500\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-96\&quot; value=\&quot;constitutes\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-94\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-29\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;180\&quot; y=\&quot;400\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;100\&quot; y=\&quot;400\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-102\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-28\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-59\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0047\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;902\&quot; y=\&quot;270\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;685\&quot; y=\&quot;340\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-106\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-28\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-58\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0047\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;785\&quot; y=\&quot;79.99999999999999\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;715\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-107\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-77\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0974\&quot; y=\&quot;7\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2515\&quot; y=\&quot;724.2581818181818\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;3000\&quot; y=\&quot;6.6666666666667425\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-108\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-52\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2395\&quot; y=\&quot;720.0009756097561\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2390\&quot; y=\&quot;822.44\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-109\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-38\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1170\&quot; y=\&quot;89.69999999999999\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1500\&quot; y=\&quot;49.69999999999999\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-38\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Full Name\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2140\&quot; y=\&quot;-90\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;gMcGTzHWLV\_Uvh6kmHHh-113\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-28\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-37\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0047\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;755\&quot; y=\&quot;79.99999999999999\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;690\&quot; y=\&quot;40\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-10\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Teacher In Charge\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1359.25\&quot; y=\&quot;995\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-11\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-10\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;939\&quot; y=\&quot;970\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1044\&quot; y=\&quot;1020\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-17\&quot; value=\&quot;&amp;amp;lt;&amp;amp;gt;&amp;lt;br&amp;gt;Extra ClassID\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;592.5\&quot; y=\&quot;-520\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-18\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-39\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-17\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0047\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;723\&quot; y=\&quot;140\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;767\&quot; y=\&quot;110\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-27\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Post It\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2820\&quot; y=\&quot;110\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-28\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-27\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2932\&quot; y=\&quot;60\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;3028\&quot; y=\&quot;130\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-29\&quot; value=\&quot;is&amp;amp;nbsp; created by\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;entryX=1;entryY=0.75;entryDx=0;entryDy=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0462\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2820\&quot; y=\&quot;145\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2510\&quot; y=\&quot;80\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;2570\&quot; y=\&quot;145\&quot; /&gt;\n &lt;/Array&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-30\&quot; value=\&quot;schedules/edits\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-49\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.4156\&quot; y=\&quot;3\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1090\&quot; y=\&quot;35\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1390\&quot; y=\&quot;120\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-31\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-49\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-54\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0012\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1881\&quot; y=\&quot;302.5\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2139\&quot; y=\&quot;492.5\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-32\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Invitation\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2140\&quot; y=\&quot;440\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-33\&quot; value=\&quot;sends\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-49\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-32\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1861\&quot; y=\&quot;682.5\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1929\&quot; y=\&quot;782.5\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-34\&quot; value=\&quot;is accepted/rejected by\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;cFPUC0em5CPohN2MoM2B-32\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.1253\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2530\&quot; y=\&quot;340\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2420\&quot; y=\&quot;241.77783203125\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-37\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Exam Header\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-40\&quot; y=\&quot;555\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-38\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-37\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;77.5\&quot; y=\&quot;420\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;178.5\&quot; y=\&quot;410\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-39\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Short Name\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2275\&quot; y=\&quot;-90\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-41\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-39\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.0003\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1175\&quot; y=\&quot;100\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1395\&quot; y=\&quot;-20\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-42\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Course ID\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1225.75\&quot; y=\&quot;995\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-43\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-42\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;948\&quot; y=\&quot;890\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;992\&quot; y=\&quot;985\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-45\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Small Textual Description\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1090\&quot; y=\&quot;995\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-46\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-45\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;899\&quot; y=\&quot;890\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;835\&quot; y=\&quot;945\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-48\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-51\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0708\&quot; y=\&quot;-6\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;655\&quot; y=\&quot;880\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;697.1530612244896\&quot; y=\&quot;1065\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1\&quot; as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-49\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Minimum Number of Students\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;820\&quot; y=\&quot;995\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-50\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-49\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;634\&quot; y=\&quot;880\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;565\&quot; y=\&quot;945\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-51\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Maximum Number of Students\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;950\&quot; y=\&quot;995\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-54\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Small Description\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-166.5\&quot; y=\&quot;555\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-55\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-54\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;-40\&quot; y=\&quot;420\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;52\&quot; y=\&quot;410\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-61\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-106\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.4286\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1330\&quot; y=\&quot;-449.7\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1330\&quot; y=\&quot;-229.70000000000005\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-68\&quot; value=\&quot;\&quot; style=\&quot;endArrow=block;endSize=16;endFill=0;html=1;rounded=0;entryX=0.575;entryY=0.009;entryDx=0;entryDy=0;entryPerimeter=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.1649\&quot; width=\&quot;160\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;940\&quot; y=\&quot;270\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1090\&quot; y=\&quot;280\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1360\&quot; y=\&quot;-890\&quot; /&gt;\n &lt;mxPoint x=\&quot;2119\&quot; y=\&quot;-890\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-70\&quot; value=\&quot;\&quot; style=\&quot;endArrow=block;endSize=16;endFill=0;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-34\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.0103\&quot; y=\&quot;5\&quot; width=\&quot;160\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1030\&quot; y=\&quot;370\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1030\&quot; y=\&quot;50\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-73\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-107\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.3489\&quot; y=\&quot;-2\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;997\&quot; y=\&quot;140\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;973\&quot; y=\&quot;350\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint y=\&quot;1\&quot; as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-75\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Meeting Duration\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1950\&quot; y=\&quot;550\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-76\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-49\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-75\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0012\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2280\&quot; y=\&quot;702.5\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2340\&quot; y=\&quot;702.5\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-77\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Number of Rows\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;3090\&quot; y=\&quot;-9.999999999999996\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-79\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Column\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2915\&quot; y=\&quot;-150\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-80\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Row\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2725\&quot; y=\&quot;-150\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-81\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-79\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1470\&quot; y=\&quot;-30\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1530\&quot; y=\&quot;-50\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-82\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-80\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1480\&quot; y=\&quot;-20\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1540\&quot; y=\&quot;-40\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-83\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;ID\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2725\&quot; y=\&quot;-280\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-84\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;cFPUC0em5CPohN2MoM2B-80\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-83\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2811\&quot; y=\&quot;130\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2779\&quot; y=\&quot;-80\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-85\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;cFPUC0em5CPohN2MoM2B-79\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.4666\&quot; y=\&quot;-2\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2821\&quot; y=\&quot;140\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2980\&quot; y=\&quot;-220\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-87\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;cFPUC0em5CPohN2MoM2B-80\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-129\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.4666\&quot; y=\&quot;2\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2605\&quot; y=\&quot;-200\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2850.3846153846152\&quot; y=\&quot;-220\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-88\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;cFPUC0em5CPohN2MoM2B-79\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-83\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2865\&quot; y=\&quot;-130\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;3109.6153846153848\&quot; y=\&quot;-220\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-92\&quot; value=\&quot;edits\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;entryX=0;entryY=0.5;entryDx=0;entryDy=0;exitX=1;exitY=0.5;exitDx=0;exitDy=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.1452\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2171\&quot; y=\&quot;121\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1360\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;2520\&quot; y=\&quot;130\&quot; /&gt;\n &lt;mxPoint x=\&quot;2520\&quot; y=\&quot;20\&quot; /&gt;\n &lt;/Array&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-94\&quot; value=\&quot;&amp;amp;lt;&amp;amp;gt;&amp;lt;br&amp;gt;Textual Description\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;40\&quot; y=\&quot;130\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-95\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-56\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-94\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;240\&quot; y=\&quot;240\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;240\&quot; y=\&quot;200\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-96\&quot; value=\&quot;&amp;amp;lt;&amp;amp;gt;&amp;lt;br&amp;gt;Question Type\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;170\&quot; y=\&quot;10\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-97\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-57\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-96\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;240\&quot; y=\&quot;240\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;240\&quot; y=\&quot;200\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-98\&quot; value=\&quot;Shared Board\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2650\&quot; y=\&quot;-320\&quot; width=\&quot;590\&quot; height=\&quot;550\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-100\&quot; value=\&quot;Person\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1790\&quot; y=\&quot;-260\&quot; width=\&quot;620\&quot; height=\&quot;460\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-101\&quot; value=\&quot;Meeting\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1900\&quot; y=\&quot;390\&quot; width=\&quot;400\&quot; height=\&quot;250\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-105\&quot; value=\&quot;Class\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;380\&quot; y=\&quot;-380\&quot; width=\&quot;570\&quot; height=\&quot;300\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-106\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Date of Birth\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1480\&quot; y=\&quot;-800\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-107\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Tax Payer Number\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1480\&quot; y=\&quot;-670\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-109\&quot; value=\&quot;Manager\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1360\&quot; y=\&quot;505\&quot; width=\&quot;210\&quot; height=\&quot;160\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-110\&quot; value=\&quot;Teacher\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1450\&quot; y=\&quot;-850\&quot; width=\&quot;540\&quot; height=\&quot;340.3\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-111\&quot; value=\&quot;Student\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1160\&quot; y=\&quot;-850\&quot; width=\&quot;470\&quot; height=\&quot;330\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-112\&quot; value=\&quot;Exam\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-320\&quot; y=\&quot;-50\&quot; width=\&quot;810\&quot; height=\&quot;690\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-113\&quot; value=\&quot;schedules\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-28\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1510\&quot; y=\&quot;180\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;-170\&quot; y=\&quot;370\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1790\&quot; y=\&quot;-920\&quot; /&gt;\n &lt;mxPoint x=\&quot;1030\&quot; y=\&quot;-920\&quot; /&gt;\n &lt;mxPoint x=\&quot;1030\&quot; y=\&quot;-410\&quot; /&gt;\n &lt;mxPoint x=\&quot;665\&quot; y=\&quot;-410\&quot; /&gt;\n &lt;/Array&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-115\&quot; value=\&quot;takes\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-39\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.0608\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1540\&quot; y=\&quot;180\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;735\&quot; y=\&quot;-310\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1380\&quot; y=\&quot;-500\&quot; /&gt;\n &lt;mxPoint x=\&quot;820\&quot; y=\&quot;-500\&quot; /&gt;\n &lt;mxPoint x=\&quot;820\&quot; y=\&quot;-640\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-117\&quot; value=\&quot;Extra Class\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;560\&quot; y=\&quot;-700\&quot; width=\&quot;190\&quot; height=\&quot;260\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-118\&quot; value=\&quot;takes\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-28\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.0967\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1350\&quot; y=\&quot;-490\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;735\&quot; y=\&quot;-320\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1370\&quot; y=\&quot;-510\&quot; /&gt;\n &lt;mxPoint x=\&quot;1090\&quot; y=\&quot;-510\&quot; /&gt;\n &lt;mxPoint x=\&quot;1090\&quot; y=\&quot;-340\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-121\&quot; value=\&quot;takes\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.533\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1350\&quot; y=\&quot;-480\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;735\&quot; y=\&quot;-330\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1390\&quot; y=\&quot;-490\&quot; /&gt;\n &lt;mxPoint x=\&quot;1110\&quot; y=\&quot;-490\&quot; /&gt;\n &lt;mxPoint x=\&quot;1110\&quot; y=\&quot;390\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-122\&quot; value=\&quot;take\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-39\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.5028\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1800\&quot; y=\&quot;-780\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;735\&quot; y=\&quot;-330\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1800\&quot; y=\&quot;-970\&quot; /&gt;\n &lt;mxPoint x=\&quot;653\&quot; y=\&quot;-970\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint x=\&quot;1\&quot; as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-124\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Evaluation\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-300\&quot; y=\&quot;555\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-125\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-124\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.0032\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;188\&quot; y=\&quot;430\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;-45\&quot; y=\&quot;565\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-127\&quot; value=\&quot;\&quot; style=\&quot;endArrow=block;endSize=16;endFill=0;html=1;rounded=0;exitX=1;exitY=0.5;exitDx=0;exitDy=0;entryX=0.4;entryY=0.014;entryDx=0;entryDy=0;entryPerimeter=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.0103\&quot; y=\&quot;5\&quot; width=\&quot;160\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1511\&quot; y=\&quot;390\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2120\&quot; y=\&quot;100\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;2100\&quot; y=\&quot;-770\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-129\&quot; value=\&quot;&amp;amp;lt;Value Object&amp;amp;gt;&amp;lt;br&amp;gt;Title\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;2915\&quot; y=\&quot;-280\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-132\&quot; value=\&quot;&amp;amp;lt;Entity&amp;amp;gt;&amp;lt;br&amp;gt;Enrollment\&quot; style=\&quot;rounded=0;whiteSpace=wrap;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1410\&quot; y=\&quot;-120\&quot; width=\&quot;120\&quot; height=\&quot;60\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-133\&quot; value=\&quot;creates\&quot; style=\&quot;endArrow=block;startArrow=block;endFill=1;startFill=1;html=1;rounded=0;entryX=0;entryY=0.25;entryDx=0;entryDy=0;exitX=1.007;exitY=0.176;exitDx=0;exitDy=0;exitPerimeter=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-31\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-27\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry width=\&quot;160\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;2440\&quot; y=\&quot;-70\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;2600\&quot; y=\&quot;-70\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;2500\&quot; y=\&quot;110\&quot; /&gt;\n &lt;mxPoint x=\&quot;2500\&quot; y=\&quot;5\&quot; /&gt;\n &lt;/Array&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-136\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-106\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.3489\&quot; y=\&quot;-2\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1402\&quot; y=\&quot;-770\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1518\&quot; y=\&quot;-660\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint y=\&quot;1\&quot; as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-140\&quot; value=\&quot;has\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-107\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.4286\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1750\&quot; y=\&quot;-760\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1610\&quot; y=\&quot;-760\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-143\&quot; value=\&quot;Enrollment\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;1350\&quot; y=\&quot;-170\&quot; width=\&quot;230\&quot; height=\&quot;140\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-147\&quot; value=\&quot;is accepted/rejected by\&quot; style=\&quot;endArrow=classic;html=1;rounded=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;cFPUC0em5CPohN2MoM2B-132\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-34\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.2424\&quot; width=\&quot;50\&quot; height=\&quot;50\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1370\&quot; y=\&quot;-770\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1440\&quot; y=\&quot;445.2382114955357\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1470\&quot; y=\&quot;80\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-149\&quot; value=\&quot;requests\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-32\&quot; target=\&quot;cFPUC0em5CPohN2MoM2B-132\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.1504\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1400\&quot; y=\&quot;-770\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;1430\&quot; y=\&quot;-270\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1400\&quot; y=\&quot;-400\&quot; /&gt;\n &lt;mxPoint x=\&quot;1470\&quot; y=\&quot;-400\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-150\&quot; value=\&quot;has\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-33\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;-0.1097\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;mxPoint x=\&quot;1160\&quot; y=\&quot;445\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;896\&quot; y=\&quot;610\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1116\&quot; y=\&quot;405\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-151\&quot; value=\&quot;has\&quot; style=\&quot;edgeStyle=orthogonalEdgeStyle;rounded=0;orthogonalLoop=1;jettySize=auto;html=1;entryX=0.5;entryY=0;entryDx=0;entryDy=0;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; source=\&quot;gMcGTzHWLV\_Uvh6kmHHh-35\&quot; target=\&quot;gMcGTzHWLV\_Uvh6kmHHh-30\&quot; edge=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;0.5572\&quot; y=\&quot;3\&quot; relative=\&quot;1\&quot; as=\&quot;geometry\&quot;&gt;\n &lt;Array as=\&quot;points\&quot;&gt;\n &lt;mxPoint x=\&quot;1780\&quot; y=\&quot;200\&quot; /&gt;\n &lt;mxPoint x=\&quot;1147\&quot; y=\&quot;200\&quot; /&gt;\n &lt;mxPoint x=\&quot;1147\&quot; y=\&quot;390\&quot; /&gt;\n &lt;/Array&gt;\n &lt;mxPoint x=\&quot;1770\&quot; y=\&quot;-730\&quot; as=\&quot;sourcePoint\&quot; /&gt;\n &lt;mxPoint x=\&quot;300\&quot; y=\&quot;390\&quot; as=\&quot;targetPoint\&quot; /&gt;\n &lt;mxPoint as=\&quot;offset\&quot; /&gt;\n &lt;/mxGeometry&gt;\n &lt;/mxCell&gt;\n &lt;mxCell id=\&quot;cFPUC0em5CPohN2MoM2B-155\&quot; value=\&quot;Course\&quot; style=\&quot;shape=umlFrame;whiteSpace=wrap;html=1;pointerEvents=0;width=100;height=30;\&quot; parent=\&quot;WIyWlLk6GJQsqaUBKTNV-1\&quot; vertex=\&quot;1\&quot;&gt;\n &lt;mxGeometry x=\&quot;790\&quot; y=\&quot;710\&quot; width=\&quot;850\&quot; height=\&quot;370\&quot; as=\&quot;geometry\&quot; /&gt;\n &lt;/mxCell&gt;\n &lt;/root&gt;\n &lt;/mxGraphModel&gt;\n &lt;/diagram&gt;\n&lt;/mxfile&gt;\n&quot;}"></div>  
<script type="text/javascript" src="https://viewer.diagrams.net/js/viewer-static.min.js"></script>  
</body>  
</html>

### [File Ends] sem4pi-22-23-61-master\libs\ModeloConceptual (1).drawio.html

### [File Begins] sem4pi-22-23-61-master\license.txt

Copyright (c) 2013-2023 the orinal author or authors.  
  
MIT License  
  
Permission is hereby granted, free of charge, to any person obtaining a copy  
of this software and associated documentation files (the "Software"), to deal  
in the Software without restriction, including without limitation the rights  
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
copies of the Software, and to permit persons to whom the Software is  
furnished to do so, subject to the following conditions:  
  
The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.  
  
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
SOFTWARE.

### [File Ends] sem4pi-22-23-61-master\license.txt

### [File Begins] sem4pi-22-23-61-master\mvnw

#!/bin/sh  
# ----------------------------------------------------------------------------  
# Licensed to the Apache Software Foundation (ASF) under one  
# or more contributor license agreements. See the NOTICE file  
# distributed with this work for additional information  
# regarding copyright ownership. The ASF licenses this file  
# to you under the Apache License, Version 2.0 (the  
# "License"); you may not use this file except in compliance  
# with the License. You may obtain a copy of the License at  
#  
# https://www.apache.org/licenses/LICENSE-2.0  
#  
# Unless required by applicable law or agreed to in writing,  
# software distributed under the License is distributed on an  
# "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY  
# KIND, either express or implied. See the License for the  
# specific language governing permissions and limitations  
# under the License.  
# ----------------------------------------------------------------------------  
  
# ----------------------------------------------------------------------------  
# Maven2 Start Up Batch script  
#  
# Required ENV vars:  
# ------------------  
# JAVA\_HOME - location of a JDK home dir  
#  
# Optional ENV vars  
# -----------------  
# M2\_HOME - location of maven2's installed home dir  
# MAVEN\_OPTS - parameters passed to the Java VM when running Maven  
# e.g. to debug Maven itself, use  
# set MAVEN\_OPTS=-Xdebug -Xrunjdwp:transport=dt\_socket,server=y,suspend=y,address=8000  
# MAVEN\_SKIP\_RC - flag to disable loading of mavenrc files  
# ----------------------------------------------------------------------------  
  
if [ -z "$MAVEN\_SKIP\_RC" ] ; then  
  
 if [ -f /etc/mavenrc ] ; then  
 . /etc/mavenrc  
 fi  
  
 if [ -f "$HOME/.mavenrc" ] ; then  
 . "$HOME/.mavenrc"  
 fi  
  
fi  
  
# OS specific support. $var \_must\_ be set to either true or false.  
cygwin=false;  
darwin=false;  
mingw=false  
case "`uname`" in  
 CYGWIN\*) cygwin=true ;;  
 MINGW\*) mingw=true;;  
 Darwin\*) darwin=true  
 #  
 # Look for the Apple JDKs first to preserve the existing behaviour, and then look  
 # for the new JDKs provided by Oracle.  
 #  
 if [ -z "$JAVA\_HOME" ] && [ -L /System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK ] ; then  
 #  
 # Apple JDKs  
 #  
 export JAVA\_HOME=/System/Library/Frameworks/JavaVM.framework/Versions/CurrentJDK/Home  
 fi  
  
 if [ -z "$JAVA\_HOME" ] && [ -L /System/Library/Java/JavaVirtualMachines/CurrentJDK ] ; then  
 #  
 # Apple JDKs  
 #  
 export JAVA\_HOME=/System/Library/Java/JavaVirtualMachines/CurrentJDK/Contents/Home  
 fi  
  
 if [ -z "$JAVA\_HOME" ] && [ -L "/Library/Java/JavaVirtualMachines/CurrentJDK" ] ; then  
 #  
 # Oracle JDKs  
 #  
 export JAVA\_HOME=/Library/Java/JavaVirtualMachines/CurrentJDK/Contents/Home  
 fi  
  
 if [ -z "$JAVA\_HOME" ] && [ -x "/usr/libexec/java\_home" ]; then  
 #  
 # Apple JDKs  
 #  
 export JAVA\_HOME=`/usr/libexec/java\_home`  
 fi  
 ;;  
esac  
  
if [ -z "$JAVA\_HOME" ] ; then  
 if [ -r /etc/gentoo-release ] ; then  
 JAVA\_HOME=`java-config --jre-home`  
 fi  
fi  
  
if [ -z "$M2\_HOME" ] ; then  
 ## resolve links - $0 may be a link to maven's home  
 PRG="$0"  
  
 # need this for relative symlinks  
 while [ -h "$PRG" ] ; do  
 ls=`ls -ld "$PRG"`  
 link=`expr "$ls" : '.\*-> \(.\*\)$'`  
 if expr "$link" : '/.\*' > /dev/null; then  
 PRG="$link"  
 else  
 PRG="`dirname "$PRG"`/$link"  
 fi  
 done  
  
 saveddir=`pwd`  
  
 M2\_HOME=`dirname "$PRG"`/..  
  
 # make it fully qualified  
 M2\_HOME=`cd "$M2\_HOME" && pwd`  
  
 cd "$saveddir"  
 # echo Using m2 at $M2\_HOME  
fi  
  
# For Cygwin, ensure paths are in UNIX format before anything is touched  
if $cygwin ; then  
 [ -n "$M2\_HOME" ] &&  
 M2\_HOME=`cygpath --unix "$M2\_HOME"`  
 [ -n "$JAVA\_HOME" ] &&  
 JAVA\_HOME=`cygpath --unix "$JAVA\_HOME"`  
 [ -n "$CLASSPATH" ] &&  
 CLASSPATH=`cygpath --path --unix "$CLASSPATH"`  
fi  
  
# For Migwn, ensure paths are in UNIX format before anything is touched  
if $mingw ; then  
 [ -n "$M2\_HOME" ] &&  
 M2\_HOME="`(cd "$M2\_HOME"; pwd)`"  
 [ -n "$JAVA\_HOME" ] &&  
 JAVA\_HOME="`(cd "$JAVA\_HOME"; pwd)`"  
 # TODO classpath?  
fi  
  
if [ -z "$JAVA\_HOME" ]; then  
 javaExecutable="`which javac`"  
 if [ -n "$javaExecutable" ] && ! [ "`expr \"$javaExecutable\" : '\([^ ]\*\)'`" = "no" ]; then  
 # readlink(1) is not available as standard on Solaris 10.  
 readLink=`which readlink`  
 if [ ! `expr "$readLink" : '\([^ ]\*\)'` = "no" ]; then  
 if $darwin ; then  
 javaHome="`dirname \"$javaExecutable\"`"  
 javaExecutable="`cd \"$javaHome\" && pwd -P`/javac"  
 else  
 javaExecutable="`readlink -f \"$javaExecutable\"`"  
 fi  
 javaHome="`dirname \"$javaExecutable\"`"  
 javaHome=`expr "$javaHome" : '\(.\*\)/bin'`  
 JAVA\_HOME="$javaHome"  
 export JAVA\_HOME  
 fi  
 fi  
fi  
  
if [ -z "$JAVACMD" ] ; then  
 if [ -n "$JAVA\_HOME" ] ; then  
 if [ -x "$JAVA\_HOME/jre/sh/java" ] ; then  
 # IBM's JDK on AIX uses strange locations for the executables  
 JAVACMD="$JAVA\_HOME/jre/sh/java"  
 else  
 JAVACMD="$JAVA\_HOME/bin/java"  
 fi  
 else  
 JAVACMD="`which java`"  
 fi  
fi  
  
if [ ! -x "$JAVACMD" ] ; then  
 echo "Error: JAVA\_HOME is not defined correctly." >&2  
 echo " We cannot execute $JAVACMD" >&2  
 exit 1  
fi  
  
if [ -z "$JAVA\_HOME" ] ; then  
 echo "Warning: JAVA\_HOME environment variable is not set."  
fi  
  
CLASSWORLDS\_LAUNCHER=org.codehaus.plexus.classworlds.launcher.Launcher  
  
# For Cygwin, switch paths to Windows format before running java  
if $cygwin; then  
 [ -n "$M2\_HOME" ] &&  
 M2\_HOME=`cygpath --path --windows "$M2\_HOME"`  
 [ -n "$JAVA\_HOME" ] &&  
 JAVA\_HOME=`cygpath --path --windows "$JAVA\_HOME"`  
 [ -n "$CLASSPATH" ] &&  
 CLASSPATH=`cygpath --path --windows "$CLASSPATH"`  
fi  
  
# traverses directory structure from process work directory to filesystem root  
# first directory with .mvn subdirectory is considered project base directory  
find\_maven\_basedir() {  
 local basedir=$(pwd)  
 local wdir=$(pwd)  
 while [ "$wdir" != '/' ] ; do  
 if [ -d "$wdir"/.mvn ] ; then  
 basedir=$wdir  
 break  
 fi  
 wdir=$(cd "$wdir/.."; pwd)  
 done  
 echo "${basedir}"  
}  
  
# concatenates all lines of a file  
concat\_lines() {  
 if [ -f "$1" ]; then  
 echo "$(tr -s '\n' ' ' < "$1")"  
 fi  
}  
  
export MAVEN\_PROJECTBASEDIR=${MAVEN\_BASEDIR:-$(find\_maven\_basedir)}  
MAVEN\_OPTS="$(concat\_lines "$MAVEN\_PROJECTBASEDIR/.mvn/jvm.config") $MAVEN\_OPTS"  
  
# Provide a "standardized" way to retrieve the CLI args that will  
# work with both Windows and non-Windows executions.  
MAVEN\_CMD\_LINE\_ARGS="$MAVEN\_CONFIG $@"  
export MAVEN\_CMD\_LINE\_ARGS  
  
WRAPPER\_LAUNCHER=org.apache.maven.wrapper.MavenWrapperMain  
  
exec "$JAVACMD" \  
 $MAVEN\_OPTS \  
 -classpath "$MAVEN\_PROJECTBASEDIR/.mvn/wrapper/maven-wrapper.jar" \  
 "-Dmaven.home=${M2\_HOME}" "-Dmaven.multiModuleProjectDirectory=${MAVEN\_PROJECTBASEDIR}" \  
 ${WRAPPER\_LAUNCHER} "$@"

### [File Ends] sem4pi-22-23-61-master\mvnw

### [File Begins] sem4pi-22-23-61-master\mvnw.cmd

@REM ----------------------------------------------------------------------------  
@REM Licensed to the Apache Software Foundation (ASF) under one  
@REM or more contributor license agreements. See the NOTICE file  
@REM distributed with this work for additional information  
@REM regarding copyright ownership. The ASF licenses this file  
@REM to you under the Apache License, Version 2.0 (the  
@REM "License"); you may not use this file except in compliance  
@REM with the License. You may obtain a copy of the License at  
@REM  
@REM https://www.apache.org/licenses/LICENSE-2.0  
@REM  
@REM Unless required by applicable law or agreed to in writing,  
@REM software distributed under the License is distributed on an  
@REM "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY  
@REM KIND, either express or implied. See the License for the  
@REM specific language governing permissions and limitations  
@REM under the License.  
@REM ----------------------------------------------------------------------------  
  
@REM ----------------------------------------------------------------------------  
@REM Maven2 Start Up Batch script  
@REM  
@REM Required ENV vars:  
@REM JAVA\_HOME - location of a JDK home dir  
@REM  
@REM Optional ENV vars  
@REM M2\_HOME - location of maven2's installed home dir  
@REM MAVEN\_BATCH\_ECHO - set to 'on' to enable the echoing of the batch commands  
@REM MAVEN\_BATCH\_PAUSE - set to 'on' to wait for a key stroke before ending  
@REM MAVEN\_OPTS - parameters passed to the Java VM when running Maven  
@REM e.g. to debug Maven itself, use  
@REM set MAVEN\_OPTS=-Xdebug -Xrunjdwp:transport=dt\_socket,server=y,suspend=y,address=8000  
@REM MAVEN\_SKIP\_RC - flag to disable loading of mavenrc files  
@REM ----------------------------------------------------------------------------  
  
@REM Begin all REM lines with '@' in case MAVEN\_BATCH\_ECHO is 'on'  
@echo off  
@REM enable echoing my setting MAVEN\_BATCH\_ECHO to 'on'  
@if "%MAVEN\_BATCH\_ECHO%" == "on" echo %MAVEN\_BATCH\_ECHO%  
  
@REM set %HOME% to equivalent of $HOME  
if "%HOME%" == "" (set "HOME=%HOMEDRIVE%%HOMEPATH%")  
  
@REM Execute a user defined script before this one  
if not "%MAVEN\_SKIP\_RC%" == "" goto skipRcPre  
@REM check for pre script, once with legacy .bat ending and once with .cmd ending  
if exist "%HOME%\mavenrc\_pre.bat" call "%HOME%\mavenrc\_pre.bat"  
if exist "%HOME%\mavenrc\_pre.cmd" call "%HOME%\mavenrc\_pre.cmd"  
:skipRcPre  
  
@setlocal  
  
set ERROR\_CODE=0  
  
@REM To isolate internal variables from possible post scripts, we use another setlocal  
@setlocal  
  
@REM ==== START VALIDATION ====  
if not "%JAVA\_HOME%" == "" goto OkJHome  
  
echo.  
echo Error: JAVA\_HOME not found in your environment. >&2  
echo Please set the JAVA\_HOME variable in your environment to match the >&2  
echo location of your Java installation. >&2  
echo.  
goto error  
  
:OkJHome  
if exist "%JAVA\_HOME%\bin\java.exe" goto init  
  
echo.  
echo Error: JAVA\_HOME is set to an invalid directory. >&2  
echo JAVA\_HOME = "%JAVA\_HOME%" >&2  
echo Please set the JAVA\_HOME variable in your environment to match the >&2  
echo location of your Java installation. >&2  
echo.  
goto error  
  
@REM ==== END VALIDATION ====  
  
:init  
  
set MAVEN\_CMD\_LINE\_ARGS=%\*  
  
@REM Find the project base dir, i.e. the directory that contains the folder ".mvn".  
@REM Fallback to current working directory if not found.  
  
set MAVEN\_PROJECTBASEDIR=%MAVEN\_BASEDIR%  
IF NOT "%MAVEN\_PROJECTBASEDIR%"=="" goto endDetectBaseDir  
  
set EXEC\_DIR=%CD%  
set WDIR=%EXEC\_DIR%  
:findBaseDir  
IF EXIST "%WDIR%"\.mvn goto baseDirFound  
cd ..  
IF "%WDIR%"=="%CD%" goto baseDirNotFound  
set WDIR=%CD%  
goto findBaseDir  
  
:baseDirFound  
set MAVEN\_PROJECTBASEDIR=%WDIR%  
cd "%EXEC\_DIR%"  
goto endDetectBaseDir  
  
:baseDirNotFound  
set MAVEN\_PROJECTBASEDIR=%EXEC\_DIR%  
cd "%EXEC\_DIR%"  
  
:endDetectBaseDir  
  
IF NOT EXIST "%MAVEN\_PROJECTBASEDIR%\.mvn\jvm.config" goto endReadAdditionalConfig  
  
@setlocal EnableExtensions EnableDelayedExpansion  
for /F "usebackq delims=" %%a in ("%MAVEN\_PROJECTBASEDIR%\.mvn\jvm.config") do set JVM\_CONFIG\_MAVEN\_PROPS=!JVM\_CONFIG\_MAVEN\_PROPS! %%a  
@endlocal & set JVM\_CONFIG\_MAVEN\_PROPS=%JVM\_CONFIG\_MAVEN\_PROPS%  
  
:endReadAdditionalConfig  
  
SET MAVEN\_JAVA\_EXE="%JAVA\_HOME%\bin\java.exe"  
  
set WRAPPER\_JAR="".\.mvn\wrapper\maven-wrapper.jar""  
set WRAPPER\_LAUNCHER=org.apache.maven.wrapper.MavenWrapperMain  
  
%MAVEN\_JAVA\_EXE% %JVM\_CONFIG\_MAVEN\_PROPS% %MAVEN\_OPTS% %MAVEN\_DEBUG\_OPTS% -classpath %WRAPPER\_JAR% "-Dmaven.multiModuleProjectDirectory=%MAVEN\_PROJECTBASEDIR%" %WRAPPER\_LAUNCHER% %MAVEN\_CMD\_LINE\_ARGS%  
if ERRORLEVEL 1 goto error  
goto end  
  
:error  
set ERROR\_CODE=1  
  
:end  
@endlocal & set ERROR\_CODE=%ERROR\_CODE%  
  
if not "%MAVEN\_SKIP\_RC%" == "" goto skipRcPost  
@REM check for post script, once with legacy .bat ending and once with .cmd ending  
if exist "%HOME%\mavenrc\_post.bat" call "%HOME%\mavenrc\_post.bat"  
if exist "%HOME%\mavenrc\_post.cmd" call "%HOME%\mavenrc\_post.cmd"  
:skipRcPost  
  
@REM pause the script if MAVEN\_BATCH\_PAUSE is set to 'on'  
if "%MAVEN\_BATCH\_PAUSE%" == "on" pause  
  
if "%MAVEN\_TERMINATE\_CMD%" == "on" exit %ERROR\_CODE%  
  
exit /B %ERROR\_CODE%

### [File Ends] sem4pi-22-23-61-master\mvnw.cmd

### [File Begins] sem4pi-22-23-61-master\pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://maven.apache.org/POM/4.0.0"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>eapli</groupId>  
 <artifactId>base</artifactId>  
 <version>1.4.0-SNAPSHOT</version>  
 <packaging>pom</packaging>  
  
 <properties>  
 <eapli.framework.core.version>v21.1.0-RELEASE</eapli.framework.core.version>  
 <eapli.framework.authz.version>v21.1.0-RELEASE</eapli.framework.authz.version>  
 <eapli.framework.pubsub.version>v21.1.0-RELEASE</eapli.framework.pubsub.version>  
   
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 <maven.compiler.source>11</maven.compiler.source>  
 <maven.compiler.target>11</maven.compiler.target>  
 <java.version>11</java.version>  
 </properties>  
  
 <modules>  
 <module>base.app.backoffice.console</module>  
 <module>base.app.user.console</module>  
 <module>base.app.other.console</module>  
 <module>base.app.common.console</module>  
 <module>base.app.bootstrap</module>  
 <module>base.bootstrappers</module>  
 <module>base.core</module>  
 <module>base.infrastructure.application</module>  
 <module>base.persistence.impl</module>  
 </modules>  
  
 <dependencies>  
 <dependency>  
 <groupId>org.bitbucket.pag\_isep.eapliframework</groupId>  
 <artifactId>eapli.framework.core</artifactId>  
 <version>${eapli.framework.core.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.bitbucket.pag\_isep.eapliframework</groupId>  
 <artifactId>eapli.framework.infrastructure.authz</artifactId>  
 <version>${eapli.framework.authz.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.bitbucket.pag\_isep.eapliframework</groupId>  
 <artifactId>eapli.framework.infrastructure.pubsub</artifactId>  
 <version>${eapli.framework.pubsub.version}</version>  
 </dependency>  
   
 <!-- keep junit4 for now -->  
 <dependency>  
 <groupId>org.junit.vintage</groupId>  
 <artifactId>junit-vintage-engine</artifactId>  
 <version>5.8.2</version>  
 <scope>test</scope>  
 </dependency>  
 <!-- support junit5 -->  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter-api</artifactId>  
 <version>5.8.2</version>  
 <scope>test</scope>  
 </dependency>  
   
 <dependency>  
 <groupId>com.h2database</groupId>  
 <artifactId>h2</artifactId>  
 <version>1.4.200</version>  
 <scope>runtime</scope>  
 </dependency>   
 <dependency>  
 <groupId>javax.xml.bind</groupId>  
 <artifactId>jaxb-api</artifactId>  
 <version>2.3.1</version>  
 </dependency>  
   
 <!-- Project Lombok -->  
 <dependency>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
 <version>1.18.24</version>  
 <scope>provided</scope>  
 </dependency>  
 </dependencies>  
  
 <dependencyManagement>  
 <dependencies>  
 </dependencies>  
 </dependencyManagement>  
  
 <repositories>  
 <repository>  
 <id>jitpack.io</id>  
 <url>https://jitpack.io</url>  
 </repository>  
 </repositories>  
   
 <build>  
 <plugins>  
 <!--  
 <plugin>  
 <groupId>org.pitest</groupId>  
 <artifactId>pitest-maven</artifactId>  
 <version>1.6.7</version>  
 </plugin>  
 -->  
   
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 <configuration>  
 <release>11</release>  
 <annotationProcessorPaths>  
 <path>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
 <version>1.18.24</version>  
 </path>  
 </annotationProcessorPaths>  
 </configuration>  
 </plugin>  
  
 <plugin>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok-maven-plugin</artifactId>  
 <version>1.18.20.0</version>  
 <executions>  
 <execution>  
 <id>delombok</id>  
 <phase>generate-sources</phase>  
 <goals>  
 <goal>delombok</goal>  
 </goals>  
 <configuration>  
 <sourceDirectory>${project.basedir}/src/main/java</sourceDirectory>  
 <outputDirectory>${project.build.directory}/delombok</outputDirectory>  
 <addOutputDirectory>false</addOutputDirectory>  
 <formatPreferences>  
 <javaLangAsFQN>skip</javaLangAsFQN>  
 </formatPreferences>  
 <verbose>false</verbose>  
 </configuration>  
 </execution>  
 <!-- This is for delomboking also your tests sources. <execution> <id>test-delombok</id>  
 <phase>generate-test-sources</phase> <goals> <goal>testDelombok</goal> </goals>  
 <configuration> <verbose>false</verbose> </configuration> </execution> -->  
 </executions>  
 </plugin>  
  
 <plugin>  
 <groupId>org.codehaus.mojo</groupId>  
 <artifactId>versions-maven-plugin</artifactId>  
 <version>2.5</version>  
 <configuration>  
 <generateBackupPoms>false</generateBackupPoms>  
 </configuration>  
 </plugin>  
 </plugins>  
 </build>  
</project>

### [File Ends] sem4pi-22-23-61-master\pom.xml

### [File Begins] sem4pi-22-23-61-master\readme.md

# Base Management  
 \_\_\_\_   
 | \_ \   
 | |\_) | \_\_ \_ \_\_\_ \_\_\_   
 | \_ < / \_` / \_\_|/ \_ \  
 | |\_) | (\_| \\_\_ \ \_\_/  
 |\_\_\_\_/ \\_\_,\_|\_\_\_/\\_\_\_|  
  
Engenharia de Aplicações (EAPLI)  
  
Polythecnic of Porto, School of Engineering  
  
---------------------------------------------  
  
This application is part of the lab project for the course unit EAPLI. Parts of   
the application were developed to show specific approaches or techniques; as such,   
the overall application is not consistent in terms of design. for production ready   
code this would obvisously be a problem as we should strive for consistency. In   
this case, it is acceptable as the inconsistencies are meant to provide samples   
of different valid approaches.  
  
\_Base logo created with [kammerl ascii signature](https://www.kammerl.de/ascii/AsciiSignature.php) using font "big"\_  
  
## Who do I talk to?  
  
Paulo Gandra de Sousa [pag@isep.ipp.pt](emailto:pag@isep.ipp.pt) / [pagsousa@gmail.com](emailto:pagsousa@gmail.com)  
  
## License and copyright  
  
Copyright (c) 2013-2023 the original author or authors.  
  
MIT License  
  
Permission is hereby granted, free of charge, to any person obtaining a copy  
of this software and associated documentation files (the "Software"), to deal  
in the Software without restriction, including without limitation the rights  
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
copies of the Software, and to permit persons to whom the Software is  
furnished to do so, subject to the following conditions:  
  
The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.  
  
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
SOFTWARE.  
  
## Build  
  
make sure Maven is installed and on the PATH  
  
If using an Oracle database, you will need to change your maven settings for   
downloading the Oracle drivers. see <https://blogs.oracle.com/dev2dev/entry/how\_to\_get\_oracle\_jdbc#settings> for more information.  
  
run script   
  
 rebuild-all.bat  
  
## Running  
  
make sure a JRE is installed and on the PATH  
  
run script   
  
 run-backoffice   
  
or   
  
 run-user.bat  
  
## Project structure  
  
- eapli.base.consoleapp  
  
 - presentation using console  
 - Main class   
 - application properties in resource folder  
  
- eapli.base.bootstrap   
 - bootstrap data. should be ignored on a "real" instalation  
  
- eapli.base.core   
 - use case controllers, model, and persistence  
  
## Architecture  
  
The application follows a typical layered approach  
  
 UI -> Controller -+-> Domain  
 | ^  
 | |  
 +-> Repositories  
  
  
### Domain objects with persistence knowledge or not  
  
Two different approaches are possible:  
  
- pure domain objects without any knowledge of the persistence  
- domain objects that can save and load thenselves from persistence (thus, an Active Record)  
  
In the first case, the controller is responsible for obtaining the domain objects   
from the repository, asking the domain objects to perform the business logic and   
then pass them back to the repository. in this case, the domain objects can "easily"   
be tested as they do not depend on any other package this gets trickier when we   
need/want to have lazy load of collections...  
  
In the second case, the controller asks the domain object class to load a certain   
instance, asks that object to perform the business operation and then asks the object   
to save itself back to the database  
  
### Passing domain objects to the UI or not  
  
The decision is to use domain objects outside of the controllers boundary. One could   
argue that domain objects should be known only "inside" the application boundary and   
as such other data structures should be returned to outside layers, i.e., DTO (Data Transfer Objects).  
  
### Performing calculations in memory or directly at the persistence layer  
  
Both approaches have advantages and disavantages:  
  
- in memory  
  
 - advantages  
  
 - allows the use of business logic in code  
 - disavantages performance may be poor  
  
- at persistence layer  
  
 - advantages  
  
 - use of aggregated SQL functions is straigth forward  
 - performance  
  
 - disavantages  
  
 - complicated business logic is hard to implement  
  
//TODO provide one example of each approach.  
  
See also <http://www.martinfowler.com/articles/dblogic.html>  
  
### Factoring out common behaviour  
  
use services at the application or domain layer  
  
### Can controllers call other controllers?  
  
it is best if they call application services  
  
### Should the UI/controller create domain objects directly  
  
Should the rules for the Creator pattern be fully enforced, e.g., the responsibility to   
create a Payment should be of Expense, or can the controller/UI create a Payment and  
 pass it to the Expense?  
  
### How to reuse behavior betwen controllers  
  
Factor out common behaviour in an application service.  
  
### When showing movements gruped by type, who performs the sum operation? UI, Controller or Domain object?  
  
the UI might not be smart enough to compute the total sum with enough precision, and   
would carry a burden for the computer running the interface  
  
the Controller might indeed perform such calculation as it has all the data is needs   
for a short period of time, but it is not the controller function to perform mathematical   
operations  
  
the domain object might indeed be the very best resource to calculate the sum for each   
expense type, but it would not make sense to delegate the domain object to the interface.  
  
### When showing movements gruped by type, should types with no values also appear?  
  
this can be another strategy  
  
## References and bibliography  
  
Start by reading the essential material listed in [EAPLI framework](https://bitbucket.org/pag\_isep/eapli.framework/src/master/README.md)  
  
### JPA  
  
- [Entities or DTOs in JPA Queries](https://thoughts-on-java.org/entities-dtos-use-projection/)  
- [Primary key mapping](https://thoughts-on-java.org/primary-key-mappings-jpa-hibernate/)  
  
### Other useful readings  
  
T.B.D.

### [File Ends] sem4pi-22-23-61-master\readme.md

## <-- File Content Ends