Bilkent University

Department of Computer Engineering

CS 319- Object-Oriented Software Engineering

CATCH UP

Project Design Report

Group 1H

İlke Kaş, 21803184

Zeynep Büşra Ziyagil, 21802646

Bilgehan Akcan, 21802901

Yaren Yılmaz, 21803561

Ömer Onat Postacı, 21802349

Instructor: Eray Tüzün

Teaching Assistants: Erdem Tuna, Elgun Jabrayilzade

Contents

Design Report

CatchUp: Classroom Helper

● Introduction

1.1 Purpose of the system

CatchUp is a classroom helper web-based application that ensures a sophisticated interaction between the instructors, teaching assistants, and students of a course. It ensures students to keep track of their project artifacts, and assignments, while it also ensures instructors and teaching assistants to keep track of the work of the project artifacts of different student groups. The application also provides a channel-based messaging platform to ensure the communication between students, group members, and teaching assistants.

When the instructor creates a course, students and teaching assistants enroll in the course. Hereby, the instructor of the course can operate the course by assigning artifacts, sharing lecture contents, creating polls, and communicating with each student or teaching assistant; whereas students can participate in the course by uploading their work, reviewing their peers, and also works of other groups.

In this project, the purpose is to implement a classroom helper application that ensures a more advanced interaction between students, teaching assistants, and instructors to sustain a course with a term project with artifacts.

1.2 Design goals

1.2.1 Usability

In CatchUp, there are various functions- some of them are unique to user types -of different user types such as students, teaching assistants, and instructors. Due to the fact that the necessary criteria to be a usable application consist of user satisfaction, effectiveness, and efficiency, the events in Catchup are executed with minimum and optimum time for being more usable. Besides, even though the application has lots of functions for 3 user types, the interface of the application will be easy and user-friendly to satisfy the user. Thereby, the architecture and navigation of the site will be understood with minimum effort by the user.

1.2.2 Reliability

Executing the functions with minimum failure is one of the most essential points of the reliability of our software since the software consists of various calculations and data operations. As mentioned, there are various functions for different user types and each user type has special functions. Therefore, data operations and executing functions with minimum failure is important.

1.2.3 Performance

The essential points of the performance of the software are response time, latency, and server connection. The time to process a request should be low to have a good performance. The tasks and functions will be executed without any inefficiency

to avoid the latency. In addition, the data connection and the requests from the users to the Amazon server will be less than or equal to 1 seconds.

1.2.4 Supportability

CatchUp will be an extendable system that has the ability of modification or contribution to the program- as other extendable systems -without disrupting the structure of the system. With the feedback from users, appropriate modifications and additions can be applied to the system.

1.2.5 Functionality

Classroom helper applications consist of various features to increase the interaction between the students, TAs, and the instructors. However, our design has extra features compared to other classroom helper applications to attract the attention of the user and increase the efficiency in the courses with projects.

● High-level software architecture

2.1 Overview

In this part of the report, we divide our system to the subsystems which makes this design more reliable, understandable and writable for our team members. In our project we will use the three-tier design architecture because we think that it suits best for our classroom helper application.

2.2 Subsystem decomposition

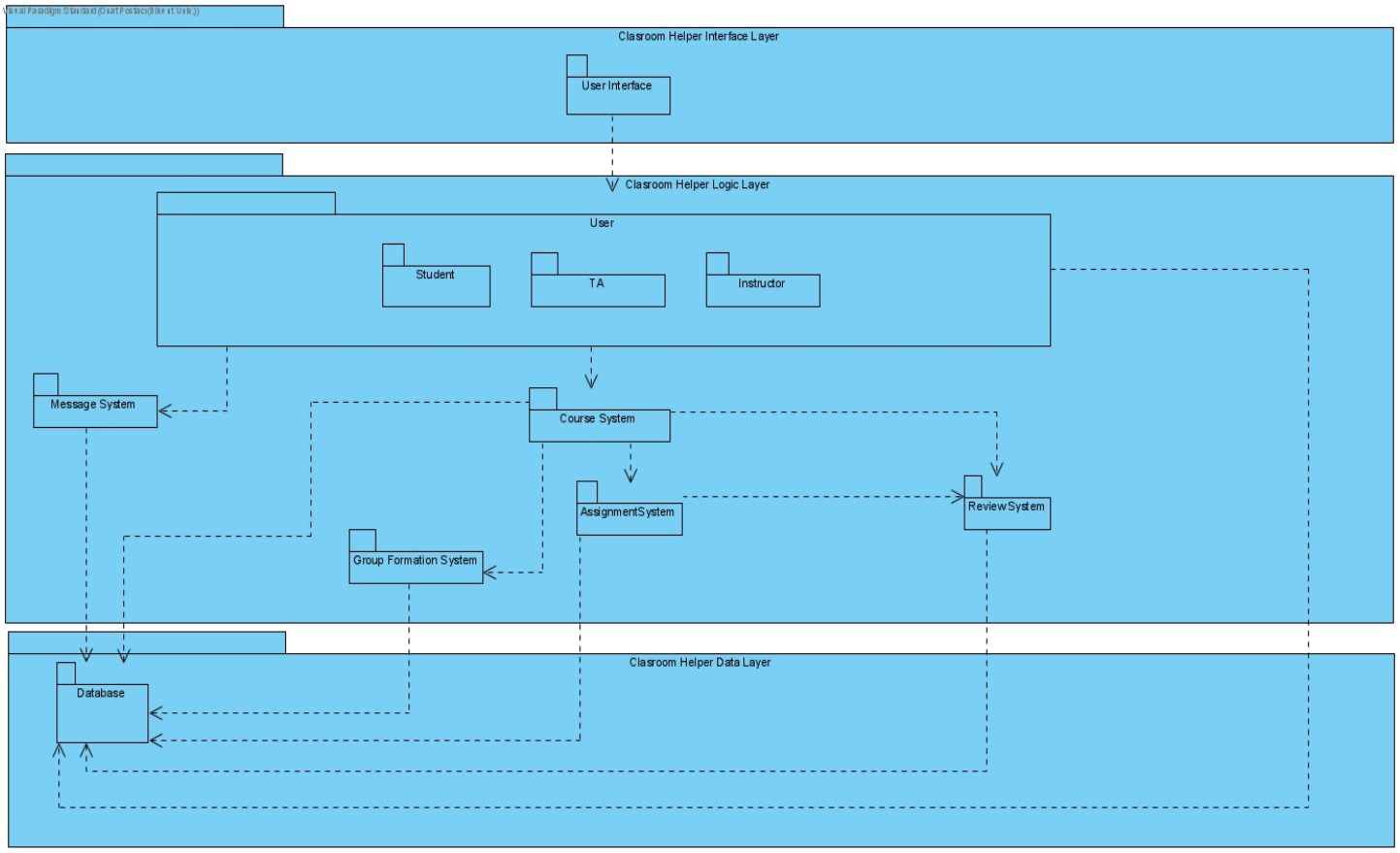


Figure 1

The system can be decomposed into three subsystems which are Classroom Helper Interface Layer, Classroom Helper Logic Layer, Classroom Helper Data Layer. Classroom Helper Interface Layer organizes the classes that constitute the user interface by acting together. Classroom Helper Logic Layer controls the application according to the user's actions. Classroom Helper Data Layer Subsystem constructs and controls the interactions between the objects of the system.

2.3 Architectural Styles

2.3.1 Layers

In the decomposition process of the system we have used a three tier architectural style which is a common way to decompose websites. We have the

interface layer on top and then application logic layer and its storage layer below. Application layer is the main controller which can interact with the other two both.

2.4 Hardware/software mapping

A server is needed to host the web application. Also, database will be required to store the users' and system's data. In order to make use of the database, PostgreSQL will be used. For the front-end part of the project, React.js which is a Javascript library will be used. Finally, for the back-end part of the project, Java8 will be used and it will be the main tool that controls the entire system. We will use IntellIJ as IDE. Catch-up requires a basic mouse and keyboard. It will work on all kinds of browsers such as Chrome, Safari, Firefox, Opera, Microsoft Edge.

2.5 Persistent data management

Our design has various data types including persistent data such as email, name, surname, and id. We will store all of our data in our database system which is formed using PostgreSQL. However, even though some of our data, such as password, can be altered, persistent data will not be changed and will be persistent. We will store classes and their attributes as data values in the database. These are conversations , messages, users , courses, assignments, advertisements, artifacts, course documents, polls, announcements and project groups. All of their information that is necessary for system maintainability will be stored in the database.

2.6 Access control and security

Users have to verify their information to sign up to the system, and also they have to authenticate themselves to sign in to the application.

	Assignment	Announcement	Poll	Course Document	Artifact	Time Table	Calendar	Peer Review	Artifact Review	Message	Advertisement
Instructor	RWD	RWD	RWD	RWD	R	RW	W	R	RWD	RW	R
Teaching Assistant	RWD	R	R	RWD	R	R	RW	R	RWD	RW	R
Student	R	R	RE	R	RWD	R	RW	RW	RWD	RW	RWD
Project Group	R	R	R	R	RWD	R	RW		RWD	R	RWD

Figure 2

R: Read

W: Write

D: Delete

E: Execute

Figure 2 demonstrates the access control matrix of the system. Instructor can generally read and write most of the objects but can read only Artifact, Peer Review and Advertisement. Besides that Instructor can write the calendar of students and teaching assistants by giving them deadlines. Teaching assistant can read all of the objects. They only write or delete assignments, course documents and artifact reviews. They also write messages like many other subjects, however none of the subjects can delete the message they wrote. Students can read all objects but only execute polls. They can write artifacts, calendar, both reviews, advertisements and messages. Finally, project groups can read most of the classes.

2.7 Boundary conditions

2.7.1 Initialization

To access the web-based application, users need to have an Internet connection and the domain name of the catch-up has to be used.

2.7.2 Termination

Users can log out the application by pressing the log out button that appears when the menu icon is pressed. Also, if there is no operation done in the application for a while, the system automatically logs out the user.

2.7.3 Failure

Catch Up requires Internet connection to access data or to make change in the application. In the case of Internet connection loss, the user will not be able to access the application. Also, the changes done in the application during the connection loss will not be saved and all the changes that are not saved before the connection loss will be lost.

● Subsystem Services

3.1 Design Pattern

We have used two types of design patterns, namely, Strategy Design Pattern and Façade Design Pattern. First of all, Strategy Design Pattern is used to encapsulate algorithms and helps to choose algorithms in runtime. Therefore, in some parts of our project, it is appropriate to use it. For example, in our application, there is a poll feature and questions can be added one by one into the poll. Whether a multiple choice question or an open-ended question is selected needs to be decided. Therefore,

Strategy Design Pattern is the best solution to decide which algorithm will be used in runtime. Another part where we have used the Strategy Design Pattern is the Student package which is inside the User package. A student can give an advertisement to find a group or to find a group member if he/she has already a group. Therefore, a choice needs to be made once again. In order to decide which one will be selected in runtime, Strategy Design Pattern is used. The last part where we have used the Strategy Design Pattern is about messaging. Since a student can send a message to an individual or to a message group, a choice needs to be made and Strategy Design Pattern helps us to decide which one to be selected in runtime.

Another design pattern that we used in our application is the Façade Design Pattern. It is used to decrease the complexity of the program. It prefers to simplify the methods and call functions from other classes so that complexity is decreased and process is carried out in the background. For example, in our application, it is used in the Classroom Helper Data Layer. In that subsystem, DatabaseManager class has objects of some other classes such as MessageManagerData or ReviewManagerData, and through these objects, DatabaseManager class calls functions of those classes without any implementation inside itself. That situation decreases the complexity and makes the system more manageable.

3.2 Classroom Helper Interface Layer

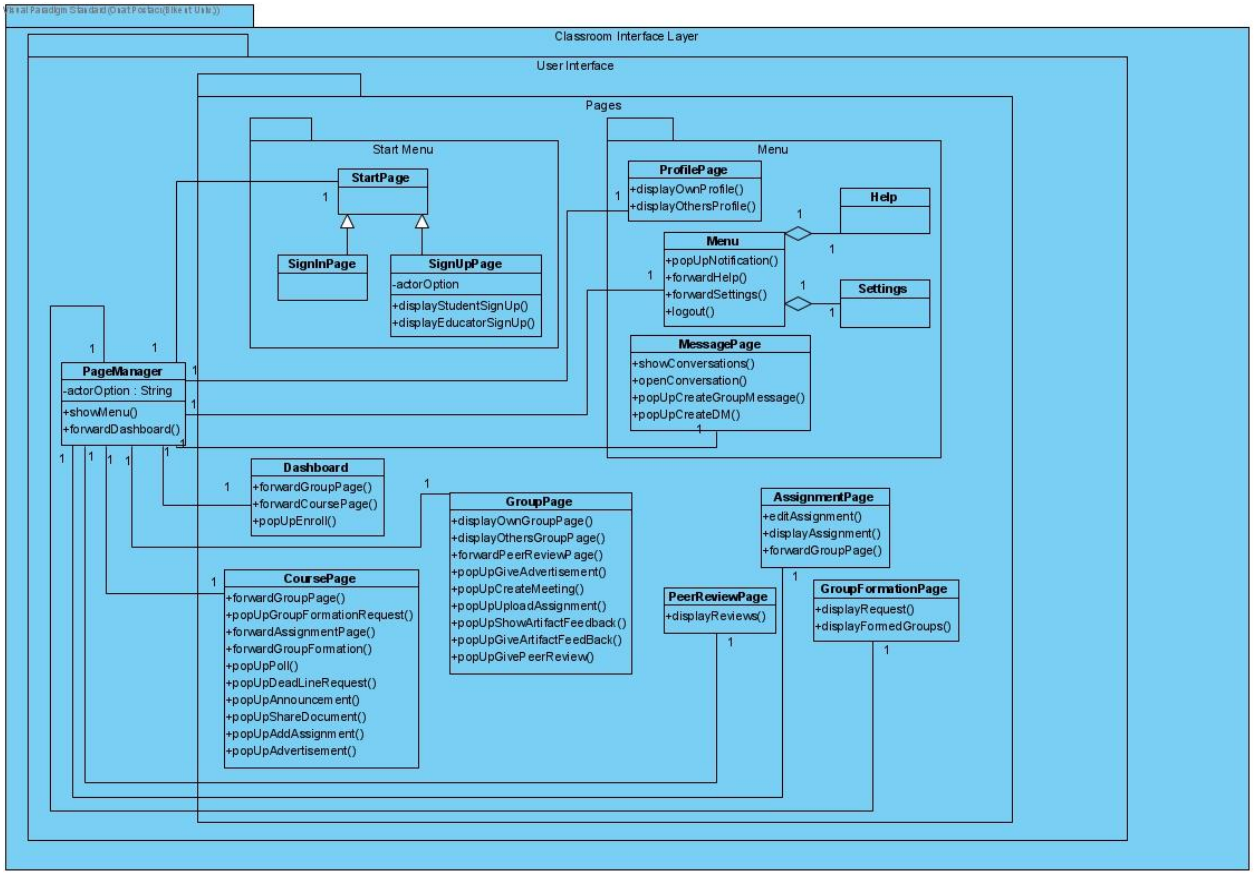


Figure 3

There is only one common User Interface package in this layer because in our system all different users see almost the same pages.

PageManager Class

This class manages all of the pages in the user interface and controls which page to go from any page . It enables users to have menu options and return to the dashboard option on every page.

Attributes

- String actorOption: The data that holds the actor type of the actions.

Operations

- public void showMenu(): Shows all of the menu items.

- public void forwardDashboard(): Forwards the user to the dashboard page.

StartPage Class

This class represents the first page when a user encounters when s/he enters the system, this is the parent version of sign in and sign up class and they inherit the start page class.

SignInPage Class

This class represents the sign in page of user interface which is shown when the user enters the system.

SignUpPage Class

This class represents the sign up page of the user interface which is shown when the user first enters the system if the user does not have an account already.

Attributes

- private String actorOption: Actor role of the user in the system. Sign up information will be entered according to this choice.

Operations

- public void displayStudentSignUp(): Displays the necessary sign up information for students.
- public void displayEducatorSignUp():Displays the necessary sign up information for educators.

Dashboard Class

This class represents the dashboard page of user interface which is shown after the user signed in or signed up to the system. Also, users can select the menu, existing course page or group page from the dashboard page. Users can enroll or create- according to user type- a new course from the dashboard page.

Operations

- public void forwardGroupPage(): Forwards the user to the group page.
- public void forwardCoursePage(): Forwards the user to the course page.
- public void popUpEnroll(): Pops up enroll course form.

CoursePage Class

This class represents the course page of the user interface which is shown after the user selects this page from the dashboard page. Also, users can select the menu or one of the groups' pages from the course page. Course page includes course information and other course related features.

Operations

- public void forwardGroupPage(): Forwards the user to group pages.
- public void popUpGroupFormationRequest(): Enables students to request for a group and form it.
- public void forwardAssignmentPage(): Forwards the instructor to assignment page.
- public void forwardGroupFormationPage(): Forwards the teaching assistant to group pages.

- public void popUpPoll(): Enables students to answer poll questions, and enables instructors to add questions to polls.
- public void popUpDeadLineRequest(): Enables instructors to see deadline requests.
- public void popUpAnnouncement(): Enables students to see announcements while enabling instructors to write announcements.
- public void popShareDocument():Enables instructors to add documents.
- public void popUpAddAssignment():Enables instructors to add assignments.
- public void popUpAdvertisement():Enables students to add assignments.

GroupPage Class

This class represents the group page of the user interface which is shown after the user selects this page from the dashboard page or course page. Also, users can select the menu from the course page. This class includes group related documents and features.

Operations

- public void displayOwnGroupPage(): Displays the group page which user is a member of.
- public void displayOthersGroupPage(): Displays the group page which the user is not a member of.
- public void forwardPeerReviewPage(): Forwards the teaching assistants or instructors to the peer review page.

- public void popUpGiveAdvertisement(): Enables students to answer poll questions, and enables instructors to add questions to polls.
- public void popUpCreateMeeting(): Enables students to create meetings.
- public void popUpUploadAssignment():Enables students to upload assignments.
- public void popUpShowArtifactFeedback(): Shows the given feedback.
- public void popUpGiveArtifactFeedback():Enables users to give feedback.
- public void popUpGivePeerReview():Enables students to review their peers.

PeerReviewPage Class

This class is to show peer reviews of one group all together at once. Instructors can see all the reviews given and received by students of the group.

Operations

- public void displayReviews(): Shows all of the reviews.

AssignmentPage Class

This class represents the assignments given deadline extension requests are shown here. Besides, editing and giving extension processes are getting done on this page.

Operations

- public void displayAssignment(): Shows the assignment, detailed.
- public void editAssignment(): Enables instructor to edit assignment.
- public void forwardGroupPage():Forwards to necessary group page.

GroupFormationPage Class

This class is created to enable teaching assistants to form project groups. Created groups and requests are shown right here.

Operations

- public void displayRequest(): Shows the requests of students, detailed.
- public void displayFormedGroups(): Shows the formed groups.

ProfilePage Class

This class represents the profile pages of the users when they select the go profile icon from the menu they are forwarded to their own profile. Otherwise, when they select and click on another profile they are also forwarded to this page but the profile they have chosen to go is shown.

Operations

- public void displayOwnProfile(): Displays the profile of the user.
- public void displayOthersProfile(): Displays the profile of the other users.

MessagePage Class

This class represents the message page of the user interface which is shown as one of the icons in the menu above the page. This page includes the conversation of users.

Operations

- public void showConversations(): Displays the conversations one has in groups or as pairs.
- public void openConversation(): Displays the messages of conversations.
- public void popUpCreateGroupMessage(): Enables users to form groups.
- public void popUpCreateDM(): Enables users to compose direct messages.

Menu Class

This class represents the menu icon above the pages. When the user clicks the menu there is a list of options which are settings, help and logout.

Operations

- public void logout(): Displays the option for user to logout.
- public void forwardHelp(): Forwards user to the help page.
- public void forwardSettings(): Forwards user to the settings page.
- public void popUpNotification(): Displays the notifications of the user from the bell icon as a pop-up window.

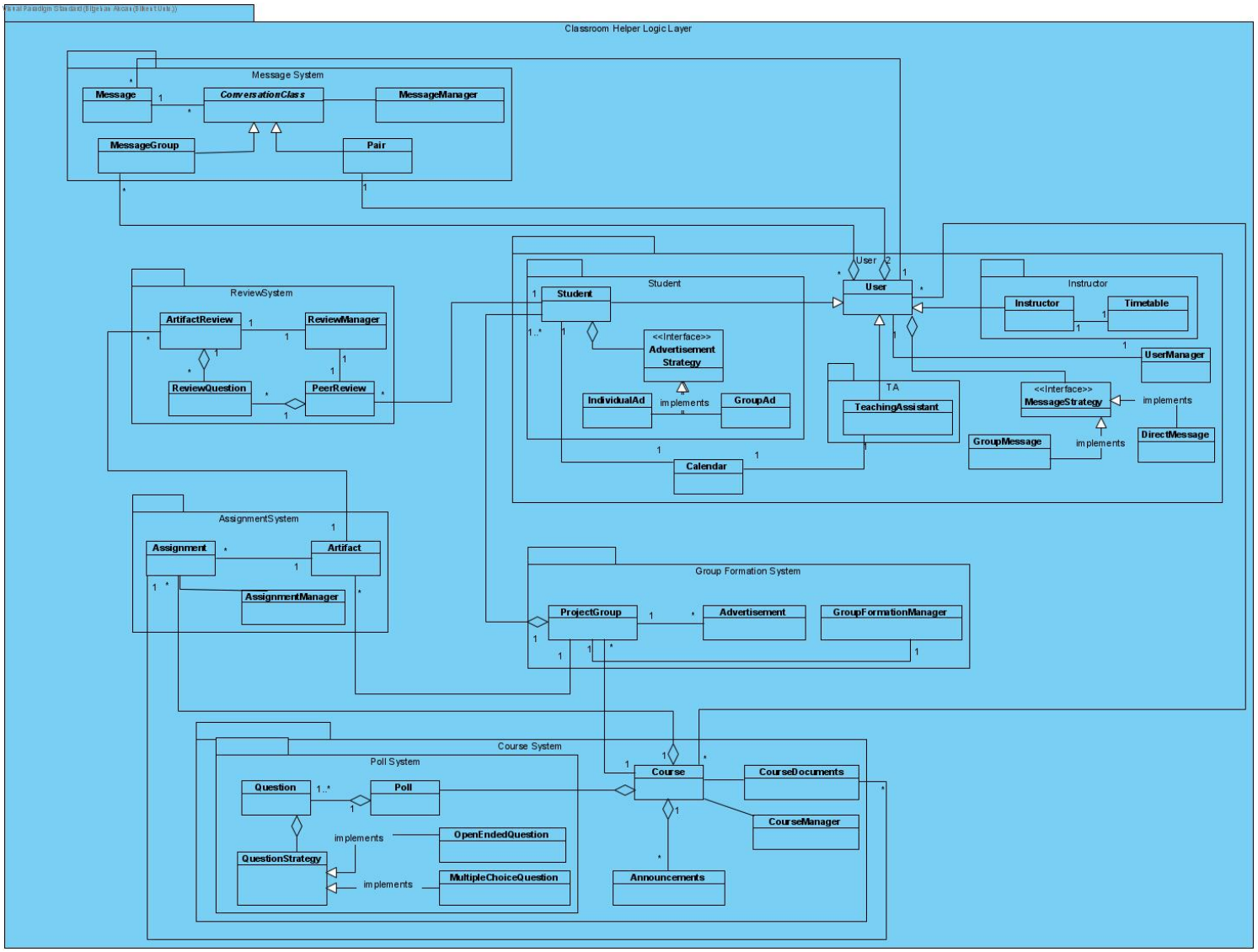
Help Class

This class represents the help page when users will use for necessary system information and they will be forwarded there through the menu.

Settings Class

This class represents the settings page when users need to modify their system choices and they will be forwarded there through the menu.

3.3 Classroom Helper Logic Layer


Classroom Helper Logic Layer
Message System
Message
ConversationClass
MessageManager
MessageGroup
Pair
ReviewSystem
ArtifactReview
ReviewManager
ReviewQuestion
PeerReview
Student
Student
<<Interface>>
Advertisement Strategy
IndividualAd
implements
GroupAd
User
User
Instructor
Instructor
Timetable
UserManager
TA
TeachingAssistant
<<Interface>>
MessageStrategy
implements
GroupMessage
implements
DirectMessage
Calendar
AssignmentSystem
Assignment
Artifact
AssignmentManager
Group Formation System
ProjectGroup
Advertisement
GroupFormationManager
Course System
Poll System
Question
Poll
implements
OpenEndedQuestion
QuestionStrategy
implements
MultipleChoiceQuestion
Course
CourseDocuments
CourseManager
Announcements

3.3.1.User Subsystem

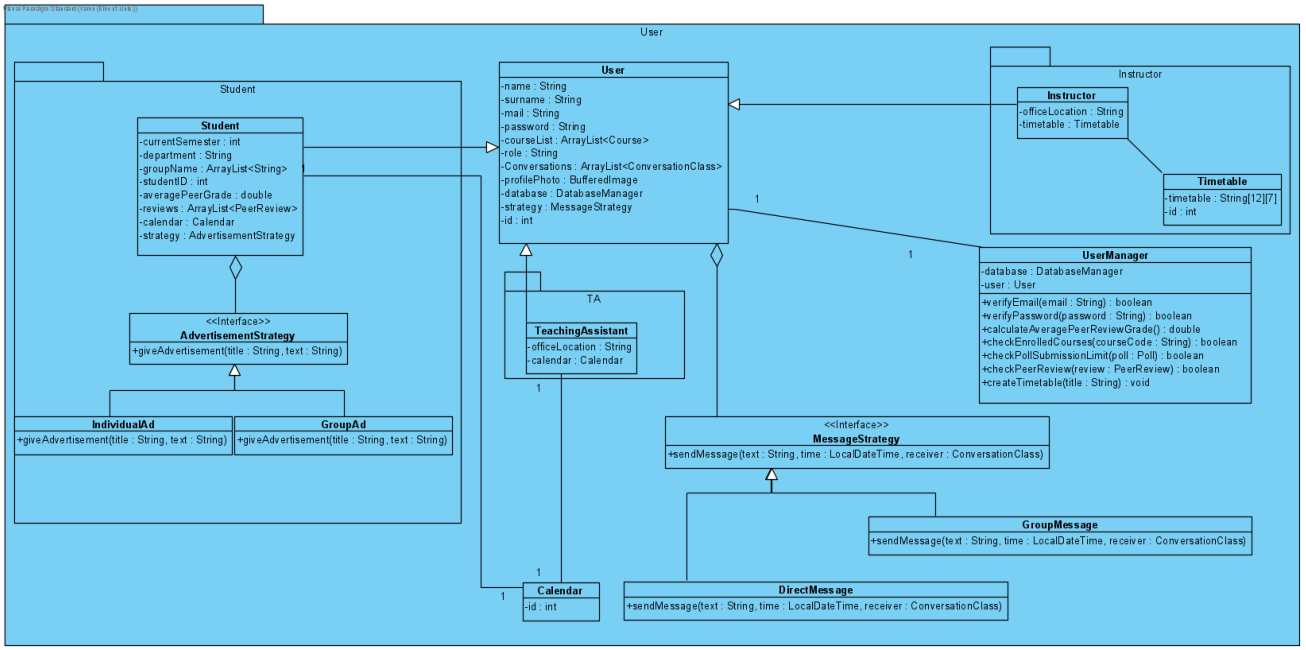


Figure 5

User Class

This class represents and defines the features and actions of all the user options in the Catch Up. All users in the system have name, surname, mail address, password and a list of courses.

Attributes

- private String name: Name of the user
- private String surname: Surname of the user
- private String mail: Mail address of the user
- private String password: Password of the user
- private int id: Id of the user
- private ArrayList<Course> courseList: Course list of the user
- private String role: Role of the user in system
- private ArrayList<ConversationClass> conversations: Existing conversations of the user

- private BufferedImage profilePhoto: Profile photo of the user
- private DatabaseManager database: Manager for user to communicate with database
- private MessageStrategy strategy: Message strategy instance to apply strategy design pattern

Operations

- public String getName(): Gets the user name
- public void setName(String name): Sets the user name
- public String getSurname(): Gets the user surname
- public void setSurname(String surname): Sets the user surname
- public int getId(): Gets the id of user
- public void setId(int id): Sets the id of the user
- public String getMail(): Gets the user mail address
- public void setMail(String mail): Sets the user mail address
- public String getPassword(): Gets the user password
- public void setPassword(String password): Sets the user password
- public ArrayList<Course> getCourseList(): Gets the user course list
- public void setCourseList(ArrayList<Course> courseList): Sets the user course list
- public String getRole(): Gets the role of the user
- public void setRole(String role): Sets the user role
- public ArrayList<ConversationClass> getConversations(); Gets the conversations of the user
- public BufferedImage getPhoto(): Gets the profile photo of the user

- public void setPhoto(BufferedImage profilePhoto): Sets the profile photo of the user
- public boolean gradeAssignment(ArtifactReview artifactReview, double grade): Gives the given grade to the given artifact
- public boolean giveFeedback(ArtifactReview artifactReview, String feedback): Gives the given feedback to the given artifact
- public void editProfile(String changedItem, String input): Edits the selected profile part
- public boolean sendMessage(String text, LocalDateTime time, ConversationClass receiver): Creates and sends a message to a given receiver
- public boolean addCourse(Course course): Adds the given course to the user's course list
- public boolean removeCourse(String courseCode): Removes the selected course from user's course list.
- public boolean createMessageGroup(int memnum, String name, ArrayList<User> groupMembers): Creates message group
- public boolean createDM(User user1): Creates pair for direct messages
- public void addPhoto(String path): Selects the photo from the computer to be assigned as profile photo
- public void editFeedback(ArtifactReview artifact, String feedback): Edits the already given feedback to an artifact
- public void editGrade(ArtifactReview artifact, double grade): Edits the already given grade to an artifact

Instructor Class

This class is a subclass of the user class. Besides the user features, instructors have their office location and time table information in the system.

Attributes

- private String officeLocation: Location of the office of instructor
- private TimeTable timeTable: Time table of instructor

Operations

- public String getOfficeLocation(): Gets the office location
- public void setOfficeLocation(String officeLocation): Sets the office location
- public TimeTable getTimeTable(): Gets time table of instructor
- public void setTimeTable(TimeTable timeTable): Sets time table of instructor
- public String createCourse(String courseName): Creates a new course
- public boolean assignAssignments(CourseDocuments document, Date duedate, String title, String description): Assigns a new assignment for course
- public boolean editAssignment(Assignment assignment, String changedItem, String input): Edits the previously created assignment
- public boolean shareCourseDocument(Course course, CourseDocuments document): Shares a course document
- public CourseDocument scanDesktop(): Scans the desktop to upload the selected course document
- public Poll createPoll(String name, int questionNum, Date dueadate): Creates a poll

- public void seeAssignmentReport(Assignment assignment): Shows assignment statistic report
- public void assignDueDateToAssistant(TeachingAssistant assistant, Assignment assignment, Date duedate): Assign due date to assistant
- public Question createQuestion(Poll poll, String question, String questionType): Creates a question for poll
- public void addAnswerToQuestion(Question question, String answerText): Adds answer option to multiple choice question
- public boolean createAnnouncement(Course course, String title, String description, Date date): Creates announcement for the course
- public void seePollReport(Poll poll): Shows the statistics of poll results
- public void seePeerReview(ProjectGroup group): Shows peer reviews that team members given to each other
- public void seeDeadlineExtension(Assignment assignment): Shows deadline extension requests for given assignment
- public void removeAssignment(Assignment assignment): Removes the given assignment
- public void removePoll(Poll poll): Removes the given poll
- public void removeQuestion(Question question): Removes the given question
- public void editTimeTable(String date, String time, String title, String editType): Edits the time table of instructor

Timetable Class

This class is for instructors to show their time table at the system so that students can see the empty time slots of the instructor in order to ask them for an appointment.

Attributes

- private String Timetable[12][7]: Multidimensional array that holds the timetable of the instructor
- private int id: Id of the timetable

Operations

- public void setTimeTable(String[12][7] timeTable):Sets the timetable
- public String[12][7] getTimeTable(); Gets the timetable
- public int getId(): Gets the id of the timetable
- public void setId(int id): Sets the id of the timetable
- public boolean addActivity(int row,int col, String activity): Adds activity to the timetable
- public boolean removeActivity(int row, int col): Removes the activity from the timetable

Teaching Assistant Class

This class is a subclass of the user class. Besides the user features, teaching assistants have their office location information in the system.

Attributes

- private String officeLocation: Office Location of the assistant
- private Calendar calendar: Calendar of the assistant

Operations

- public String getOfficeLocation(): Gets the office location of the TA
- public void setOfficeLocation(String office):Sets the office location of the TA
- public Calendar getCalendar(): Gets the calendar of TA
- public void setCalendar(Calendar calendar): Sets the calendar of TA

- public boolean enrollCourse(String courseCode): Enrolls the course by codes
- public boolean assignAssignments(CourseDocument document, Date duedate, String title, String description): Assigns assignments
- public void editAssignment(Assignment assignment, String changedItem, String input): Edits assignments
- public boolean formGroup(ArrayList<int> ids, String name, int memberNum , int maxMemNum , Course course): Forms group
- public boolean deleteGroup(ProjectGroup group): Deletes the group
- public CourseDocument scanDesktop(): Scans the desktop to upload the selected course document
- public void removeAssignment(Assignment assignment): Removes the given assignment

Student Class

This class is a subclass of the user class. Besides the user features, students have the information of their current semester, group name, id, department and peer grade average in the system.

Attributes

- private int currentSemester: Current semester of the student
- private String groupName: Name of the student's group
- private int studentID: ID of the student
- private String department: Department of the student
- private double averagePeerGrade: Average of the received peer grades of the student

- private ArrayList<PeerReview> reviews: All the peer reviews of student
- private Calendar calendar: Calendar of the student
- private AdvertisementStrategy strategy: Advertisement strategy instance to apply strategy design pattern

Operations

- public int getCurrentSemester(): Gets the current semester of student
- public void setCurrentSemester(int currentSemester): Sets the current semester of the student
- public String getGroupName(): Gets the group name of student
- public void setGroupName(String groupName): Sets the group name of student
- public int getStudentID(): Gets the student ID
- public void setStudentID(int studentID): Sets the student ID
- public String getDepartment(): Gets the student department
- public void setDepartment(String department): Sets the student department
- public double getAveragePeerGrade(): Gets the average peer grade of the student
- public void setAveragePeerGrade(double grade): Sets the average peer grade of the student
- public ArrayList<double> getPeerGrades(): Gets the all peer grades of the student
- public ArrayList<String> getPeerFeedbacks(): Gets the all peer feedbacks of the student

- public Calendar getCalendar(): Gets the calendar of student
- public void setCalendar(Calendar calendar): Sets the calendar of the student
- public boolean givePeerGrade(Student student, double grade): Gives grade to the selected student
- public boolean givePeerFeedback(Student student, String feedback) : Gives feedback to the selected student
- public boolean enrollCourse(String courseCode): Enrolls the student to the course with given course code
- public boolean uploadAssignment(Assignment assignment,Artifact artifact): Uploads the project artifact to the assignment
- public void requestFormGroup(String assistantMail, String text): Sends a group formation request to the assistant
- public boolean giveAdvertisement(String title, String text): Gives advertise in order to find group
- public boolean removeAdvertisement(int id): Removes the given advertise
- public boolean submitPoll(Poll poll, ArrayList<String> text): Submits the answered pole
- public boolean addMeetingTime(String title, Date time): Adds a meeting to the given day
- public boolean removeMeeting(String title): Removes the added meeting
- public Artifact scanDesktop(): Scans the desktop to upload the selected artifact

- public boolean hasReviewed(Student student): Checks if this student reviewed given student or not
- public void editAssignment(Assignment assignment, Artifact artifact): Edits the given artifact for the given assignment
- public boolean addToCalendar(Date date, String text): Adds the given date to calendar

AdvertisementStrategy Interface

This class is an interface class to apply strategy pattern for Student class.

Operations

- public boolean giveAdvertisement(String title, String text)

IndividualAd Class

This class is for giving advertisement for students individually

Operations

- public boolean giveAdvertisement(String title, String text)

GroupAd Class

This class is for giving advertisement for students for the name of their group

Operations

- public boolean giveAdvertisement(String title, String text)

Calendar Class

This class is for students and teaching assistants to see the upcoming assignments or meetings on a calendar with due dates. We will extend java.util.calendar class for it.

Attributes

- private int id: Id of the calendar

Operations

- public boolean addActivity(Date date, Time time, String title): Adds activity to the calendar
- public boolean removeActivity(Date date): Removes activity from the calendar
- public int getId(): Gets the id of the calendar
- public void setId(int id): Sets the id of the calendar
- public ArrayList<String> sendNotificationInfo(): Sends the activity information to the notifications

MessageStrategy Interface Class

This class is an interface class to apply strategy design pattern for User class

Operations

- public boolean sendMessage(String text, LocalDateTime time, ConversationClass receiver)

GroupMessage Class

This class is for sending a message to a message group

Operations

- public boolean sendMessage(String text, LocalDateTime time, ConversationClass receiver)

DirectMessage Class

This class is for sending a direct message to one person

Operations

- public boolean sendMessage(String text, LocalDateTime time, ConversationClass receiver)

UserManager Class

Attributes

- private DatabaseManager database: Manager that provides database communication
- private User user: User object to be managed

Operations

- public boolean verifyEmail(String email): Verifies if the given email already exists in the system or not
- public boolean verifyPassword(String password): Verifies if the given password is correct for the account
- public double calculateAveragePeerReviewGrade(): Calculates the average of the peer review grades of a student
- public boolean checkEnrolledCourses(String courseCode): Checks if the student already enrolled in the course with given course code
- public boolean checkPollSubmissionLimit(Poll poll): Checks the poll submission limit given for students to answer the poll
- public boolean checkPeerReviewLimit(PeerReview review): Checks the peer review limit given for students to review their teammates

3.3.2 Course SubSystem

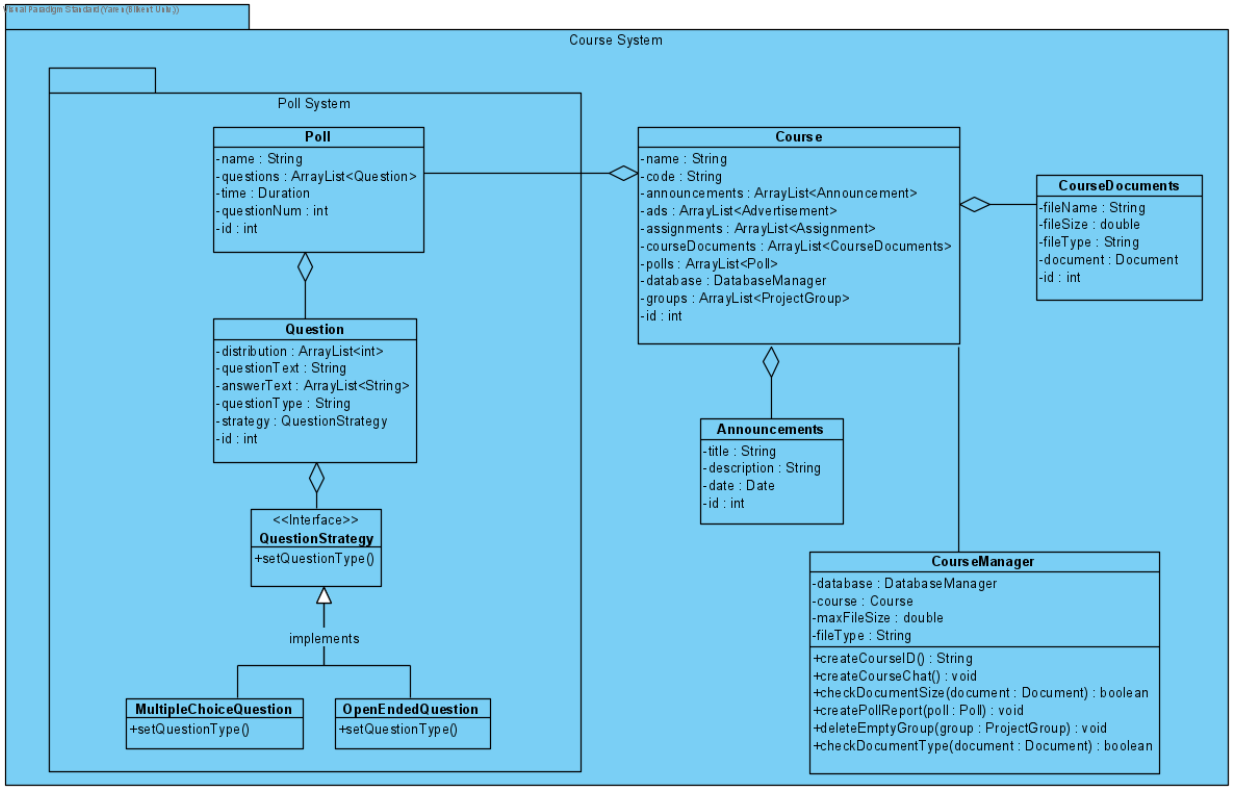


Figure 6

Course Class

All users in the system can have many courses. These courses have a name, a unique code and a course ID. Also, there are many assignments, announcements, advertisements, course documents, polls, recorded project groups in every course.

Attributes

- private String name: Name of the course
- private String code: Code of the course
- private int courseID: ID of the course
- private ArrayList<Announcement> announcements: Announcements that are done in the course.

- private ArrayList<Advertisement> ads: Advertisements that are given for that course.
- private ArrayList<Assignment> assignments: Assignments that are assigned for that course.
- private ArrayList<CourseDocuments> courseDocuments: Course documents that are uploaded for that course.
- private ArrayList<Poll> polls: Polls that are created for that course.
- private ArrayList<ProjectGroup> groups: Project Groups that are registered in the course
- private DatabaseManager database: Manager that provides database connection

Operations

- public String getName(): Gets the name of the course.
- public void setName(String name): Sets the name of the course.
- public String getCode(): Gets the code of the course.
- public void setCode(String code):Sets the code of the course.
- public int getID(): Gets the ID of the course.
- public void setID(int ID): Sets the ID of the course.
- public ArrayList<Announcements> getAnnouncements(): Gets the announcements of the course.
- public ArrayList<Advertisement> getAds(): Gets the advertisements of the course.
- public ArrayList<Assignment> getAssignments(): Gets the assignments of the course.

- public ArrayList<CourseDocuments> getCourseDocuments(): Gets the course documents of the course.
- public ArrayList<Poll> getPolls(): Gets the polls of the course.
- public boolean addAnnouncement(String title, String description, Date date): Add announcements to the course.
- public boolean addGroup(ProjectGroup group): Add group to the course.
- public ArrayList<ProjectGroup> getProjectGroups(): Gets the registered project groups in the course.

CourseDocuments Class

This class is for instructors to share project related assignments in the system. Each course document has a file name, file size, file type and documentID. It is for uploading files to the related course by all users.

Attributes

- private String fileName: Name of the course document (file)
- private double fileSize: Size of the course document (file)
- private String fileType: Type of the course document (file)
- private int documentID: ID of the course document

Operations

- public String getFileName(): Gets the name of the file.
- public void setFileName(String fileName): Sets the name of the file.
- public double getFileSize(): Gets the size of the file.
- public void setFileSize(String size): Sets the size of the file.
- public String getFileType(): Gets the type of the file.
- public void setFileType(String type): Sets the type of the file.

- public int getDocumentID(): Gets the ID of the course document.
- public void setDocumentID(int documentID): Sets the ID of the course document.

Announcements Class

This class is for instructors to make announcements to all of the course's students. Each announcement has a title, description, date and ID.

Attributes

- private String title: Title of the announcement
- private String description: Description of the announcement
- private Date date: Date of the announcement
- private int announcementID: ID of the announcement

Operations

- public String getTitle(): Gets the title of the announcement.
- public void setTitle(String title): Sets the title of the announcement.
- public String getDescription(): Gets the description of the announcement.
- public void setDescription(String description): Sets the description of the announcement.
- public Date getDate(): Gets the date of the announcement.
- public void setDate(Date announcementDate): Sets the date of the announcement.
- public int getAnnouncementID(): Gets the ID of the announcement.
- public void setAnnouncementID(int announcementID): Sets the ID of the announcement.

Poll System

Poll Class

Polls can be created by instructors for students to reflect their opinions on course subjects. Each poll has a name, questions, certain number of questions, a time duration to be answered and an ID.

Attributes

- private String name: Name of the poll
- private ArrayList<Question> questions: Questions in the poll
- private Duration time: Accessibility time of the poll
- private int questionNum: Number of the questions of the poll
- private int pollID: ID of the poll

Operations

- public Duration getTime(): Gets the accessibility time of the poll.
- public void setTime(Duration time): Sets the accessibility time of the poll.
- public String getName(): Gets the name of the poll.
- public void setName(String name): Sets the name of the poll.
- public ArrayList<Question> getQuestions(): Gets the questions in the poll.
- public void addQuestion(): Adds questions to the question ArrayList of Poll class.
- public int getQuestionNum(): Gets the question number in the poll.
- public void setQuestionNum(int num): Sets the question number in the poll.
- public int getPollID(): Gets the ID of the poll
- public void setPollID(int pollID): Sets the ID of the poll

Question Class

This class represents poll questions. Every question has a question type, question text to ask the question, list of answer text to demonstrate different answer options, list of distribution for each answer option and an ID.

Attributes

- private ArrayList<int> distribution: Distribution of the votes for each question
- private String questionText: Text of the question
- private ArrayList<String> answerText: Text of the answer
- private String questionType: Type of the question
- private QuestionStrategy strategy: Instance of the QuestionStrategy interface used for implementing strategy design pattern
- private int questionID: ID of the question

Operations

- public String getQuestionText(): Gets the text of the question.
- public void setQuestionText(String text): Sets the text of the question.
- public String getAnswer(): Gets the answer of the question.
- public void setAnswer(String answer): Sets the answer of the question.
- public ArrayList<int> getDistribution(): Gets the distribution
- public void setDistribution(ArrayList<int> distr): Sets the distribution.
- public ArrayList<double> calculateDistributionPercentage(): Calculates the distribution percentage.
- public String getQuestionType(): Gets the type of the question.
- public void setQuestionType(String questionType): Sets the type of the question.
- public int getQuestionID(): Gets the ID of the question
- public void setQuestionID(int questionID): Sets the ID of the question.

QuestionStrategy Interface

This interface is used to implement the strategy design pattern which helps us to select question type in runtime.

- public void setQuestionType(String questionType)

MultipleChoiceQuestion Class

This class is for setting the poll question type to multiple choice.

- public void setQuestionType(String questionType): Sets the question types to multiple choice questions.

OpenEndedQuestion Class

This class is for setting the poll question type to open ended.

- public void setQuestionType(String questionType): Sets the question types to open ended questions.

CourseManager Class

This class basically manages the course class operations and helps to support the system's nonfunctional requirements related to the course.

Attributes

- private DatabaseManager database: Manager that provides database communication
- private Course course: Course object to be managed
- private double maxFileSize: Maximum file size that can be accepted
- private String fileType: File type that can be accepted

Operations

- public String createCourseID(): Creates course ID.
- public void createCourseChat(): Creates course chat window.

- public boolean checkDocumentSize(Document document): Controls the size of the document that will be uploaded.
- public void createPollReport(Poll poll): Creates the poll report.
- public void deleteEmptyGroup(ProjectGroup group): Deletes the project group with no member.
- public boolean checkDocumentType(Document document): Controls the file type of the document that will be uploaded.

3.3.3 Message System

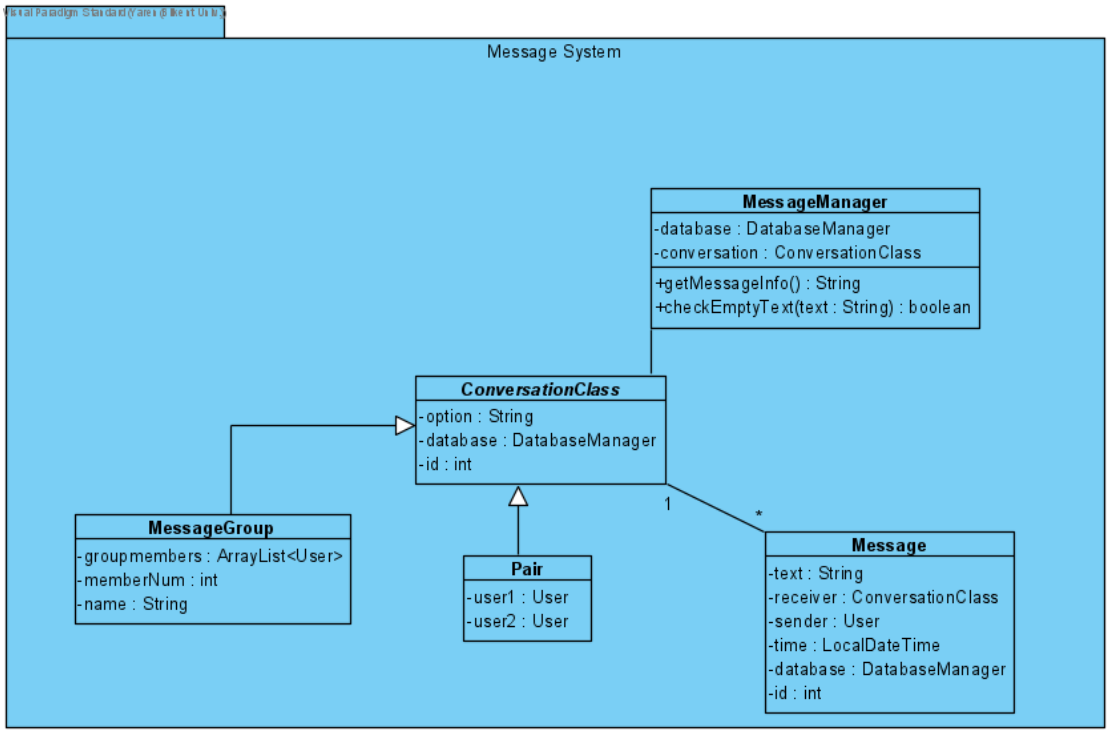


Figure 6

Message Class

This class is for messaging in the system. Messages have a sender, a receiver, time, a text that includes the message information and ID.

Attributes

- private String text: Text of the message
- private ConversationClass receiver: Receiver of the message
- private User sender: User that send the message
- private LocalDateTime time: Time of the message that has been sent
- private DatabaseManager database: Manager that provides database communication
- private int messageID: ID of the message

Operations

- public String getText(): Gets the message text
- public void setText(String text): Sets the message text
- public User getReceiver(): Gets the receiver
- public void setReceiver(ConversationClass receiver): Sets the receiver
- public User getSender(): Gets the sender of the message
- public void setSender(User sender): Sets the sender of the message
- public LocalDateTime getTime(): Gets the time of the message
- public void setTime(LocalDateTime time): Sets the time of the message
- public int getMessageID(): Gets the ID of the message
- public void setMessageID(int messageID): Sets the ID of the message

Pair Class

This class is the subclass of conversation class. It is for private messages between two users. Thus, every pair has user1 and user2.

Attributes

- private User user1: one of the users
- private User user2: another user

Operations

- public User getUser1(): Gets the user1.
- public void setUser1(User user1): Sets the user1.
- public User getUser2(): Gets the user2.
- public void setUser2(User user2): Sets the user2.

MessageGroup Class

This class is the subclass of conversation class. It is for group messages in the system. Message groups consist of one or more users. Every message group has a name, number of members and list that includes the users in the group.

Attributes

- private ArrayList<User> groupMembers: Array list that keeps the group members
- private int memberNum: Number of the members in the group
- private String name: Name of the groups

Operations

- public ArrayList<User> getGroupMembers(): Gets the group members
- public void setGroupMembers(ArrayList<User> groupMembers): Sets the group members
- public int getMemberNum(): Returns the number of members in the group
- public void setMemberNum(int memberNum): Sets the number of members in the group
- public String getName(): Gets the name of the group
- public void setName(String name): Sets the name of the group
- public boolean addMember(Student student): Adds the member to the message group

- public boolean removeMember(int studentId): Removes the one of the member with given ID.

ConversationClass Class

This class is for representing people that take roles in messaging. Each conversation class can have one or more messages.

Attributes

- private String option: Shows whether Conversation Class is pair or message group
- private DatabaseManager database: Manager that provides database connection
- private int conversationID: ID of the conversation

Operations

- public void setOption(String option): Sets the option of conversation class.
- public String getOption():Gets the option of conversation class.
- public int getConversationID(): Gets the conversation ID.
- public void setConversationID(int conversationID): Sets the conversationID

MessageManager Class

This class basically manages the conversation class operations and helps to support the system's nonfunctional requirements related to the conversational class.

Attributes

- private DatabaseManager database: Manager that provides database connection

- private ConversationClass conversation: ConversationClass object to be managed

Operations

- public String getMessageInfo(): Gets the previous messages.
- public boolean checkEmptyText(String text): Controls the text message not to be empty.

3.3.4 Review Subsystem

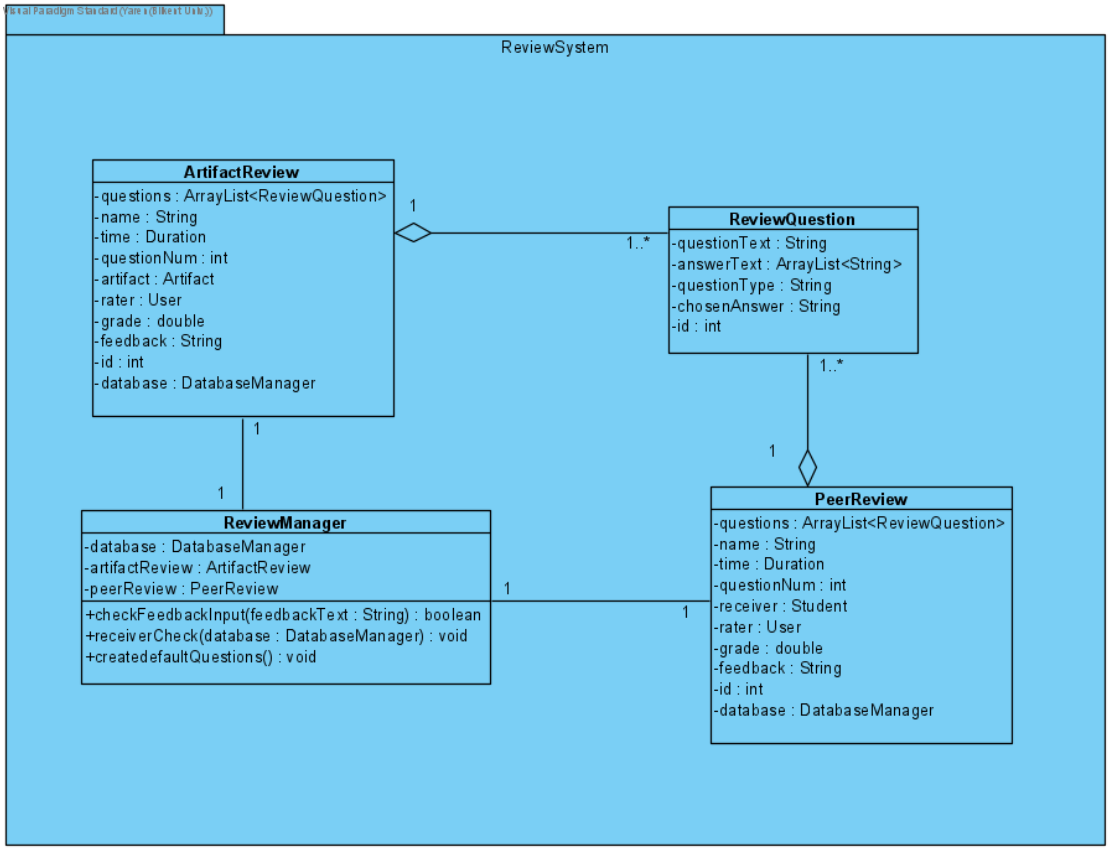


Figure 7

ReviewManager Class

This class controls the operations on the artifact review and peer review objects. It has a database object to control operations on these review objects.

Attributes

- private DatabaseManager database: Database object for connection.

- private ArtifactReview artifactReview: Artifact review object to operate on.
- private PeerReview peerReview: Peer review object to operate on.

Operations

- public void checkFeedbackInput():Checks if the feedback is given before grade is given.
- public void receiverCheck(DatabaseManager db):Checks if the receiver object of reviews exists.
- public void defaultQuestion():Creates default questions.

ArtifactReview Class

This class is for representing the reviews given to an artifact.

Attributes

- private int id: Id for database connection.
- private ArrayList<ReviewQuestion> questions: Review questions that users use to review the object.
- private DatabaseManager database: Database object for connection.
- private String name: Name of the object.
- private Duration time: Review duration for users to do the reviews.
- private int questionNum:Number of review questions.
- private Artifact artifact: Artifact for reviews.
- private User rater:User who reviews.
- private double grade: Given grade by reviewing.
- private String feedback:Given feedback by reviewing.

Operations

- public double getGrade(): Gets the grade attribute.

- public String getFeedback():Gets the feedback attribute.
- public void setGrade(double grade):Sets the grade attribute.
- public void setFeedback(String feedback):Sets the feedback attribute.
- public void setID(int id): Sets the id attribute.
- public int getID(): Gets the id attribute.
- public void addQuestions(ReviewQuestion question): Adds question to the question list.
- public void removeQuestions(ReviewQuestion question): Removes the question from the question list.
- public int searchQuestion(ReviewQuestion question): Finds the question and returns its index at list.
- public ArrayList<ReviewQuestion> getQuestions(): Gets the question list.
- public String getName(): Gets the name of the object.
- public void setName(String name): Sets the name of the object.
- public Duration getTime(): Gets the time attribute.
- public void setTime(Duration time):Sets the time attribute.
- public int getQuestionNum(): Gets the number of review questions.
- public void setQuestionNum(int num): Sets the number of review questions.
- public Artifact getArtifact(): Gets the artifact for reviews.
- public void setArtifact(Artifact artifact):Sets the artifact for reviews.

- public User getRater():Gets the user who reviews.
- public void setRater(User user):Sets the user who reviews.

PeerReview Class

This class is for representing the reviews given to an artifact.

Attributes

- private int id: Id for database connection.
- private ArrayList<ReviewQuestion> questions: Review questions

that users use to review the object.

- private DatabaseManager database: Database object for connection.
- private String name: Name of the object.
- private Duration time: Review duration for users to do the reviews.
- private int questionNum:Number of review questions.
- private Student receiver: Student for reviews.
- private User rater:User who reviews.
- private double grade: Given grade by reviewing.
- private String feedback:Given feedback by reviewing.

Operations

- public double getGrade(): Gets the grade attribute.
- public String getFeedback():Gets the feedback attribute.
- public void setGrade(double grade):Sets the grade attribute.
- public void setFeedback(String feedback):Sets the feedback attribute.
- public void setID(int id): Sets the id attribute.
- public int getID(): Gets the id attribute.
- public void addQuestions(ReviewQuestion question): Adds question

to the question list.

- public void removeQuestions(ReviewQuestion question): Removes the question from the question list.
- public int searchQuestion(ReviewQuestion question): Finds the question and returns its index at list.
- public ArrayList<ReviewQuestion> getQuestions(): Gets the question list.
- public String getName(): Gets the name of the object.
- public void setName(String name): Sets the name of the object.
- public Duration getTime(): Gets the time attribute.
- public void setTime(Duration time):Sets the time attribute.
- public int getQuestionNum(): Gets the number of review questions.
- public void setQuestionNum(int num): Sets the number of review questions.
- public Artifact getReceiver(): Gets the receiver for reviews.
- public void setReceiver(Artifact artifact):Sets the receiver for reviews.
- public User getRater():Gets the user who reviews.
- public void setRater(User user):Sets the user who reviews.

ReviewQuestion Class

This class represents review questions. Every question has a question type, question text to ask the question, list of answer text to demonstrate different answer options and an ID.

Attributes

- private String questionText: Text of the question
- private ArrayList<String> answerText: Text of the answer

- private String questionType: Type of the question
- private int id: ID of the question
- private String chosenAnswer: Chosen answer of the question.

Operations

- public String getQuestionText(): Gets the text of the question.
- public void setQuestionText(String text): Sets the text of the question.
- public String getAnswer(): Gets the answer of the question.
- public void setAnswer(String answer): Sets the answer of the question.
- public String getQuestionType(): Gets the type of the question.
- public void setQuestionType(String questionType): Sets the type of the question.
- public String getQuestionID(): Gets the ID of the question
- public void setQuestionID(String questionID): Sets the ID of the question.
- public void setChosenAnswer(String answer): Sets chosen answer of the question.
- public String getChosenAnswer(): Gets chosen answer of the question.

3.3.5 Group Formation Subsystem

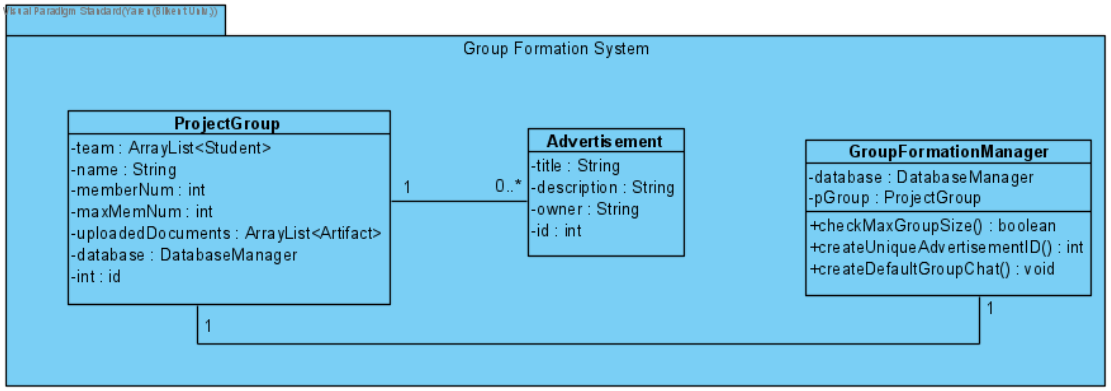


Figure 8

ProjectGroup Class

Project groups consist of students. Thus, this class has the list of students that are in the project group. Also, every project group has a name, number of current members and number of maximum members that can be in a group.

Attributes

- private String name: Name of the group
- private ArrayList<Student> team: Team members of group
- private int memberNum: Current member number of group
- private int maxMemNum: Maximum member number in group
- private ArrayList<Artifact> uploadedArtifacts: Uploaded documents of the group
- private DatabaseManager database: Manager to communicate with database
- private int id: Id of the project group

Operations

- public String getName(): Gets the group name
- public void setName(String name): Sets the group name

- public ArrayList<Student> getTeam(): Gets the team members
- public int getMemberNum(): Gets the current group size
- public void setMemberNum(int memberNum): Sets the current group size
- public int getMaxMemNum(): Gets the maximum group size
- public void setMaxMemNum(int maxMemNum): Sets the current group size
- public ArrayList<Artifact> getUploadedDocuments(): Gets the uploaded documents of the group
- public int getID(): Gets the project group id
- public void setID(int id): Sets the project group id
- public boolean removeAdvertisement(int id): Removes the given advertisement
- public void requestDeadlineExtension(Assignment assignment): Requests extension deadline for assignment
- public boolean addMember(Student student): Adds the selected student to the group
- public boolean removeMember(Student student): Removes the selected student from group
- public void addAssignment(Assignment assignment,Artifact artifact): Uploads the group assignment
- public boolean addMeetingDate(String title, Date time): Adds meeting date for a group
- public boolean removeMeeting(String title): Removes the given meeting

- public void editAssignment(Assignment assignment, Artifact artifact): Edits the given assignment

Advertisement Class

This class is for individual students and groups to give ads for a team or a team member to make group formation easier.

Attributes

- private String title: Title of the advertisement
- private String description: Description of the advertisement
- private String owner: Owner of the advertisement
- private int id: ID of the advertisement

Operations

- public String getTitle(): Gets the title of the advertisement
- public void setTitle(String title): Sets the title of the advertisement
- public String getDescription(): Gets the description of the advertisement
- public void setDescription(String description): Sets the description of the advertisement
- public String getOwner(): Gets the owner of the advertisement
- public void setOwner(String owner): Sets the owner of the advertisement
- public int getID(): Gets the ID of the advertisement
- public void setID(int id): Sets the ID of the advertisement

GroupFormationManager Class

Attributes

- private DatabaseManager database: Manager that provides database communication
- private ProjectGroup pGroup: Project group object to be managed

Operations

- public boolean checkMaxGroupSize(): Checks the maximum group size for adding new member to not exceed the upper limit
- public int createUniqueAdvertisementID(): Creates a unique if for the advertisement
- public void createDefaultGroupChat(): Creates a message group that includes members of the project group by default

3.3.6 Assignment Subsystem

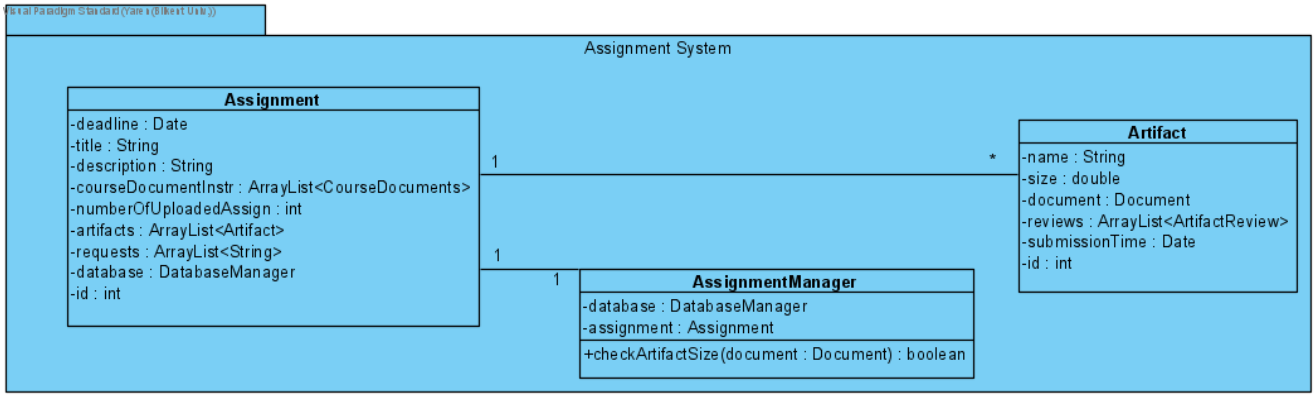


Figure 9

Assignment Class

This class is for instructors to give project related assignments such as reports to project groups and for project groups to submit the assignment. Each assignment has a title, an id, description, a deadline. Also, it has two different course documents list one for instructors to give assignment details with a document such as pdf and another list that includes each groups' submitted assignments.

Attributes

- private Date deadline: Deadline date of assignment.
- private String title: Title of assignment.
- private String description: Description of assignment.
- private ArrayList<CourseDocuments> courseDocumentsInstr: Course documents that explain assignment.
- private int numberOfUploadedAssign: Number of uploaded assignments.
- private ArrayList<Artifact> artifacts: Artifacts of assignment.
- private ArrayList<String> requests:Deadline extension requests of assignment.
- private DatabaseManager database: Database object for connection.
- private int id:Id for database connection.

Operations

- public Date getDeadline():Gets the deadline.
- public void setDeadline(Date deadline):Sets the deadline.
- public String getTitle():Gets the title.
- public void setTitle(String title):Sets the title.
- public String getDescription():Gets the description.
- public void setDescription(String description):Sets the description.
- public ArrayList<String> getRequests(): Gets the requests.
- public void addRequest(String request):Adds request.
- public void setRequest(ArrayList<String> requests):Sets the requests.
- public ArrayList<CourseDocument> getCourseDocument():Gets the course documents.
- public void addCourseDocument(CourseDocument doc):Adds course documents.

- public int getNumberOfUploadedAssign():Gets the number of assignments.
- public void setNumberOfUploadedAssign(int number):Sets the number of assignments.
- public ArrayList<Artifact> getArtifacts():Gets the artifacts.
- public void addArtifact(Artifact artifact):Adds artifact to the list.
- public void sendRequestNotification(Instructor instructor): Notifies the instructor of the course of the given assignment.
- public void setID(int id):Sets the id.
- public in getID():Gets the id.

Artifact Class

This class is for representing artifacts of the projects that are created and uploaded by students.

Attributes

- private String name:Name of artifact.
- private double size:Size of the artifact.
- private Document document:Document of artifact.
- private ArrayList<ArtifactReview> reviews: Reviews given for this artifact.
- private Date submissionTime:Submission time of the artifact.
- private int id:Id for database.

Operations

- public String getName():Gets the name.
- public void setName(String artifactName):Sets the name.
- public double getSize():Gets the size.
- public void setSize(String size):Sets the size.
- public ArrayList<ArtifactReview> getReviews():Gets the reviews.

- public void setReviews(ArrayList<ArtifactReview> reviews):Sets the reviews.
- public void addReview(ArtifactReview review):Adds review.
- public void setDocument(Document document):Sets the document.
- public Document getDocument():Gets the document.
- public String getType():Gets the type.
- public void setType(String type):Sets the type.
- public boolean giveArtifactFeedback(String feedback):Gives feedback.
- public boolean giveArtifactGrade(double grade):Gives grade.
- public int calculateStudentGradeAverage():Calculates average grade.
- public Date getSubmissionTime():Gets the submission time.
- public void setSubmissionTime():Sets the submission time.
- public int getID():Gets the id.
- public void setID():Sets the id.

AssignmentManager Class

This class manages the assignment operations.

Attributes

- private DatabaseManager database: Database object for connection.
- private Assignment assignment:Assignment object to operate on.

Operations

- public checkArtifactSize():Checks if size limit is exceeded.

Classroom Helper Data Layer

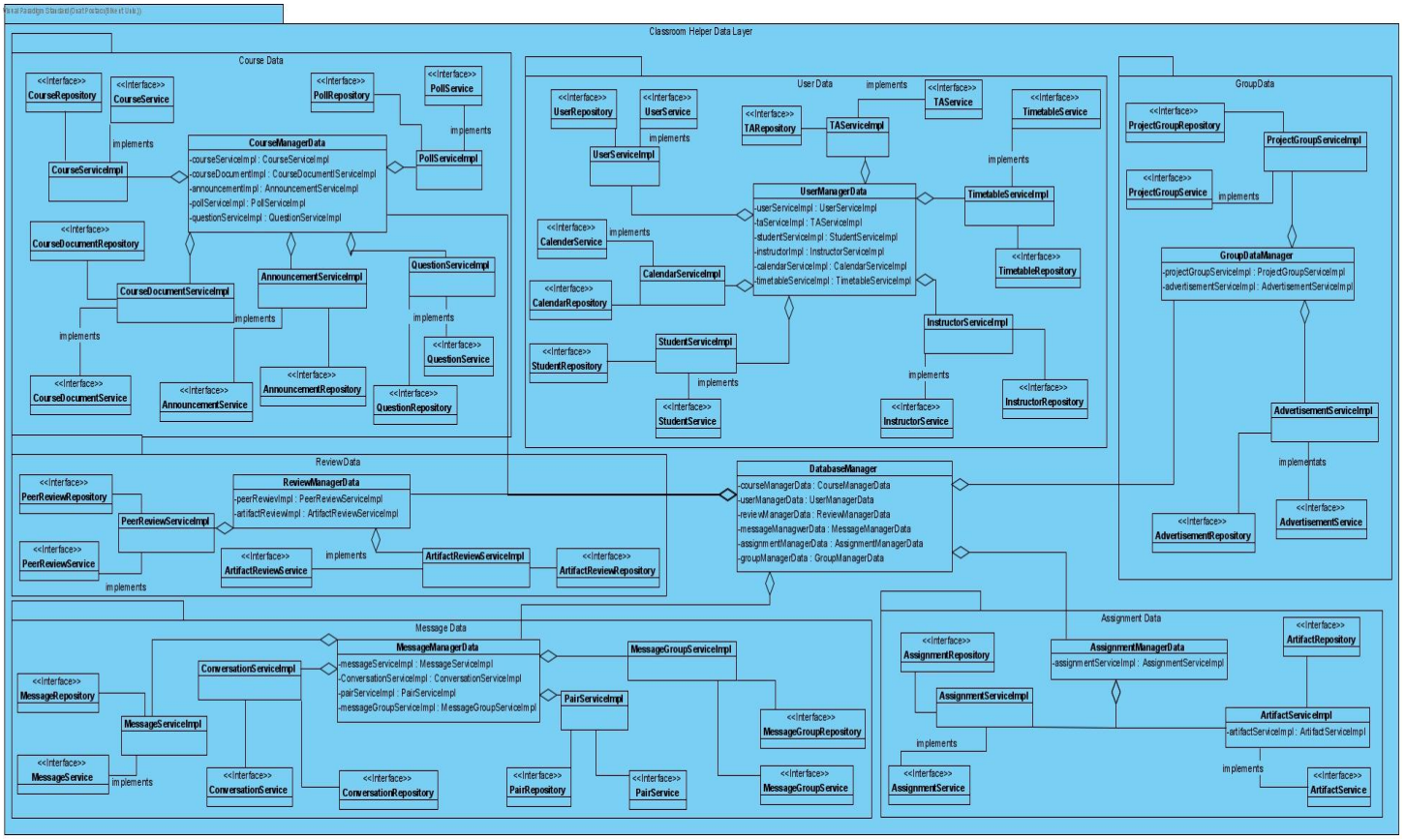
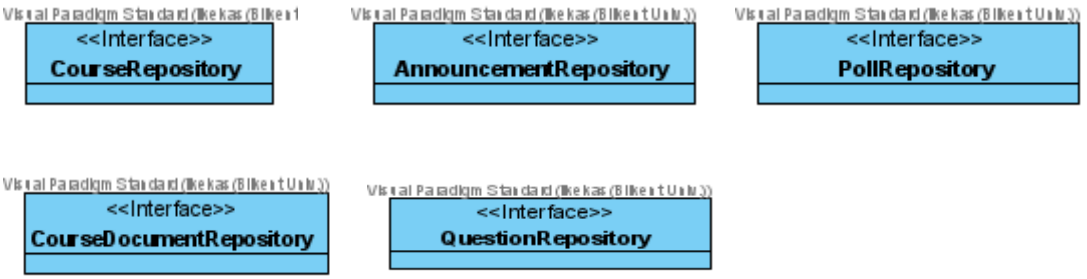


Figure 10

CourseData Subsystem

Repositories



All repositories hold each course, course document, announcement, poll, and question. to be added to the system. It will extend the JPA Repository class that has many functions that we can use for database operations.

CourseService Interface

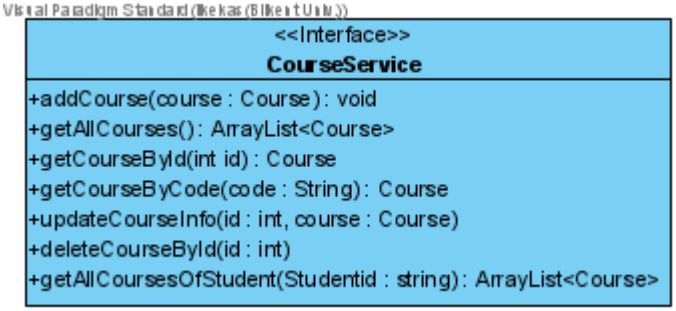


Figure 11

This course service interface will use these functions to save, get, find necessary information from the database.

- public void addCourse(Course course)
- public ArrayList<Course> getAllCourses()
- public Course getCourseById(int id)
- public Course getCourseByCode(String code)
- public void updateCourseInfo(int id, Course course)
- public void deleteCourseById(int id)
- public ArrayList<Course> getAllCoursesOfStudent(string studentId)

CourseServiceImpl Class

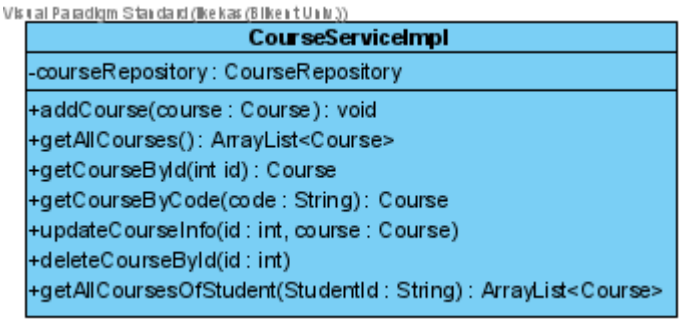


Figure 12

This class will be @service and it will implement the CourseService interface to save, get, find necessary information from the database.

- public void addCourse(Course course): This function will add course to the database system.
- public ArrayList<Course> getAllCourses(): This function will get all courses in the database system.
- public Course getCourseById(int id): This function will return the course from the database by given id.
- public Course getCourseByCode(String code): This function will return the course from the database by given course code.
- public void updateCourseInfo(int id, Course course): This function will update the course information with given id.
- public void deleteCourseById(int id): This function will delete the course with the given id from the database system.
- public ArrayList<Course> getAllCoursesOfStudent(string studentId): This function will return all the courses of the student with given id from the database system.

CourseDocumentService Interface

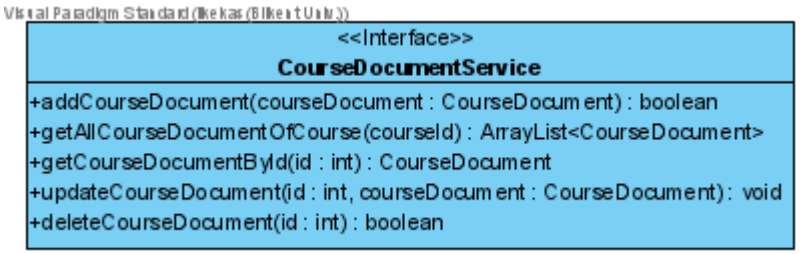


Figure 13

This course document service interface will use these functions to save, get, find necessary information from the database.

- public boolean addCourseDocument(CourseDocument courseDocument)

- public ArrayList<CourseDocument> getAllCourseDocumentOfCourse(int courseId)
- public CourseDocument getCourseDocumentById(int id)
- public void updateCourseDocument(int id, CourseDocument document)
- public void deleteCourseDocumentById(int id)

CourseDocumentImpl Class

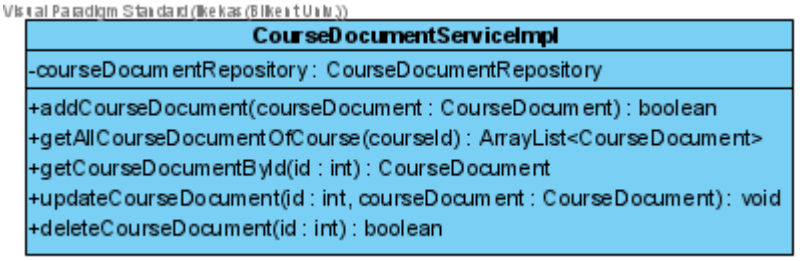


Figure 14

This class will be @service and it will implement the CourseDocumentService interface to save, get, find necessary information from the database.

- public boolean addCourseDocument(CourseDocument courseDocument): This function will add course documents to the database system.
- public ArrayList<Course> getAllCourseDocumentOfCourse(int courseId): This function will get all course documents of the course with given id from the database system.
- public CourseDocument getCourseDocumentById(int id): This function will return the course document from the database by given id.
- public void updateCourseDocument(int id, CourseDocument document): This function will update the course document with given id.
- public void deleteCourseDocument(int id): This function will delete the course document with the given id from the database system.

AnnouncementService Interface

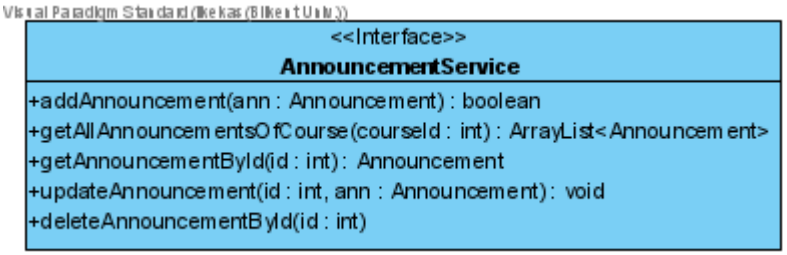


Figure 15

This announcement service interface will use these functions to save, get, find necessary information from the database.

- public boolean addAnnouncement(Announcement ann)
- public ArrayList<Announcement> getAllAnnouncementOfCourse(int courseId)
- public Announcement getAnnouncementById(int id)
- public void updateAnnouncement(int id,Announcement ann)
- public void deleteAnnouncementById(int id)

AnnouncementImpl Class

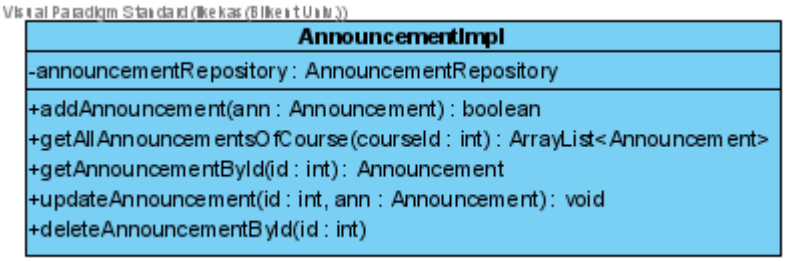


Figure 16

This class will be @service and it will implement the AnnouncementService interface to save, get, find necessary information from the database.

- public boolean addAnnouncement(Announcement annt): This function will add an announcement to the database system.

- public ArrayList<Announcement> getAllAnnouncementOfCourse(int courseId): This function will get all announcements of the course with given id from the database system.
- public Announcement getAnnouncementById(int id): This function will return the announcement from the database by given id.
- public void updateAnnouncement(int id,Announcement ann): This function will update the announcement with given id.
- public void deleteAnnouncementById(int id): This function will delete the announcement with the given id from the database system.

PollService Interface

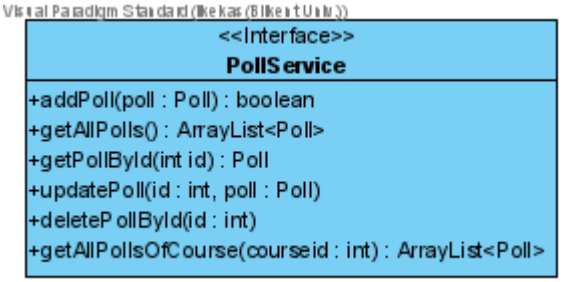


Figure 17

This poll service interface will use these functions to save, get, find necessary information from the database.

- public boolean addPoll(Poll poll)
- public ArrayList<Poll> getAllPollsOfCourse(int courseId)
- public Announcement getPollById(int id)
- public void updatePoll(int id,Poll poll)
- public void deletePollById(int id)
- public ArrayList<Poll> getAllPollsOfCourse(int courseid)

PollServiceImpl Class

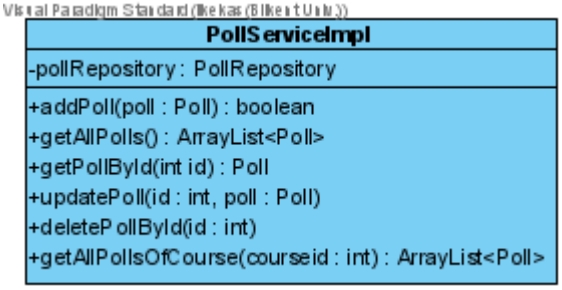


Figure 18

This class will be @service and it will implement the PollService interface to save, get, find necessary information from the database.

- public boolean addPoll(Poll poll): This function will add a poll to the database system.
- public ArrayList<Poll> getAllPollsOfCourse(int courseId): This function will get all polls of the course with given id from the database system.
- public Poll getPollById(int id): This function will return the poll from the database by given id.
- public void updatePoll(int id,Poll poll): This function will update the poll with given id.
- public void deletePollById(int id): This function will delete the poll with the given id from the database system.
- public ArrayList<Poll> getAllPollsOfCourse(int courseid): This function will get all polls of the course with given id from the database system.

QuestionService Interface

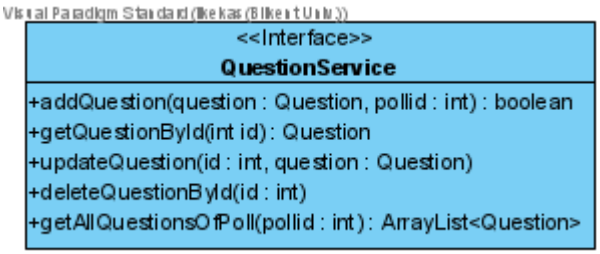


Figure 19

This question service interface will use these functions to save, get, find necessary information from the database.

- public boolean addQuestion(Question question,int pollid)
- public Announcement getQuestionById(int id)
- public void updateQuestion(int id,Question question)
- public void deleteQuestionById(int id)
- public ArrayList<Question> getAllQuestionsOfPoll(int pollid)

QuestionServiceImpl Class

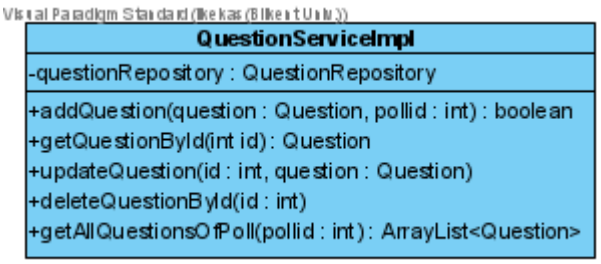


Figure 20

This class will be @service and it will implement the Question Service interface to save, get, find necessary information from the database.

- public boolean addQuestion(Question question, int pollid): This function will add a question to the poll with given id in the database system.
- public Question getQuestionById(int id): This function will return the question from the database by given id.
- public void updateQuestion(int id,Question question): This function will update the question with given id.
- public void deleteQuestionById(int id): This function will delete the question with the given id from the database system.

- public ArrayList<Poll> getAllQuestionsOfPoll(int pollid): This function will get all questions of the poll with given id from the database system.

CourseManagerData Class

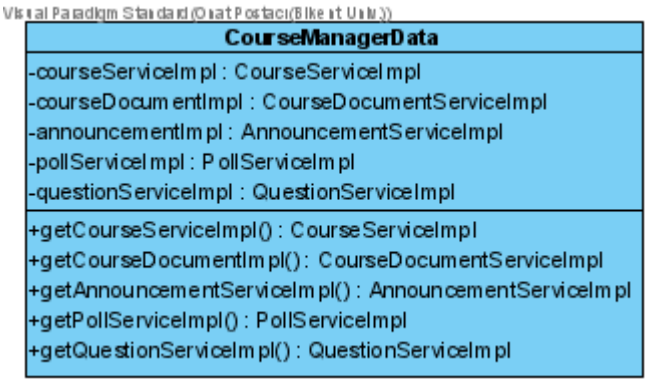


Figure 21

This class has all of the classes of the implementation of the service interfaces in this package. From this class, we can use all service implementations to manage the database of the objects of this package.

- public CourseServiceImpl getCourseServiceImpl(): This function will return the CourseServiceImpl object of the class.
- public CourseDocumentServiceImpl getCourseDocumentServiceImpl(): This function will return the CourseDocumentServiceImpl object of the class.
- public AnnouncementServiceImpl getAnnouncementServiceImpl(): This function will return the AnnouncementServiceImpl object of the class.
- public PollServiceImpl getPollServiceImpl(): This function will return the PollServiceImpl object of the class.
- public QuestionServiceImpl getQuestionServiceImpl(): This function will return the QuestionServiceImpl object of the class.

UserData Subsystem

Repositories

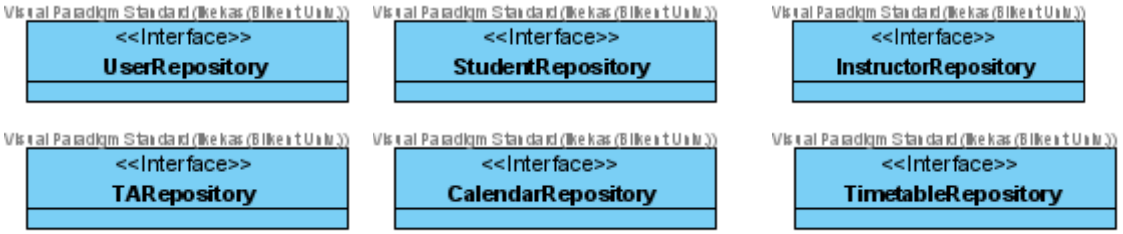


Figure 22

This repository holds each user, student, instructor, TA, calendar, and timetable to be added to the system. It will extend the JPA Repository class that has many functions that we can use for database operations.

UserService Interface

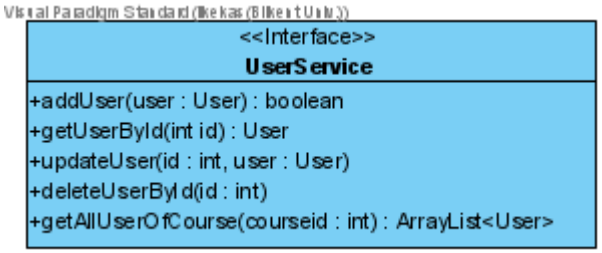


Figure 23

This user service interface will use these functions to save, get, find necessary information from the database.

- public boolean addUser(User user)
- public User getUserById(int id)
- public void updateUser(int id,User user)
- public void deleteUserById(int id)
- public ArrayList<User> getAllUserOfCourse(int courseid)

UserServiceImpl Class

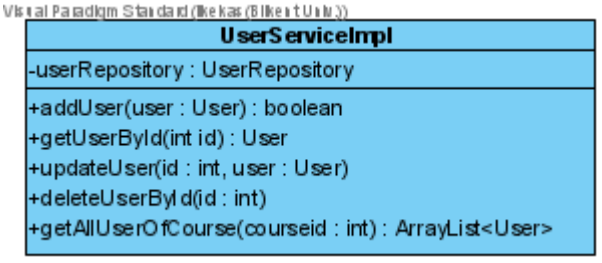


Figure 24

This class will be @service and it will implement the User Service interface to save, get, find necessary information from the database.

- public boolean addUser(User user): This function will add a user to the database system.
- public User getUserById(int id): This function will return the user from the database by given id.
- public void updateUser(int id,User user): This function will update the user with given id.
- public void deleteUserById(int id): This function will delete the user with the given id from the database system.
- public ArrayList<User> getAllUserOfCourse(int courseid): This function will get all users of the course with given id from the database system.

StudentService Interface

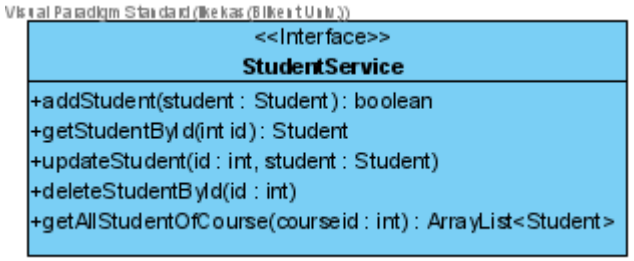


Figure 25

This student service interface will use these functions to save, get, find necessary information from the database.

- public boolean addStudent(Student student)
- public Student getStudentById(int id)
- public void updateStudent(int id,Student student)

- public void deleteStudentById(int id)
- public ArrayList<Student> getAllStudentOfCourse(int courseid)

StudentServiceImpl Class

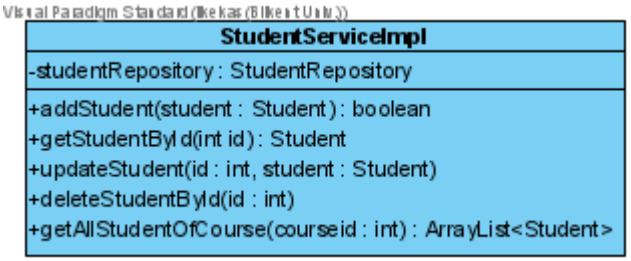


Figure 26

This class will be @service and it will implement the Student Service interface to save, get, find necessary information from the database.

- public boolean addStudent(Student student): This function will add a student to the database system.
- public Student getStudentById(int id): This function will return the student from the database by given id.
- public void updateStudent(int id,Student student): This function will update the student with given id.
- public void deleteStudentById(int id): This function will delete the student with the given id from the database system.
- public ArrayList<Student> getAllStudentsOfCourse(int courseid): This function will get all students of the course with given id from the database system.

InstructorService Interface

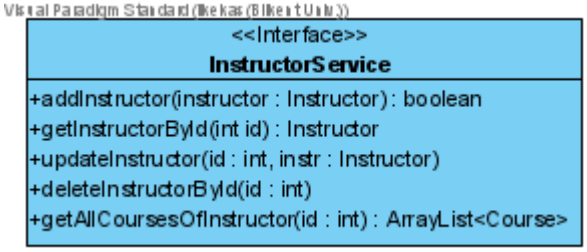
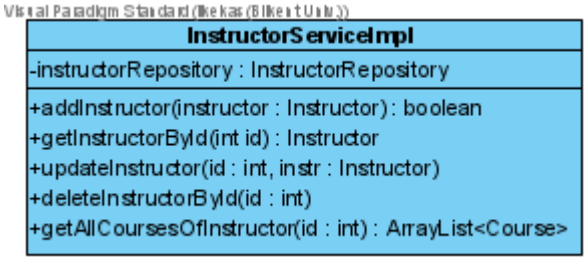


Figure 27

This instructor service interface will use these functions to save, get, find necessary information from the database.

- public boolean addInstructor(Instructor instr)
- public Instructor getInstructorById(int id)
- public void updateInstructor(int id,Instructor instructor)
- public void deleteInstructorById(int id)
- public ArrayList<Course> getAllCoursesOfInstructor(int id)

InstructorServiceImpl Class



This class will be @service and it will implement the Instructor Service interface to save, get, find necessary information from the database.

- public boolean addInstructor(Instructor instr): This function will add an instructor to the database system.
- public Instructor getStudentById(int id): This function will return the instructor from the database by given id.

- public void updateInstructor(int id,Instructor instr): This function will update the instructor with given id.
- public void deleteInstructorById(int id): This function will delete the instructor with the given id from the database system.
- public ArrayList<Course> getAllCoursessOfInstructor(int id): This function will get all courses of the instructors with given id from the database system.

TAService Interface

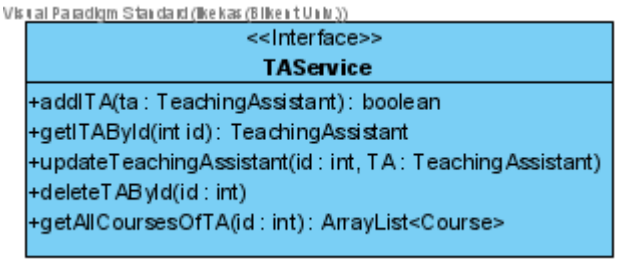


Figure 28

This TA service interface will use these functions to save, get, find necessary information from the database.

- public boolean addTA(TeachingAssistant TA)
- public TeachingAssitant getTAById(int id)
- public void updateTA(int id,TeachingAssistant TA)
- public void deleteTAById(int id)
- public ArrayList<Course> getAllCoursesOfTA(int id)

TAServiceImpl Class

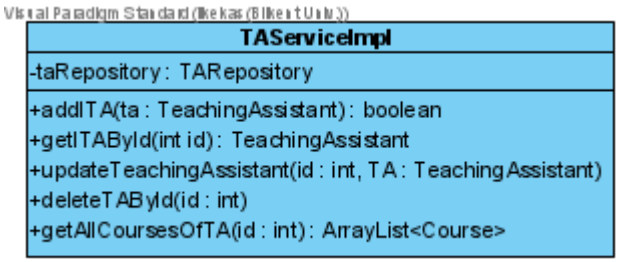


Figure 29

This class will be @service and it will implement the TA Service interface to save, get, find necessary information from the database.

- public boolean addTA(TeachingAssistant TA): This function will add an Teaching Assistant to the database system.
- public TeachingAssistant getTAById(int id): This function will return the Teaching Assistant from the database by given id.
- public void updateTA(int id,TeachingAssistant TA): This function will update the Teaching Assistant with given id.
- public void deleteTAById(int id): This function will delete the Teaching Assistant with the given id from the database system.
- public ArrayList<Course> getAllCoursessOfTA(int id): This function will get all courses of the Teaching Assistant with given id from the database system.

CalenderService Interface

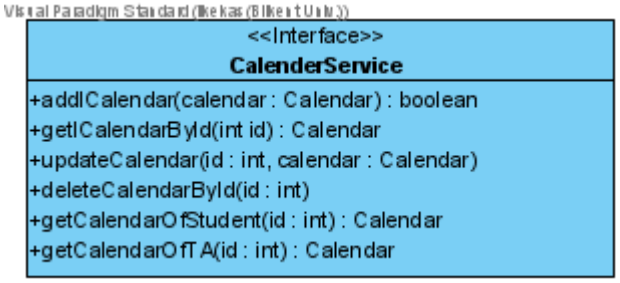


Figure 30

This Calendar service interface will use these functions to save, get, find necessary information from the database.

- public boolean addCalendar(Calendar calendar)
- public Calendar getCalendarById(int id)
- public void updateCalendar(int id,Calendar calendar)

- public void deleteCalendarById(int id)
- public Calendar getCalendarOfStudent(int id)
- public Calendar getCalendarOfTA(int id)

CalenderServiceImpl Class

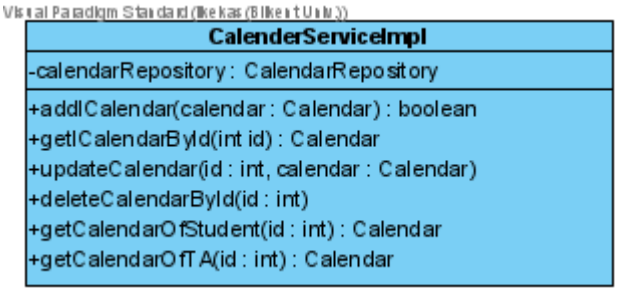


Figure 31

This class will be @service and it will implement the Calendar Service interface to save, get, find necessary information from the database.

- public boolean addCalendar(Calendar calendar): This function will add a calendar to the database system.
- public Calendar getCalendarById(int id): This function will return the calendar from the database by given id.
- public void updateCalendar (int id,Calendar calendar): This function will update the calendar with given id.
- public void deleteCalendarById(int id): This function will delete the calendar with the given id from the database system.
- public Calendar getCalendarOfStudent(int id): This function will get all calendar of the Student with given id from the database system.
- public Calendar getCalendarOfTA(int id): This function will get all calendars of the teaching assistant with given id from the database system.

TimetableService Interface

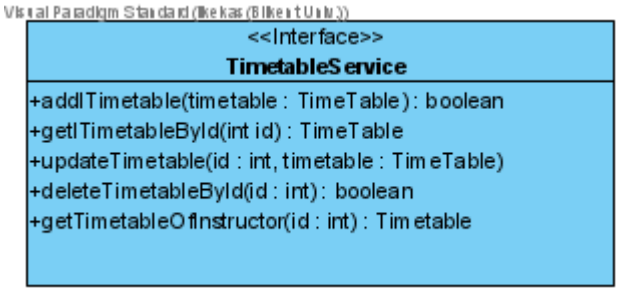
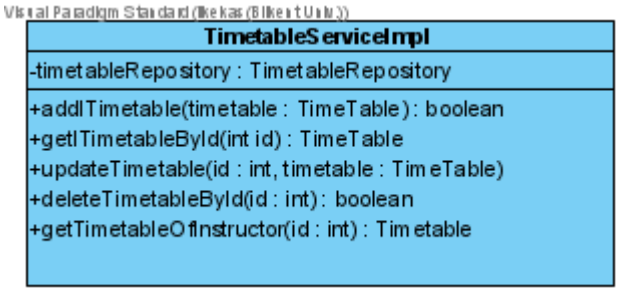


Figure 32

This Timetable service interface will use these functions to save, get, find necessary information from the database.

- public boolean addTimetable(TimeTable timetable)
- public TimeTable getTimetableById(int id)
- public void updateTimetable(int id,TimeTable timetable)
- public void deleteTimetableById(int id)
- public TimeTable getTimetableOfInstructor(int id)

TimetableServiceImpl Class

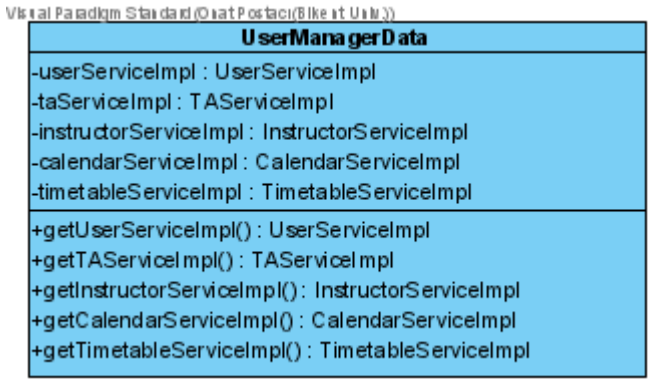


This class will be @service and it will implement the Timetable Service interface to save, get, find necessary information from the database.

- public boolean addTimetable(TimeTable timetable): This function will add a timetable to the database system.
- public TimeTable getTimetableById(int id): This function will return the timeteable from the database by given id.

- public void updateTimetable (int id,TimeTable timetabel): This function will update the timetable with given id.
- public void deleteTimetableById(int id): This function will delete the timetable with the given id from the database system.
- public TimeTable getTimetableOfInstructor(int id): This function will get the timetable of the Instructor with the given id from the database system.

UserManagerData



This class has all of the classes of the implementation of the service interfaces in this package. From this class, we can use all service implementations to manage the database of the objects of this package.

- public UserServiceImpl getUserServiceImpl(): This function will return the UserServiceImpl object of the class.
- public TAServiceImpl getTAServiceImpl(): This function will return the TAServiceImpl object of the class.
- public InstructorServiceImpl getInstructorServiceImpl(): This function will return the InstructorServiceImpl object of the class.
- public CalendarServiceImpl getCalendarServiceImpl(): This function will return the CalendarServiceImpl object of the class.

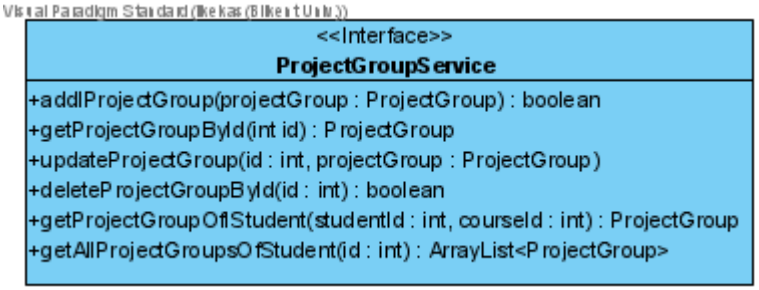
- public TimetableServiceImpl getTimetableServiceImpl(): This function will return the TimetableServiceImpl object of the class.

GroupData Subsystem

Repositories

These repositories hold each project group and advertisement to be added to the system. It will extend the JPA Repository class that has many functions that we can use for database operations.

ProjectGroupService Interface



This Project group service interface will use these functions to save, get, find necessary information from the database.

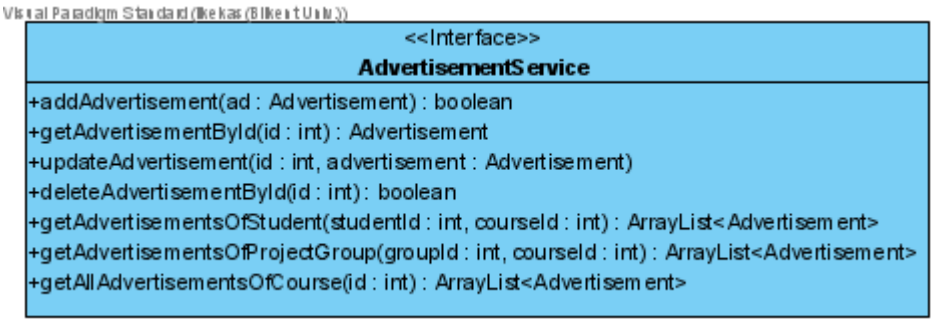
- public boolean addProjectGroup(ProjectGroup projectGroup)
- public ProjectGroup getProjectGroupById(int id)
- public void updateProjectGroup(int id,ProjectGroup projectGroup)
- public void deleteProjectGroupById(int id)
- public ProjectGroup getProjectGroupOfStudent(int studentId,int courseId)
- public ArrayList<ProjectGroup> getAllProjectGroupsOfStudent(int id)

ProjectGroupServiceImpl Class

This class will be @service and it will implement the Project Group Service interface to save, get, find necessary information from the database.

- public boolean addProjectGroup(ProjectGroup projectGroup): This function will add a project group to the database system.
- public ProjectGroup getProjectGroupById(int id): This function will return the project group from the database by given id.
- public void updateProjectGroup (int id,ProjectGroup projectGroup): This function will update the project group with given id.
- public void deleteProjectGroupById(int id): This function will delete the project group with the given id from the database system.
- public ProjectGroup getProjectGroupOfStudent(int studentId, int courseId): This function will get the project group of the Student with given id in specific course with also given id from the database system.
- public ProjectGroup getAllProjectGroupsOfStudent(int id): This function will get all project groups of the student with given id from the database system.

AdvertisementService Interface



This advertisement service interface will use these functions to save, get, find necessary information from the database.

- public boolean addAdvertisement(Advertisement advertisement)

- public Advertisement getAdvertisementById(int id)
- public void updateAdvertisement(int id,Advertisement advertisement)
- public void deleteAdvertisementById(int id)
- public ArrayList<Advertisement> getAdvertisementsOfStudent(int studentId,int courseId)
- public ArrayList<Advertisement> getAdvertisementsOfProjectGroup(int groupId,int courseId)
- public ArrayList<Advertisement> getAllAdvertisementsOfCourse(int id)

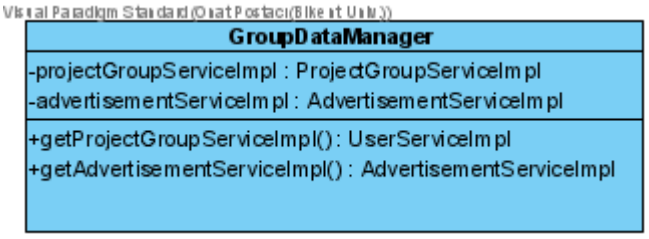
AdvertisementServiceImpl Class

This class will be @service and it will implement the Advertisement Service interface to save, get, find necessary information from the database.

- public boolean addAdvertisement(Advertisement advertisement): This function will add an advertisement to the database system.
- public Advertisement getAdvertisementById(int id): This function will return the advertisement from the database by given id.
- public void updateAdvertisement (int id,Advertisement advertisement): This function will update the advertisement with given id.
- public void deleteAdvertisementById(int id): This function will delete the advertisement with the given id from the database system.

- public ArrayList<Advertisement> getAdvertisementsOfStudent(int studentId,int courseId): This function will get the advertisements of the student with given id in the course with given id from the database.
- public Advertisement getAdvertisementsOfProjectGroup(int groupId,int courseId): This function will get the advertisements of the project group with given id in the course with given id from the database.
- public ArrayList<Advertisement> getAllAdvertisementsOfCourse(int id): This function will get all of the advertisement of the course with given id from database system.

GroupDataManager Class



This class has all of the classes of the implementation of the service interfaces in this package. From this class, we can use all service implementations to manage the database of the objects of this package.

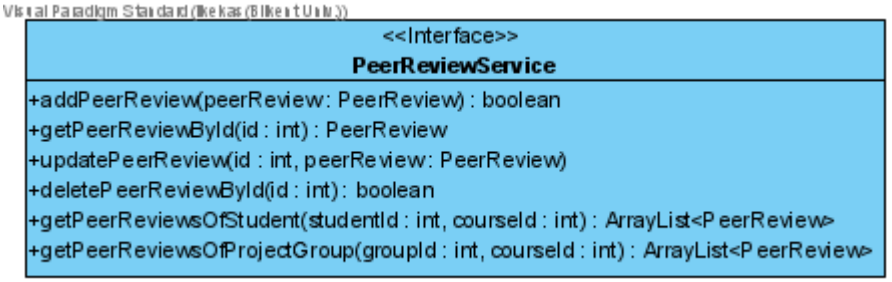
- public ProjectGroupServiceImpl getProjectGroupServiceImpl(): This function will return the GroupServiceImpl object of the class.
- public AdvertisementServiceImpl getAdvertisementServiceImpl(): This function will return the AdvertisementServiceImpl object of the class.

ReviewData Subsystem

Repositories

These repositories hold each artifact review and peer review to be added to the system. It will extend the JPA Repository class that has many functions that we can use for database operations.

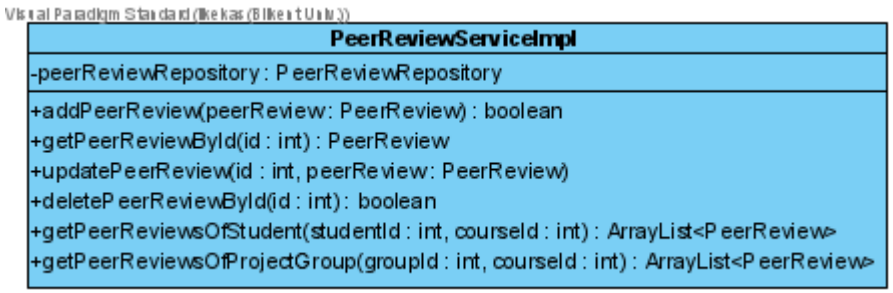
PeerReviewService Interface



This peer review service interface will use these functions to save, get, find necessary information from the database.

- public boolean addPeerReview(PeerReview peerReview)
- public PeerReview getPeerReviewById(int id)
- public void updatePeerReview(int id,PeerReview peerReview)
- public void deletePeerReviewById(int id)
- public ArrayList<PeerReview> getPeerReviewsOfStudent(int studentId,int courseId)
- public ArrayList<PeerReview> getPeerReviewsOfProjectGroup(int groupId,int courseId)

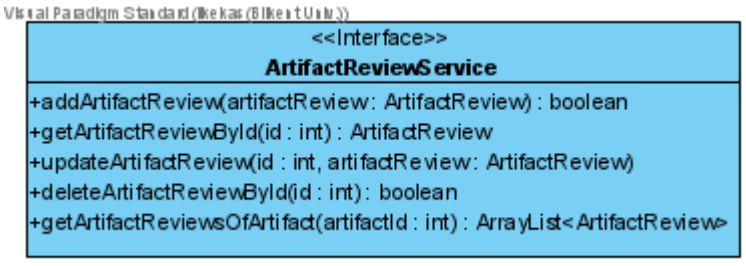
PeerReviewServiceImpl Class



This class will be @service and it will implement the peer review Service interface to save, get, find necessary information from the database.

- public boolean addPeerReview(PeerReview peerReview): This function will add a peer review to the database system.
- public PeerReview getPeerReviewById(int id): This function will return the peer review from the database by given id.
- public void updatePeerReview (int id,PeerReview peerReview): This function will update the peer review with given id.
- public void deletePeerReviewById(int id): This function will delete the peer review with the given id from the database system.
- public ArrayList<PeerReview> getAPeerReviewsOfStudent(int studentId,int courseId): This function will get the peer reviews of the student with given id in the course with given id from the database.
- public ArrayList<PeerReview> getPeerReviewsOfProjectGroup(int groupId,int courseId): This function will get the peer reviews of the project group with given id in the course with given id from the database.

ArtifactReviewService Interface



This artifact review service interface will use these functions to save, get, find necessary information from the database.

- public boolean addArtifactReview(ArtifactReview artifactReview)

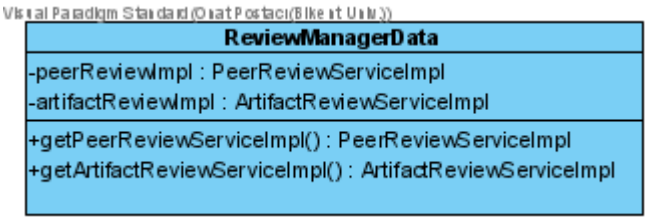
- public ArtifactReview getArtifactReviewById(int id)
- public void updateArtifactReview(int id,ArtifactReview artifactReview)
- public void deleteArtifactReviewById(int id)
- public ArrayList<ArtifactReview> getArtifactReviewsOfArtifact(int studentId,int courseId)

ArtifactReviewServiceImpl Class

This class will be @service and it will implement the artifact review Service interface to save, get, find necessary information from the database.

- public boolean addArtifactReview(ArtifactReview artifactReview): This function will add an artifact review to the database system.
- public ArtifactReview getArtifactReviewById(int id): This function will return the artifact review from the database by given id.
- public void updateArtifactReview (int id,ArtifactReview artifactReview): This function will update the artifact review with given id.
- public void deleteArtifactReviewById(int id): This function will delete the artifact review with the given id from the database system.
- public ArrayList<ArtifactReview> getArtifactReviewsOfArtifact(int artifactId): This function will get the artifact review of the student with given id in the course with given id from the database.

ReviewManagerData

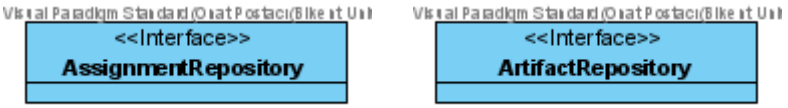


This class has all of the classes of the implementation of the service interfaces in this package. From this class, we can use all service implementations to manage the database of the objects of this package.

- public PeerReviewServiceImpl getPeerReviewServiceImpl(): This function will return the PeerReviewServiceImpl object of the class.
- public ArtifactReviewServiceImpl getArtifactReviewServiceImpl(): This function will return the ArtifactReviewServiceImpl object of the class.

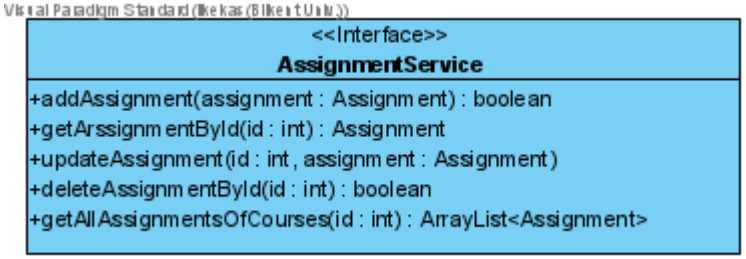
AssignmentData Subsystem

Repositories



These repositories hold each assignment and artifacts to be added to the system. It will extend the JPA Repository class that has many functions that we can use for database operations.

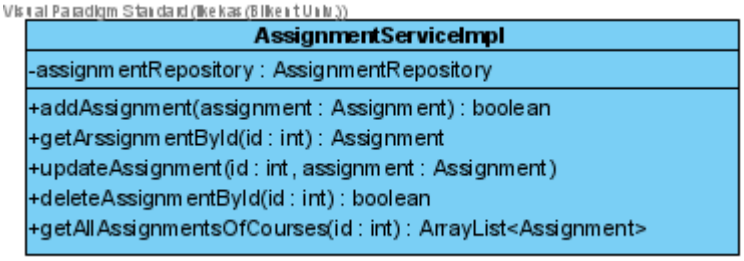
AssignmentService Interface



This assignment service interface will use these functions to save, get, find necessary information from the database.

- public boolean addAssignment(Assignment assignment)
- public Assignment getAssignment(int id)
- public void updateAssignment(int id,Assignment assignment)
- public void deleteAssignmentById(int id)
- public ArrayList<Assignment> getAssignmentsOfCourse(int id)

AssignmentServiceImpl Class

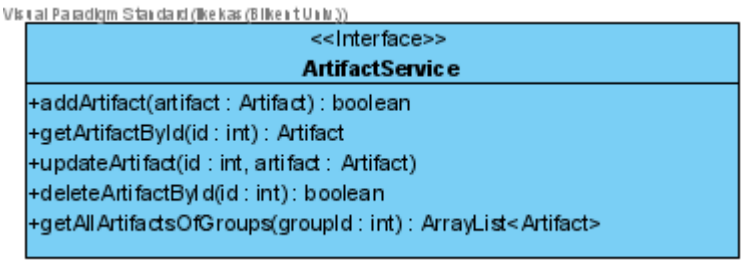


This class will be @service and it will implement the assignment Service interface to save, get, find necessary information from the database.

- public boolean addAssignment(Assignment assignment): This function will add an assignment to the database system.
- public Assignment getAssignmentById(int id): This function will return the assignment from the database by given id.
- public void updateAssignment(int id,Assignment assignment): This function will update the assignment with given id.
- public void deleteAssignmentById(int id): This function will delete the assignment with the given id from the database system.

- public ArrayList<Assignment> getAllAssignmentsOfCourses(int id): This function will get the assignments of the courses with given id in the course with given id from the database.

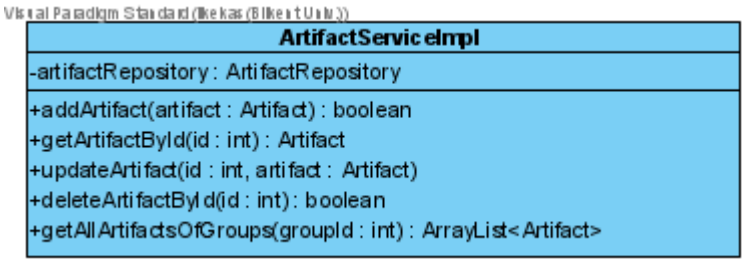
ArtifactService Interface



This artifact service interface will use these functions to save, get, find necessary information from the database.

- public boolean addArtifact(Artifact artifact)
- public Artifact getArtifactById(int id)
- public void updateArtifact(int id,Artifact artifact)
- public void deleteArtifactById(int id)
- public ArrayList<Artifact> getArtifactOfGroups(int groupId)

ArtifactServiceImpl Class

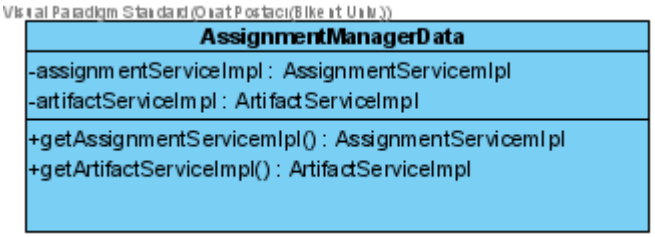


This class will be @service and it will implement the assignment Service interface to save, get, find necessary information from the database.

- public boolean addArtifact(Artifact artifact): This function will add an artifact to the database system.

- public Artifact getArtifactById(int id): This function will return the artifact from the database by given id.
- public void updateArtifact(int id,Artifact artifact): This function will update the artifact with given id.
- public void deleteArtifactById(int id): This function will delete the artifact with the given id from the database system.
- public ArrayList<Artifact> getAllArtifactsOfGroups(int id): This function will get the artifact of the courses with given id in the course with given id from the database.

ArtifactManagerData Class



This class has all of the classes of the implementation of the service interfaces in this package. From this class, we can use all service implementations to manage the database of the objects of this package.

- public AssignmentServiceImpl getAssignmentServiceImpl(): This function will return the AssignmentServiceImpl object of the class.
- public ArtifactServiceImpl getArtifactServiceImpl(): This function will return the ArtifactServiceImpl object of the class.

MessageData Subsystem

Repositories

These repositories hold each message, conversation, pair, and message group to be added to the system. It will extend the JPA Repository class that has many functions

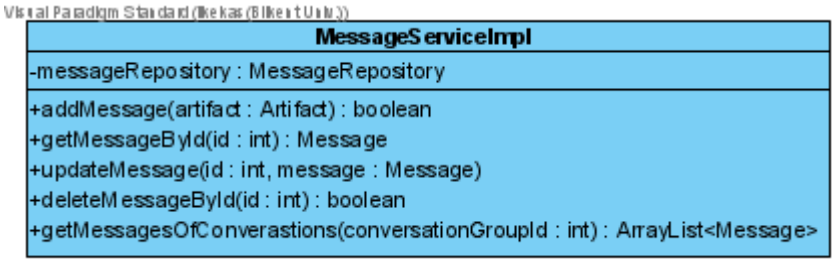
that we can use for database operations.

MessageService Interface

This message service interface will use these functions to save, get, find necessary information from the database.

- public boolean addMessage(Message message)
- public Message getMessage(int id)
- public void updateMessage (int id,Message message)
- public void deleteMessageById(int id)
- public ArrayList<Message> getMessagesOfConversations(int conversationGroupId)

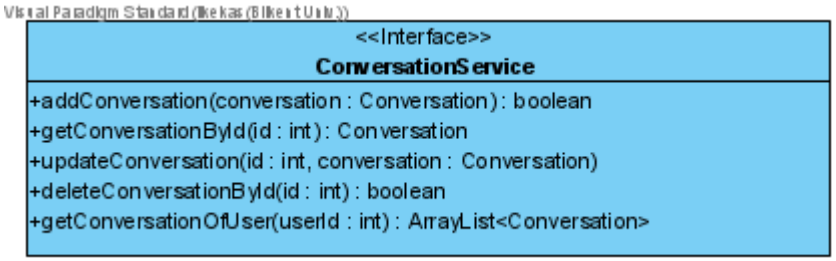
MessageServiceImpl



This class will be @service and it will implement the message Service interface to save, get, find necessary information from the database.

- public boolean addMessage(Message message): This function will add a message to the database system.
- public Message getMessage(int id): This function will return the message from the database by given id.
- public void updateMessage (int id,Message message): This function will update the message with given id.
- public void deleteMessageById(int id): This function will delete the message with the given id from the database system.
- public ArrayList<Message> getMessagesOfConversations(int conversationGroupId): This function will get the conversation with given ids from the database.

ConversationService Interface

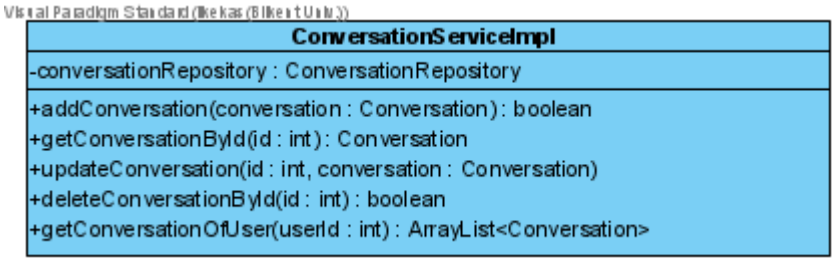


This conversation service interface will use these functions to save, get, find necessary information from the database.

- public boolean addConversation(Conversation conversation)
- public Conversation getConversation(int id)
- public void updateConversation (int id,Conversation conversation)
- public void deleteConversationById(int id)

- public ArrayList<Conversation> getConversationOfUser(int conversationId,int userId)

ConversationServiceImpl



This class will be @service and it will implement the conversation Service interface to save, get, find necessary information from the database.

- public boolean addConversation(Conversation conversation): This function will add a conversation to the database system.
- public Conversation getConversation(int id): This function will return the conversation from the database by given id.
- public void updateConversation (int id,Conversation conversation): This function will update the conversation with given id.
- public void deleteConversationById(int id): This function will delete the conversation with the given id from the database system.
- public ArrayList<Conversation> getConversationOfUser(int conversationId,int userId): This function will get the conversation with given ids from the database.

MessageGroupService Interface

This message group service interface will use these functions to save, get, find necessary information from the database.

- public boolean addMessageGroup(MessageGroup messagegroup)
- public MessageGroup getMessageGroup(int id)
- public void updateMessageGroup (int id,MessageGroup messagegroup)
- public void deleteMessageGroupById(int id)
- public ArrayList<MessageGroup> getMessageGroupsOfUser(int userId)

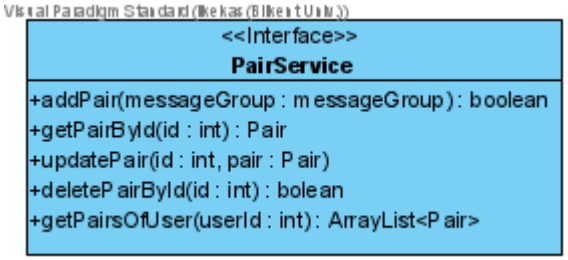
MessageGroupServiceImpl

This class will be @service and it will implement the message group Service interface to save, get, find necessary information from the database.

- public boolean addMessageGroup(MessageGroup messagegroup): This function will add a message group to the database system.
- public MessageGroup getMessageGroup(int id): This function will return the message group from the database by given id.
- public void updateMessageGroup (int id,MessageGroup messagegroup): This function will update the message group with given id.
- public void deleteMessageGroupById(int id): This function will delete the message group with the given id from the database system.

- public ArrayList<MessageGroup> getMessageGroupsOfUser(int userId): This function will get the conversation with given ids from the database.

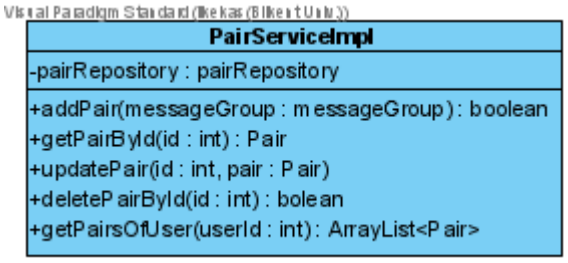
PairService Interface



This pair service interface will use these functions to save, get, find necessary information from the database.

- public boolean addPair(Pair pair)
- public Pair getPair(int id)
- public void updatePair(int id,Pair pair)
- public void deletePairById(int id)
- public ArrayList<Pair> getPairsOfUser(int userId)

PairServiceImpl



This class will be @service and it will implement the pair Service interface to save, get, find necessary information from the database.

- public boolean addPair(Pair pair): This function will add a pair to the database system.
- public MessageGroup getPair(int id): This function will return the message group from the database by given id.
- public void updatePair (int id,Pair pair): This function will update the pair with given id.
- public void deletePairById(int id): This function will delete the pair with the given id from the database system.
- public ArrayList<Pair> getPairsOfUser(int userId): This function will get the pair with given ids from the database.

MessageManagerData Class

This class has all of the classes of the implementation of the service interfaces in this package. From this class, we can use all service implementations to manage the database of the objects of this package.

- public MessageServiceImpl getMessageServiceImpl(): This function will return the MessageServiceImpl object of the class.
- public ConversationServiceImpl getConversationServiceImpl(): This function will return the ConversationServiceImpl object of the class.
- public PairServiceImpl getPairServiceImpl(): This function will return the PairServiceImpl object of the class.

- public MessageGroupServiceImpl getMessageGroupServiceImpl(): This function will return the MessageGroupServiceImpl object of the class.

● Low-level Design

4.1 Object design trade-offs

4.1.1 Functionality vs Cost

Functionality is essential to execute the most basic tasks to be a classroom helper application. As we add new features to our design, our tasks increase. Our aim is to execute the tasks and increase the functionality with minimal cost. Therefore, we choose functionality over cost.

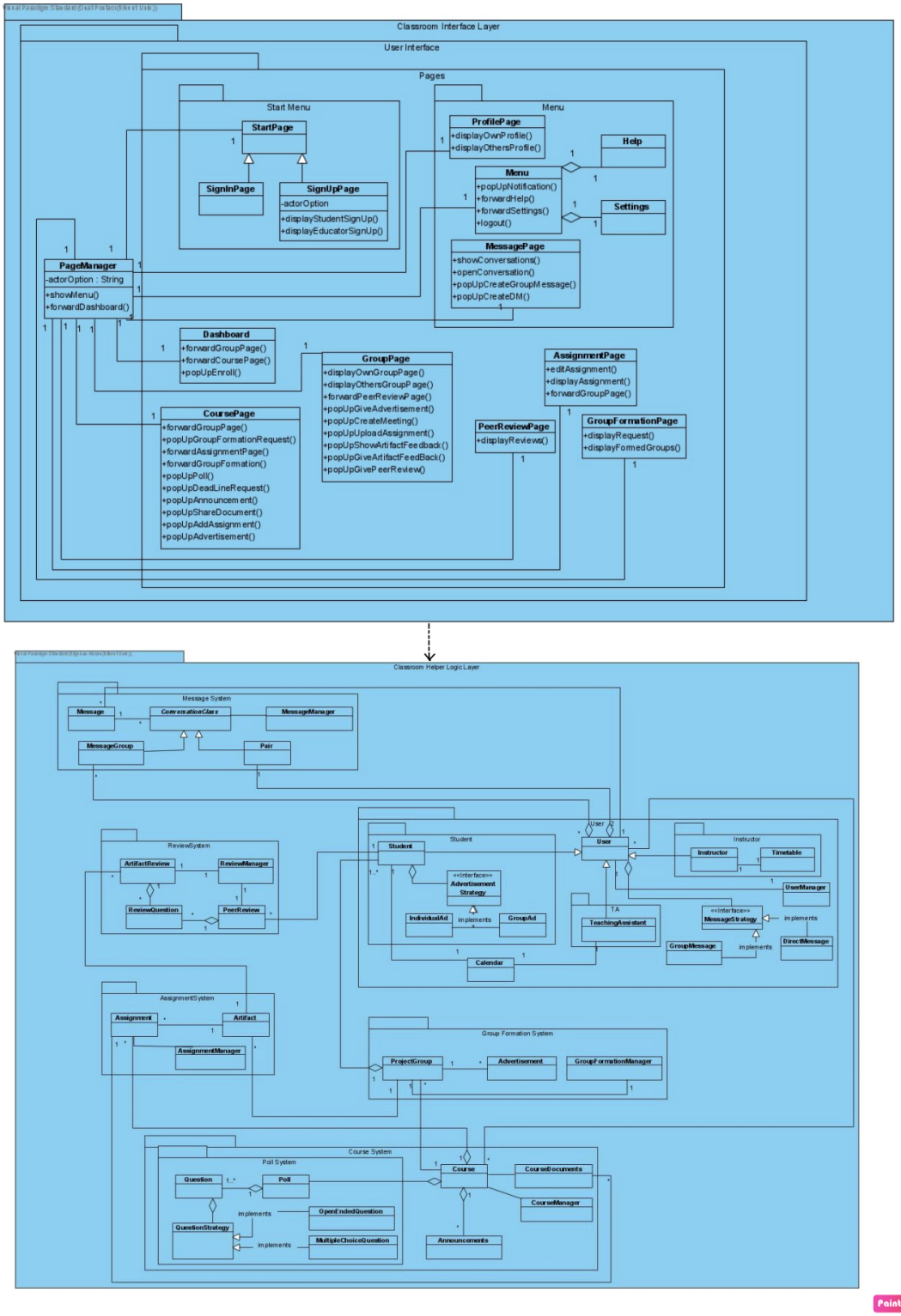
4.1.2 Usability vs Security

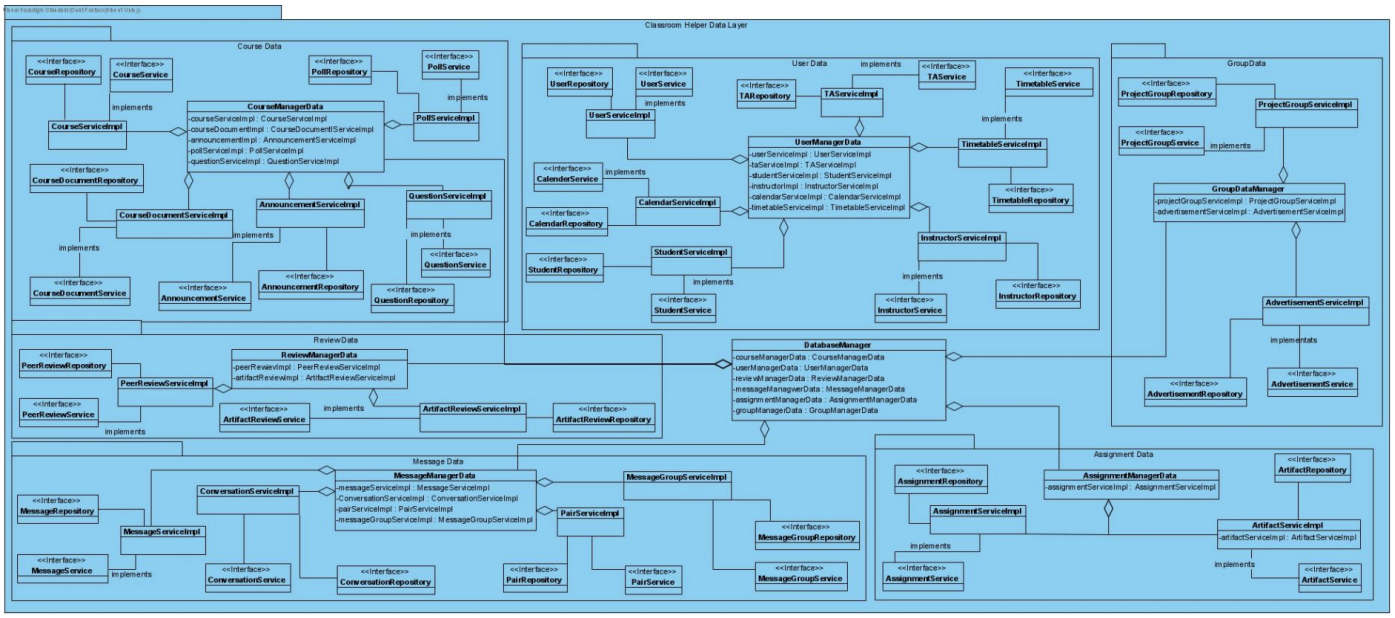
Our program will hold some private data such as the password of the user, reviews of other students' works, or grades; therefore, the security of the program is a priority for us compared to usability even though usability is significant.

4.1.3 Efficiency vs Portability

Using the same software in different environments is essential for our design since it will be developed by group work. Therefore, even though efficiency is inevitably important, portability is indispensable.

4.2 Final object design





4.3 Packages

In Catch Up, there are three layers named Classroom Helper Interface Layer, Classroom Helper Logic Layer and Classroom Helper Data Layer. They are all packages. We use subsystems to decrease complexity of the system by dividing the system into smaller parts.

Our first package is the Classroom Helper Interface Layer package which only has a User Interface package. There is only one common User Interface package in this layer because in our system all different users see almost the same pages. Packages can be divided into smaller parts by creating more packages. For example, inside the User Interface package, there is another package named Pages along with a class for managing pages. Pages package contains different kinds of pages. In order to classify the pages, two more packages are used. One of them is the Menu package. This package contains menu-related pages which are the Help and Setting pages, and

Profile and Message pages. Another package is the Start Menu package consisting of the pages related to the opening screen of the application such as sign in and sign up pages.

Another package is the Classroom Helper Logic Layer package. There are different packages inside the Classroom helper Logic Layer package. First one is the User package which consists of Student, TA, Instructor packages and a UserManagement class. Then there is a Course System package which includes course related classes, CourseManager class and a Poll System package. Moreover, there is a Review System package that provides review service to the system which consists of ArtifactReview, PeerReview and ReviewManagement classes. Then, there is a Group Formation System package which consists of ProjectGroup, Advertisement and GroupFormationManager classes to provide group formation service to the system. Another package is Assignment System which consists of Assignment, Artifact and AssignmentManager classes. Last package in this layer is the Message System that includes Message, ConversationClass, Pair, MessageGroup and MessageManagement classes to provide messaging service to the system.

The last package in our system is the Classroom Helper Data Layer package. In this package, we have two interfaces and one implementation class for each entity in our system. We have repository interfaces that hold each user, student, instructor etc. to be added to the database system. It will extend the JPA Repository class that has many functions that we can use for database operations. Other interfaces are services interfaces that contain the operations necessary to get information from database and set necessary changes to database. Finally, implementation classes will implement services interfaces to access databases. We use manager classes for each subsystem in the data layer, since it will be easier to reach their functions through get

function. Similarly, we use the DataBaseManager class to reach database functions from the logic layer easily. The reason why we have so many structures in the data layer is modularity. If we need data that is not obtained by a defined function in our system, it will be easier to write a function thanks to this modularity. We are also going to implement Java packages which are lang, util and awt. Besides that we will also use the com package to connect database which will be implemented by PostgreSQL to our program.

4.4 Class Interfaces

Interfaces define methods like all classes; however, they do not implement these methods like normal classes. Yet, classes that implement interfaces implement the methods defined by the interfaces. In our design we use the interfaces of KeyboardListener and MouseListener by implementing them in our InputManager class which exists in the package of "Application Management".

4.4.1 KeyboardListener Interface

The KeyboardListener interface keeps track of the events happening on the keyboard and thereby by implementing it, our InputManager class keeps track of the inputs from the keyboard. When a user presses the "enter" key to execute a task, the class will help to execute the following function.

4.4.2 MouseListener Interface

The MouseListener interface keeps track of the clicks of the mouse and thereby by implementing it, our InputManager class keeps track of the inputs from the mouse actions.

● Conclusion

For a project, it is highly essential to divide it into smaller parts in order to handle the project and share the tasks among stakeholders in an easier way. Therefore, for the design part of the project, the system is decomposed into subsystems. We have 3 layers for our project, namely Classroom Helper Interface Layer, Logic Layer and Data Layer. Also, subsystems can be decomposed into smaller parts by creating packages. Packages contain classes that carry out similar activities and contribute to the project in a similar way. Classroom Helper Interface Layer basically organizes the user interface classes and in that subsystem, interactions between the packages and classes are shown to make the user interface design more understandable. Classroom Helper Logic Layer can be considered as the brain of the system. It controls the actions taken by the user and helps to make changes in the user interface and the database by using manager classes. It also focuses on the interactions between the objects of the system. Lastly, the Classroom Helper Data Layer will handle the operations in database systems. Classes in that layer directly interact with the database and they are also packaged to create a meaningful whole in the data layer. Therefore, even though all these subsystems do different jobs, they come together and form a meaningful whole. In addition, while decomposing system and defining the boundaries of every class, package and subsystem, it is also important to pay attention to the design trade-offs such as functionality, usability or efficiency and design accordingly.

- # References

[1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.