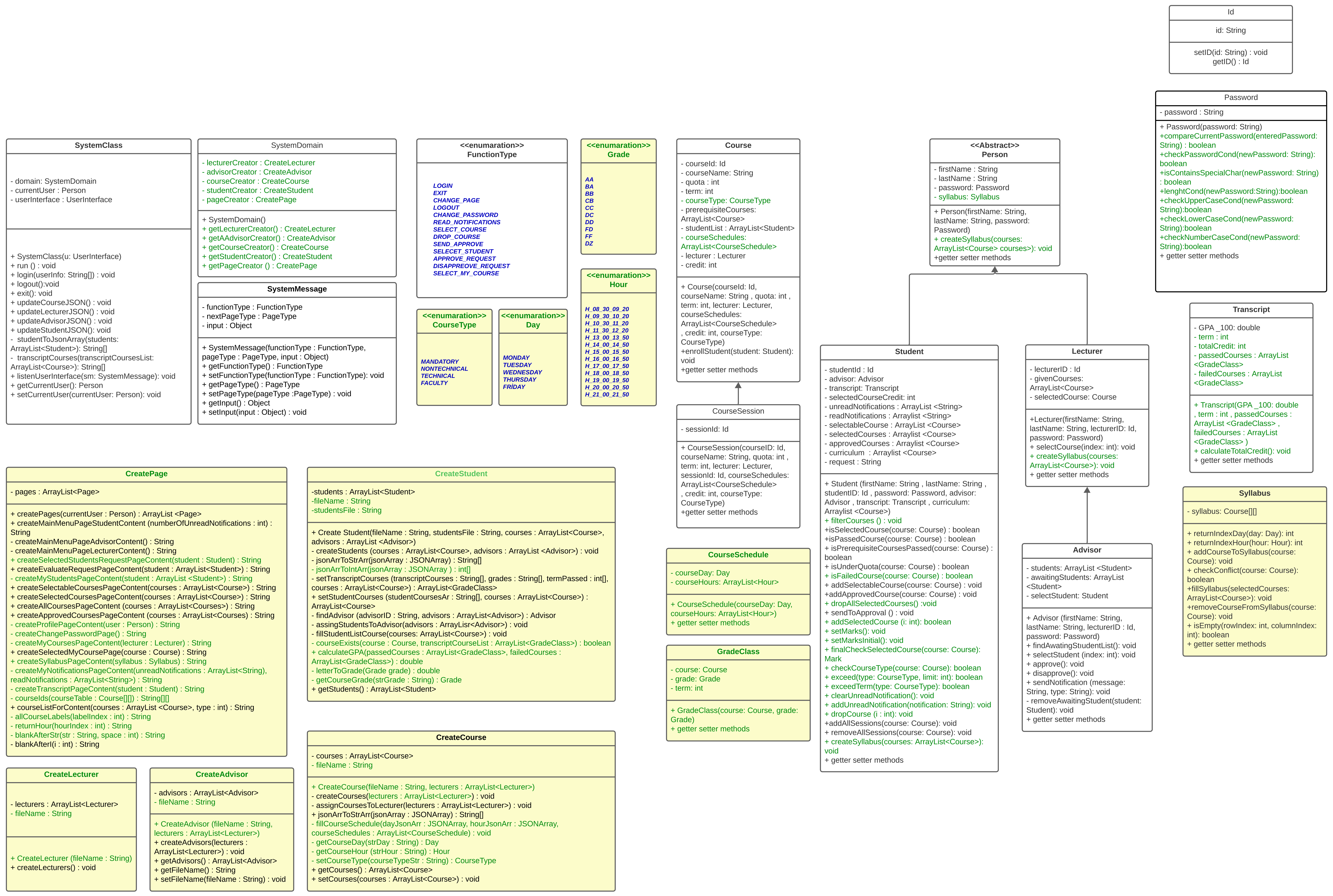
Id
id: String
setID(id: String) : void
getID() : Id
Password
- password : String
+ Password(password: String)
+compareCurrentPassword(enteredPassword: String) : boolean
+checkPasswordCond(newPassword: String): boolean
+isContainsSpecialChar(newPassword: String) : boolean
+lenghtCond(newPassword:String):boolean
+checkUpperCaseCond(newPassword: String):boolean
+checkLowerCaseCond(newPassword: String):boolean
+checkNumberCaseCond(newPassword: String):boolean
+ getter setter methods
SystemClass
- domain: SystemDomain
- currentUser : Person
- userInterface : UserInterface
+ SystemClass(u: UserInterface)
+ run () : void
+ login(userInfo: String[]) : void
+ logout():void
+ exit(): void
+ updateCourseJSON() : void
+ updateLecturerJSON() : void
+ updateAdvisorJSON() : void
+ updateStudentJSON(): void
- studentToJsonArray(students: ArrayList<Student>): String[]
- transcriptCourses(transcriptCoursesList: ArrayList<Course>): String[]
+ listenUserInterface(sm: SystemMessage): void
+ getCurrentUser(): Person
+ setCurrentUser(currentUser: Person): void
SystemDomain
- lecturerCreator : CreateLecturer
- advisorCreator : CreateAdvisor
- courseCreator : CreateCourse
- studentCreator : CreateStudent
- pageCreator : CreatePage
+ SystemDomain()
+ getLecturerCreator() : CreateLecturer
+ getAAdvisorCreator() : CreateAdvisor
+ getCourseCreator() : CreateCourse
+ getStudentCreator() : CreateStudent
+ getPageCreator () : CreatePage
SystemMessage
- functionType : FunctionType
- nextPageType : PageType
- input : Object
+ SystemMessage(functionType : FunctionType, pageType : PageType, input : Object)
+ getFunctionType() : FunctionType
+ setFunctionType(functionType : FunctionType): void
+ getPageType() : PageType
+ setPageType(pageType :PageType) : void
+ getInput() : Object
+ setInput(input : Object) : void
<<enumaration>>
FunctionType
LOGIN
EXIT
CHANGE_PAGE
LOGOUT
CHANGE_PASSWORD
READ_NOTIFICATIONS
SELECT_COURSE
DROP_COURSE
SEND_APPROVE
SELECET_STUDENT
APPROVE_REQUEST
DISAPPREOVE_REQUEST
SELECT_MY_COURSE
<<enumaration>>
Grade
AA
BA
BB
CB
CC
DC
DD
FD
FF
DZ
<<enumaration>>
Hour
H_08_30_09_20
H_09_30_10_20
H_10_30_11_20
H_11_30_12_20
H_13_00_13_50
H_14_00_14_50
H_15_00_15_50
H_16_00_16_50
H_17_00_17_50
H_18_00_18_50
H_19_00_19_50
H_20_00_20_50
H_21_00_21_50
<<enumaration>>
CourseType
MANDATORY
NONTECHNICAL
TECHNICAL
FACULTY
<<enumaration>>
Day
MONDAY
TUESDAY
WEDNESDAY
THURSDAY
FRİDAY
Course
- courseId: Id
- courseName: String
- quota : int
- term: int
- courseType: CourseType
- prerequisiteCourses: ArrayList<Course>
- studentList : ArrayList<Student>
- courseSchedules: ArrayList<CourseSchedule>
- lecturer : Lecturer
- credit: int
+ Course(courseId: Id, courseName: String , quota: int , term: int, lecturer: Lecturer, courseSchedules: ArrayList<CourseSchedule> , credit: int, courseType: CourseType)
+enrollStudent(student: Student): void
+getter setter methods
CourseSession
- sessionId: Id
+ CourseSession(courseID: Id, courseName: String, quota: int , term: int, lecturer: Lecturer, sessionId: Id, courseSchedules: ArrayList<CourseSchedule> , credit: int, courseType: CourseType)
+getter setter methods
<<Abstract>>
Person
- firstName : String
- lastName : String
- password: Password
- syllabus: Syllabus
+ Person(firstName: String, lastName: String, password: Password)
+ createSyllabus(courses: ArrayList<Course> courses>): void
+getter setter methods
Student
- studentId : Id
- advisor: Advisor
- transcript: Transcript
- selectedCourseCredit: int
- unreadNotifications : ArrayList <String>
- readNotifications : Arraylist <String>
- selectableCourse : ArrayList <Course>
- selectedCourses : Arraylist <Course>
- approvedCourses : Arraylist <Course>
- curriculum : Arraylist <Course>
- request : String
+ Student (firstName: String , lastName: String , studentID: Id , password: Password, advisor: Advisor , transcript: Transcript , curriculum: Arraylist <Course>)
+ filterCourses () : void
+isSelectedCourse(course: Course) : boolean
+isPassedCourse(course: Course) : boolean
+ isPrerequisiteCoursesPassed(course: Course) : boolean
+ isUnderQuota(course: Course) : boolean
+ isFailedCourse(course: Course) : boolean
+ addSelectableCourse(course: Course) : void
+addApprovedCourse(course: Course) : void
+ dropAllSelectedCourses() :void
+ sendToApproval () : void
+ addSelectedCourse (i: int): boolean
+ setMarks(): void
+ setMarksInitial(): void
+ finalCheckSelectedCourse(course: Course): Mark
+ checkCourseType(course: Course): boolean
+ exceed(type: CourseType, limit: int): boolean
+ exceedTerm(type: CourseType): boolean
+ clearUnreadNotification(): void
+ addUnreadNotification(notification: String): void
+ dropCourse (i : int): void
+addAllSessions(course: Course): void
+ removeAllSessions(course: Course): void
+ createSyllabus(courses: ArrayList<Course>): void
+ getter setter methods
Lecturer
- lecturerID : Id
- givenCourses: ArrayList<Course>
- selectedCourse: Course
+Lecturer(firstName: String, lastName: String, lecturerID: Id, password: Password)
+ selectCourse(index: int): void
+ createSyllabus(courses: ArrayList<Course>): void
+ getter setter methods
Advisor
- students: ArrayList <Student>
- awaitingStudents: ArrayList <Student>
- selectStudent: Student
+ Advisor (firstName: String, lastName: String, lecturerID : Id, password: Password)
+ findAwatingStıudentList(): void
+ selectStudent (index: int): void
+ approve(): void
+ disapprove(): void
+ sendNotification (message: String, type: String): void
- removeAwaitingStudent(student: Student): void
+ getter setter methods
Transcript
- GPA _100: double
- term : int
- totalCredit: int
- passedCourses : ArrayList <GradeClass>
- failedCourses : ArrayList <GradeClass>
+ Transcript(GPA _100: double , term : int , passedCourses : ArrayList <GradeClass> , failedCourses : ArrayList <GradeClass>)
+ calculateTotalCredit(): void
+ getter setter methods
Syllabus
- syllabus: Course[][]
+ returnIndexDay(day: Day): int
+ returnIndexHour(hour: Hour): int
+ addCourseToSyllabus(course: Course): void
+ checkConflict(course: Course): boolean
+fillSyllabus(selectedCourses: ArrayList<Course>): void
+removeCourseFromSyllabus(course: Course): void
+ isEmpty(rowIndex: int, columnIndex: int): boolean
+ getter setter methods
CourseSchedule
- courseDay: Day
- courseHours: ArrayList<Hour>
+ CourseSchedule(courseDay: Day, courseHours: ArrayList<Hour>)
+ getter setter methods
GradeClass
- course: Course
- grade: Grade
- term: int
+ GradeClass(course: Course, grade: Grade)
+ getter setter methods
CreatePage
- pages : ArrayList<Page>
+ createPages(currentUser : Person) : ArrayList <Page>
+ createMainMenuPageStudentContent (numberOfUnreadNotifications : int) : String
- createMainMenuPageAdvisorContent() : String
- createMainMenuPageLecturerContent() : String
+ createSelectedStudentsRequestPageContent(student : Student) : String
+ createEvaluateRequestPageContent(student : ArrayList<Student>) : String
- createMyStudentsPageContent(student : ArrayList <Student>) : String
+ createSelectableCoursesPageContent(courses : ArrayList<Course>) : String
+ createSelectedCoursesPageContent(courses : ArrayList<Course>) : String
+ createAllCoursesPageContent (courses : ArrayList<Courses>) : String
+ createApprovedCoursesPageContent (courses : ArrayList<Courses) : String
- createProfilePageContent(user : Person) : String
- createChangePasswordPage() : String
- createMyCoursesPageContent(lecturer : Lecturer) : String
+ createSelectedMyCoursePage(course : Course) : String
+ createSyllabusPageContent(syllabus : Syllabus) : String
- createMyNotificationsPageContent(unreadNotifications : ArrayList<String), readNotifications : ArrayList<String>) : String
- createTranscriptPageContent(student : Student) : String
- courseIds(courseTable : Course[][]) : String[][]
+ courseListForContent(courses : ArrayList <Course>, type : int) : String
- allCourseLabels(labelIndex : int) : String
- returnHour(hourIndex : int) : String
- blankAfterStr(str : String, space : int) : String
- blankAfterI(i : int) : String
CreateStudent
-students : ArrayList<Student>
-fileName : String
-studentsFile : String
+ Create Student(fileName : String, studentsFile : String, courses : ArrayList<Course>, advisors : ArrayList <Advisor>)
- createStudents (courses : ArrayList<Course>, advisors : ArrayList <Advisor>) : void
- jsonArrToStrArr(jsonArray : JSONArray) : String[]
- jsonArrToIntArr(jsonArray : JSONArray) : int[]
- setTranscriptCourses (transcriptCourses : String[], grades : String[], termPassed : int[], courses : ArrayList<Course>) : ArrayList<GradeClass>
+ setStudentCourses (studentCoursesAr : String[], courses : ArrayList<Course>) : ArrayList<Course>
- findAdvisor (advisorID : String, advisors : ArrayList<Advisor>) : Advisor
- assingStudentsToAdvisor(advisors : ArrayLisr<Advisor>) : void
- fillStudentListCourse(courses: ArrayList<Course>) : void
- courseExists(course : Course, transcriptCourseList : ArrayList<GradeClass>) : boolean
+ calculateGPA(passedCourses : ArrayList<GradeClass>, failedCourses : ArrayList<GradeClass>) : double
- letterToGrade(Grade grade) : double
- getCourseGrade(strGrade : String) : Grade
+ getStudents() : ArrayList<Student>
CreateCourse
- courses : ArrayList<Course>
- fileName : String
+ CreateCourse(fileName : String, lecturers : ArrayList<Lecturer>)
- createCourses(lecturers : ArrayList<Lecturer>) : void
- assignCoursesToLecturer(lecturers : ArrayList<Lecturer>) : void
+ jsonArrToStrArr(jsonArray : JSONArray) : String[]
- fillCourseSchedule(dayJsonArr : JSONArray, hourJsonArr : JSONArray, courseSchedules : ArrayList<CourseSchedule) : void
- getCourseDay(strDay : String) : Day
- getCourseHour (strHour : String) : Hour
- setCourseType(courseTypeStr : String) : CourseType
+ getCourses() : ArrayList<Course>
+ setCourses(courses : ArrayList<Course>) : void
CreateLecturer
- lecturers : ArrayList<Lecturer>
- fileName : String
+ CreateLecturer (fileName : String)
+ createLecturers() : void
CreateAdvisor
- advisors : ArrayList<Advisor>
- fileName : String
+ CreateAdvisor (fileName : String, lecturers : ArrayList<Lecturer>)
+ createAdvisors(lecturers : ArrayList<Lecturer>) : void
+ getAdvisors() : ArrayList<Advisor>
+ getFileName() : String
+ setFileName(fileName : String) : void