Bilkent University

Department of Computer Engineering

CS 319 Object-Oriented Software Engineering

Term Project

CS319: Peer Review App: Snitch

Design Report

Group 1I

Group Members:

Ümit Çivi 21703064

Onat Korkmaz 21704028

Abdulkadir Erol 21703049

Erdem Ege Eroğlu 21601636

Muhammed Maruf Şatır 21702908

Instructor: Eray Tüzün

Teaching Assistant(s): Elgun Jabrayilzade, Erdem Tuna, Barış Ardıç

Progress/Design Report IT2

25.04.2021

This report is submitted to the Department of Computer Engineering of Bilkent University.

Table of Contents

1.Introduction

1.1 Purpose of the system

The purpose of the system is to provide an all-in-one system that will operate peer reviews in a group project. The system provides students to scale and comment on peers of the same group. Students will be able to view other groups' artifacts and comment on the artifacts. Hosts and TAs can categorize, arrange or rank the reviews after the process is over. Moreover, in this system, hosts will be able to delete groups, change groups or assign students to groups.

1.2 Design Goals

1.2.1 Functionality

In addition to the basics of the peer review system, we will try to implement additional functionalities. For instance, in the case of a collapsing database, we will send email that will include the information about grades every 4 hours on grading day.

1.2.2 Performance

The peer-reviewing system will have a limited number of users which is about 175 users. There will not be mass throughput so the performance of the system will not be highly affected by the execution of the tasks at the same time. In terms of response time, the peer-reviewing system will have a response time maximum of 0.25 milliseconds.

1.2.3 Dependability

The peer-review system will have great durability as the system will be not affected by the wrong user input since the system will display warning messages. The system will be able to provide service every time. The system will not threaten human lives and will not have negative effects on the environment. Moreover, the reliability of the system will not be an issue, the system will provide reliable information both while storing data and retrieving data from the

database for display results. The language decision which is PHP will also assist the reliability of the system.

1.2.4 Rapid Development

The aim of our team will be certain to catch up with the demo deadline. As 5 people in our team, we will start coding after the demo presentation. Firstly, we will try to implement the basic functionalities of the peer review system such as creating groups, adding a student to particular groups, and certainly peer grading-commenting. If there will be sufficient time to deadline, we will try to implement additional functionalities to our peer review system.

1.2.5 Readability

We do not have enough clue about others' style of coding. Hence, we will use meaningful names to identifiers and comment on the functionalities of the program in crucial points.

1.2.6 Ease of Learning

In the peer review system, we had decided to make the GUI of the system simple and easy to understand. We will make this aim possible by making the button names reflecting their functionalities, sorting will be indicated by the choosing specific checkbox, commenting and grading will include just the choosing peer, grading and commenting on him/her.

1.2.7 Usability

As we will develop this as a web based application, a device that has access to enter websites, can use the system as their input devices. System will provide valid output to the interaction of users to the elements of GUI.

1.3 Trade-Offs

In this part all of the design goal trade-offs outlined will be addressed on the basis of our course contents. We will be declaring two opposing design goals in terms of not only importance but also reasons why we have decided to do so.

1.3.1 Functionality and Usability

We are planning to focus on the usability side. This is mainly because we want to implement a peer review system that is quite easy to use and our users are able to complete their work quickly and correctly. We really care about our users' time. Thus, this usability advantage will increase our users' productivity.

1.3.2 Cost and Robustness

Since this project has a large contribution to the letter grade of the course with 4 credits, we do not care about cost. We will be always trying to increase robustness. Rather than cost, we will spend more time on our project.

1.3.3 Efficiency and Portability

We will use PHP as our programming language. Since PHP is platform independent, we do not need to focus on portability. We will build a web application but, platform dependent means same code may not run on all browsers. Thus, that is not a design goal for our. We will be focusing on efficiency.

1.3.4 Rapid Development and Functionality

According to priority we might cut-off some functionalities which are labeled as low priority in order to increase the pace of development and maximize our cost-effectiveness. That is why we are planning to focus on the rapid development side.

1.3.5 Cost and Reusability

We want to implement a peer review system that is usable for a long term. After this semester, coming cs students should use our peer review system to finalize their projects. Therefore, in this trade-off we will focus on reusability.

1.3.6 Backwards Compatibility and Readability

We do not have a previous peer review project; therefore, backward compatibility is not a design goal for our project. That is why we will automatically be optimizing our readability. Since the project requires group work, we need to focus on readability because all group members ought to understand all pieces of code.

1.3.7 Security and Usability

In this trade-off, we will focus on security for the consistency of grades and we do not want people out of the course. Also, grades should be private to anyone. That is why we need to optimize security.

1.4 Definitions, Acronyms, and Abbreviations

In this section acronyms, abbreviations which we will use in our peer review system project is defined.

1.4.1 Personal Computer (PC)

A personal computer in short PC, is a multi-purpose computer that is intended for personal use.

1.4.2 Graphical User Interface (GUI)

Graphical User Interface in short GUI, is any user interface that makes the interaction with the user possible.

1.4.3 Model View Controller (MVC)

Model view controller in short MVC, consist of three parts. Those parts are Model, View, and Controller. Actual entities and objects compose of model parts. View part is composed of objects having to do with any user experience. Control objects compose to control parts.

1.5 Overview

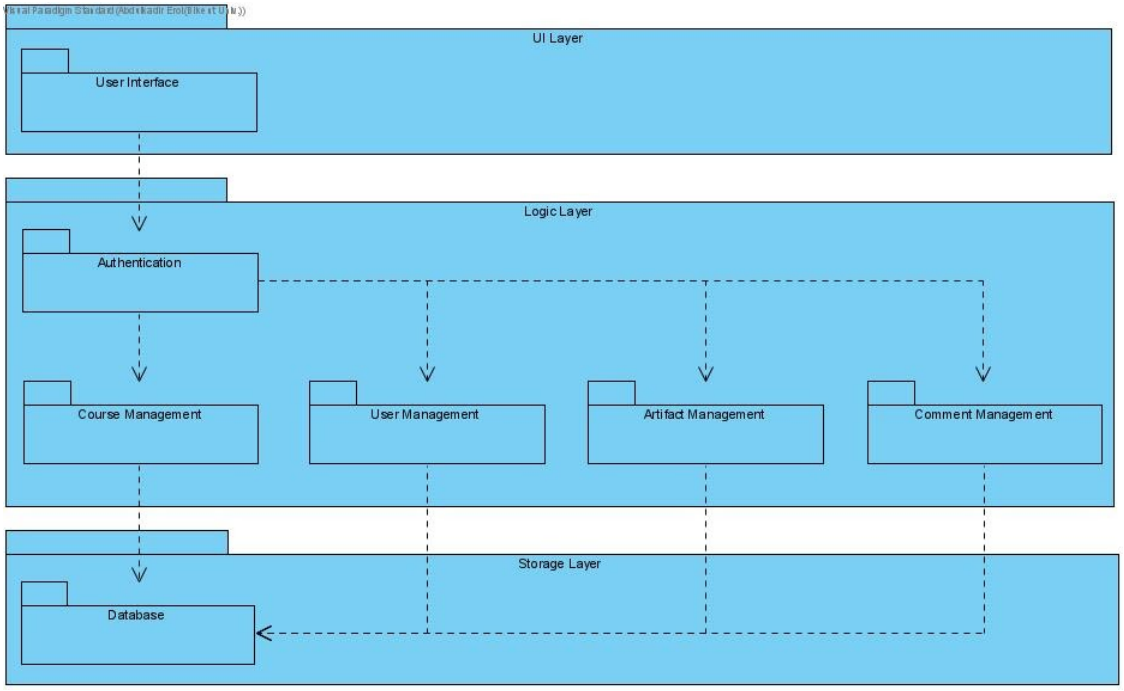
In our peer review project, we have three main design goals that we will be focusing on mostly. Those design goals are reusability, usability and robustness. In terms of trade-offs, we will be optimizing usability as opposed to functionality, robustness as opposed to cost, rapid development as opposed to functionality, reusability as opposed to cost, readability as opposed backwards compatibility, and security as opposed to usability.

2. Proposed Software Architecture

2.1 Overview

Our system is a web application and the MVC architecture will be used for decomposing our system into subsystems. We implemented MVC design by using 3-tier architecture. All the changes and related adjustments will be saved in the database. No additional hardware except keyboard and mouse will be needed to run the system. HTML, CSS, and PHP will be used to create the system.

2.2 Subsystem Decomposition

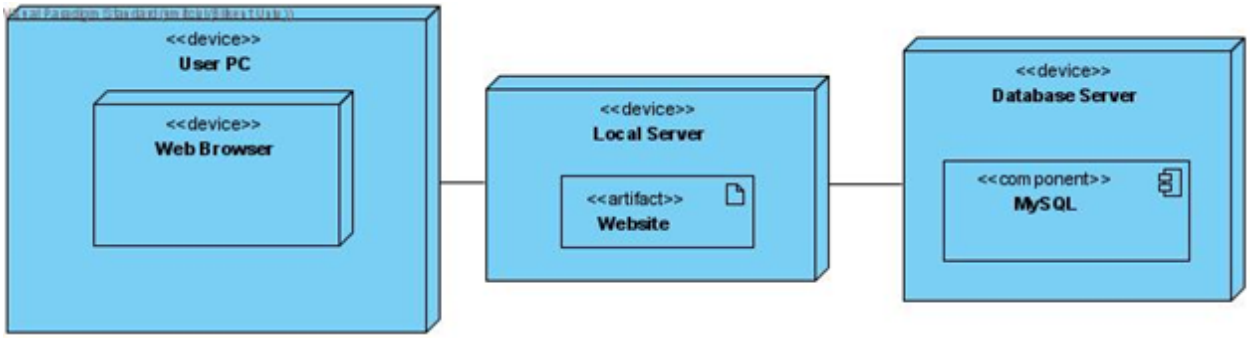


2.3 Hardware/Software Mapping

The project will be implemented with PHP. The user is required to have an internet connection and a web browser to run the website. The project requires a mouse in order to navigate inside the website and a keyboard to enter the comments regarding an artifact or students.

For storage, we will use a database. Therefore, there will not be any hard disk usage in a PC except the local server.

Since PHP is a server-side scripting language, the web server used in the website should have a compatible version with the language. We are using PHP 8.0, therefore, the minimum required Apache version is 2.4.0.



2.4 Persistent Data Management

For persistent data management, MySQL (a relational database) will be used to store and manage the data. Since the group members do not know how to use MySQL, phpMyAdmin, which is an administration tool for MySQL databases, will be used. The database will store the data of student/host accounts, artifacts, peer reviews, artifact reviews, groups list, sections and course information. In order to increase the security of the system, only a few people will access the database using authentication.

2.5 Access Control and Security

We will be using network components for our project. That makes our peer review open to attack and difficult to control. Initially, our system allows only bilkent web mail. Therefore, we will not be storing sensitive data and there will be some features to keep the grades unchanged. Initially, our system will accept new users with e-mail configuration, an account of these features, there will be no one out of the course and each student can have a single account in our system. We will be using a remote database, in order to prevent grades from changing, students will have limited time to grade their peers and they start their own grading period on their own thanks to email verification. Every 4 hours of grading day and at the end of the grading time period TA's get an email about grades. Grades cannot be changed, so If there are differences with e-mail that is sent to TA's and the current system database. Grades at E-mail will be valid. Also, one student can't see another's comment until the teacher allows.

Table 1: Access control matrix for main SNITCH objects

Objects / Actors	Host	Student
Course	<<create>> getCourseInformation setCourseInformation viewCourse	
Section	<<create>> getSectionInformation setSectionInformation viewSection	
Group	<<create>> getGroupInformation setGroupInformation viewGroup create/delete groups	viewGroup getGroupInformation
Student	getStudentInformation viewStudent changeGroupOfStudent	<<create>> getStudentInformation setStudentInformation viewStudent reviewPeer
Artifact	getArtifactInformation viewArtifact reviewArtifact	<<create>> getArtifactInformation setArtifactInformation viewArtifact reviewArtifact
StudentComment	getStudentComment viewStudentComment	<<create>> getStudentComment setStudentComment viewStudentComment
ArtifactComment	<<create>> getArtifactComment setArtifactComment viewArtifactComment	<<create>> getArtifactComment setArtifactComment viewArtifactComment
Deadline	<<create>> getDeadline setDeadline viewDeadline extendDeadline	getDeadline viewDeadline
Host	<<create>> getHostInformation setHostInformation viewHosts	

2.6 Global Software Control

In the peer review system, we had decided to use an event driven control mechanism. Since the peer review system has to interact with one user at a time and events will not be concurrent, the usage of event driven control will assist us to make simpler the interaction between user and the system. Peer review system will react to the events with the order of their arrival time. Moreover, the event handler mechanism will detect the movement of the user such as clicking the mouse and pressing key.

2.7 Boundary Conditions

2.7.1 Initialization

This is a Web application, so for a user to initialize it, s/he is required to have connection to the Internet. After the user is accessed to the app, the first thing that comes up on the interface is to register or log-in. Entering invalid login information will result in unsuccessful initialization of the app. Until the user enters valid login information, the interface will keep asking for email and password. Another scenario is when the user presses the sign-up button, s/he should have a valid Bilkent mail account. For authentication of the student, a mail address with “bilkent.edu.tr” extension should be given with a valid password (minimum 8 characters, uppercase and lowercase letters etc.).

2.7.2 Termination

When the close button is pressed, the user is logged off from his/her account and any unsaved changes will be lost. Also, there will be an exit button to sign off, and there will be a warning message to remind users to save the data.

2.7.3 Failure

Internet connection is required to access the application. Internet connection loss during peer review or artifact review will cause the deletion of unsaved data. Therefore, it is important to save the changes made by the users before exiting the application.

In all cases where data is saved to database or fetched from database for the first time, the connection to the database must be established. In a case in which the connection cannot be established, the user will be notified about the problem. Moreover, the user will be prompted to perform the unsuccessful task again.

3. Subsystem Services

User Interface

The User Interface subsystem is a part of UI Layer. It provides each user with a predefined interface depending on their roles, which are hosts (instructor & teacher assistant) and student. From here the users will have access to the functions according to their roles. Authorization services are required to login, sign-up and display the interface. It provides communication between the users and the rest of the subsystems.

Authentication

Authentication subsystem is the initial subsystem of Logic Layer and it provides an access to other Logic Layer subsystems. Also, handles user registration if the user does not have an account. With this subsystem, a layer of security is established. This subsystem works together with the Database subsystem to authenticate users.

Course Management

The Course Management subsystem is a part of Logic Layer. It provides any functions related to course and the sections.

User Management

The User Management subsystem is also a part of Logic Layer. It provides any user related functions in the system. Any functions done by hosts (instructor and teaching assistant) and students related to changing user specifications are in this system.

Artifact Management

The artifact management subsystem is also a part of Logic Layer. It provides artifact related functions.

Comment Management

The Comment Management subsystem is also a part of Logic Layer. It provides comment related functions. Comments include the reviews about the students or a particular artifact.

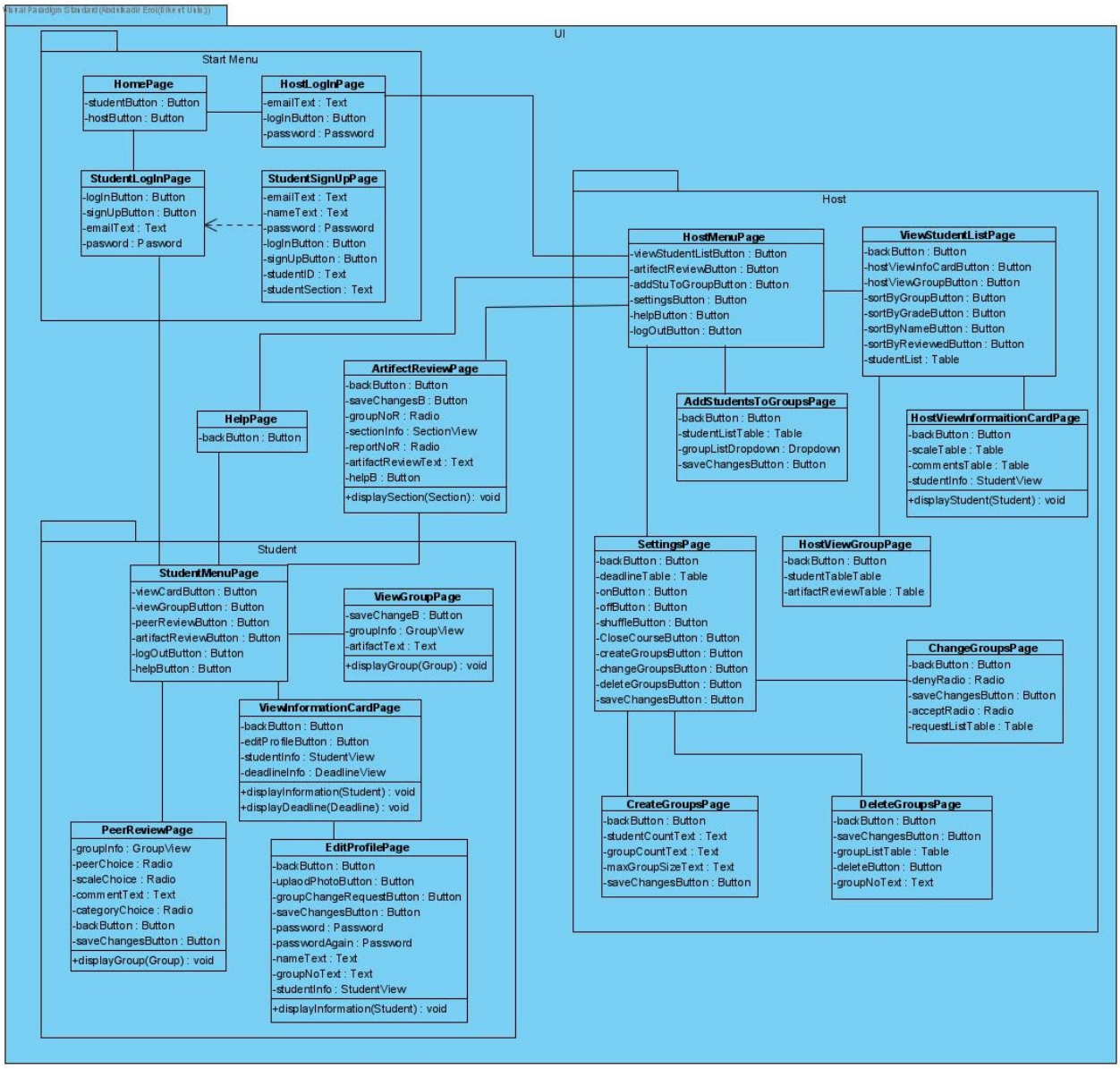
Database

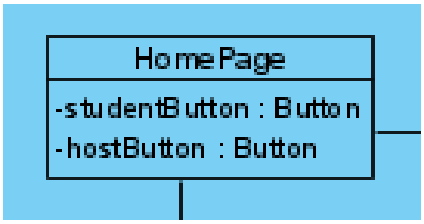
The Database subsystem is a part of Storage Layer. It provides storage and retrieval to the whole system. All other subsystems interact with the database subsystem to fetch or store crucial data.

3.1 Object Design

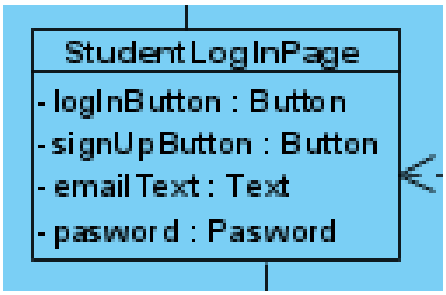
As an object design, the subsystem decomposition diagram (in page 8) is used. As it can be seen, there are three layers in the subsystem decomposition, UI Layer, Logic Layer, and Storage Layer.

3.1.1 UI Layer

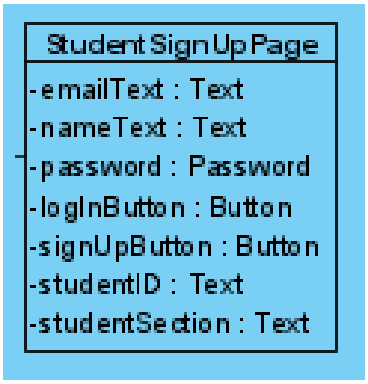




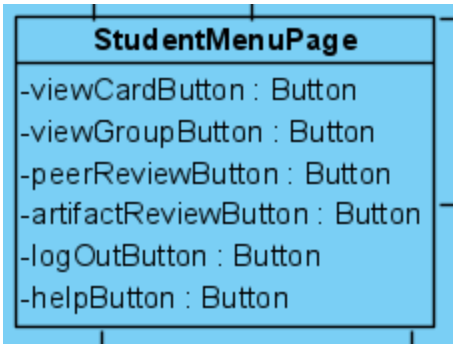
Class Name: HomePage
Attributes
• private Button studentButton: directs user to StudentLogInPage • private Button hostButton: directs user to HostLogInPage
Methods



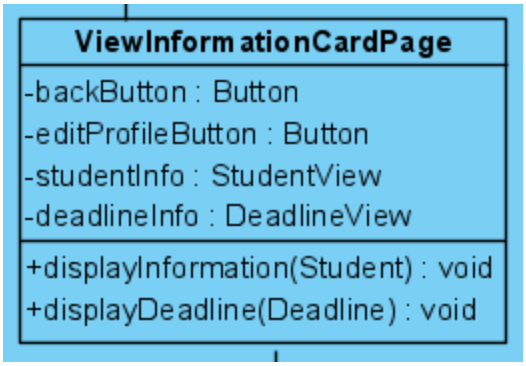
Class Name StudentLogInPage
Attributes
• private Button logInButton: directs student to StudentMenuPage • private Button signUpButton: directs student to StudentSignUpPage • private Text emailText: text field for email • private Password password: password field for password
Methods



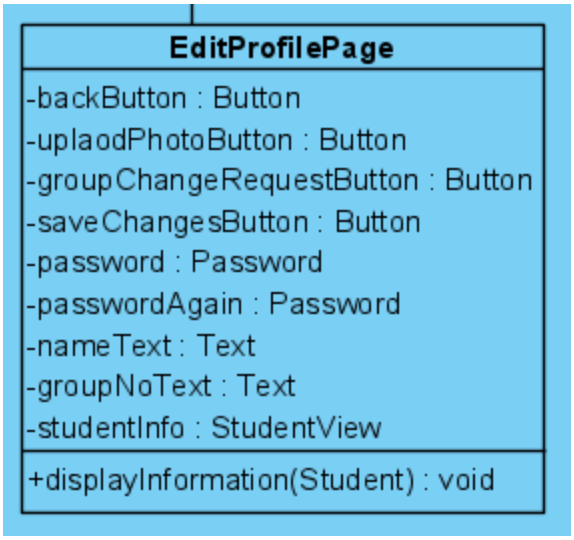
Class Name StudentSignUpPage
Attributes
• private Button loginButton: directs student to StudentLogInPage • private Button signUpButton: submits changes and directs student to StudentLogInPage • private Text nameText: text field for name • private Text emailText: text field for email • private Password password: password field for password • private Text studentID: text field for student id • private Text studentSection: text field for student section
Methods



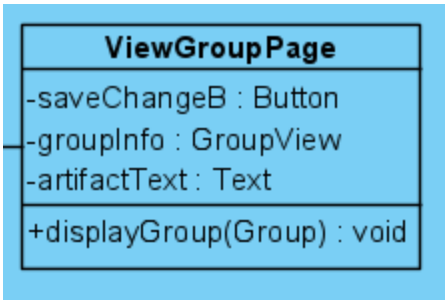
Class Name StudentMenuPage
Attributes
• private Button viewCardButton: directs student to ViewInformationCardPage • private Button viewGroupButton: directs student to ViewGroupPage • private Button peerReviewButton: directs student to PeerReviewPage • private Button artifactReviewButton: directs student to ArtifactReviewPage • private Button helpButton: directs student to HelpPage • private Button logOutButton: directs student to log out from the system
Methods



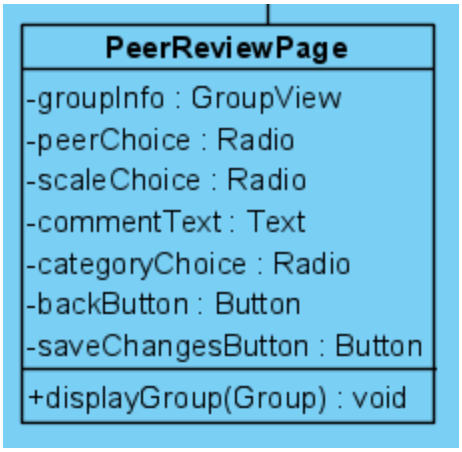
Class Name ViewInformationCardPage
Attributes
• private Button backButton: directs student to StudentMenuPage • private Button editProfileButton: directs student to ViewGroupPage • private StudentView studentInfo: holds information about student • private DeadlineView deadlineInfo: holds deadline date for submission
Methods
• public void displayInformation(Student): displays student's relevant information • public void displayDeadline(Deadline): displays deadline for a submission



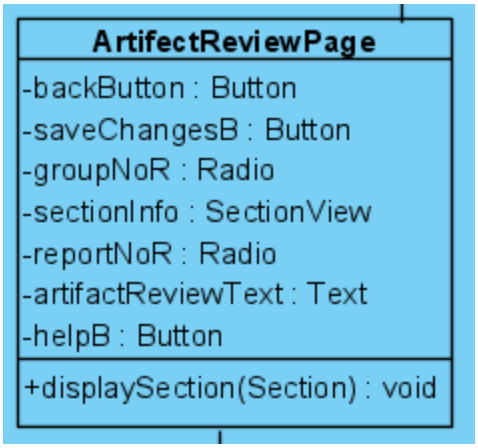
Class Name EditProfilePage
Attributes
• private Button backButton: directs student to ViewInformationCardPage • private Button uploadPhotoButton: uploads a photo • private Button saveChangesButton: submit changes • private Button groupChangeRequestButton: sends request to the host for group change • private Text nameText: text field for name • private Password password: password field for password • private Password passwordAgain: password field for password again • private Text groupNoText: text field for group no text • private StudentView studentInfo:
Methods
• public void displayInformation(Student): displays student's relevant information



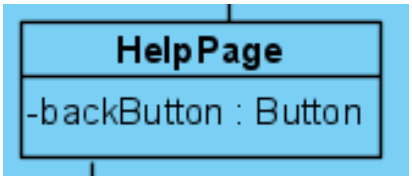
Class Name ViewGroupPage
Attributes
• private Button saveChangesButton: submit changes • private GroupView groupInfo: holds information about group • private Text artifactText: text field for artifact
Methods
• public void displayGroups(Group): displays group members



Class Name PeerReviewPage
Attributes
• private Button backButton: directs student to StudentMenuPage • private Button saveChangesButton: submit changes • private Radio peerChoice: used for choosing among peers • private Radio categoryChoice: used for choosing among categories • private Radio scaleChoice: used for choosing among scales • private Text commentText: text field for comment • private GroupView groupInfo: holds group information
Methods
• public void displayGroup(Group): displays group members

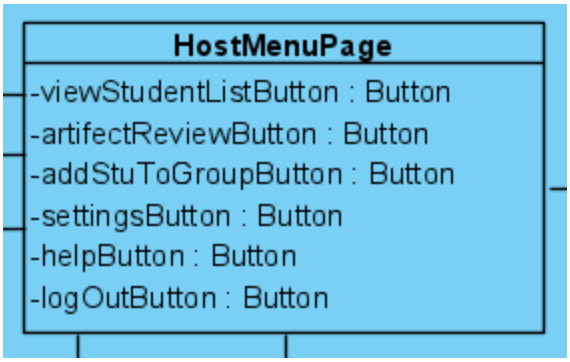


Class Name ArtifactReviewPage
Attributes
• private Button backButton: directs student to StudentMenuPage • private Button saveChangesButton: submit changes • private Radio groupNoR: used for choosing among groups • private Radio reportNoR: used for choosing among reports • private Text artifactReviewText: text field for artifact review • private SectionView sectionInfo: section view for sections
Methods
• public void displaySection(Section): displays group lists

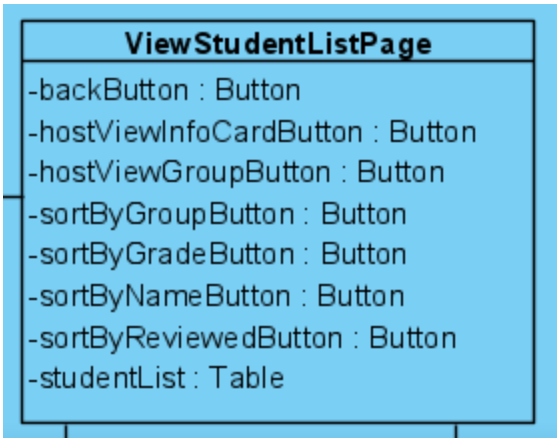


Class Name HelpPage
Attributes
• private Button backButton: directs student to StudentMenuPage
Methods

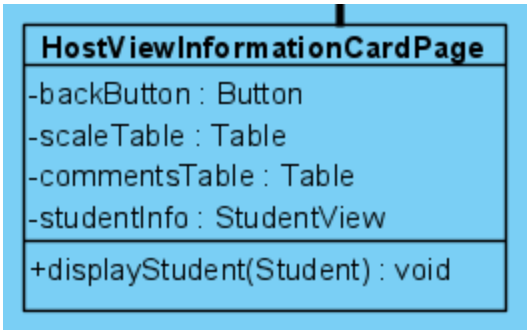
Class Name HostLogInPage
Attributes
• private Button logInButton: directs host to HostMenuPage • private Text emailText: text field for email • private Password password: password field for password
Methods



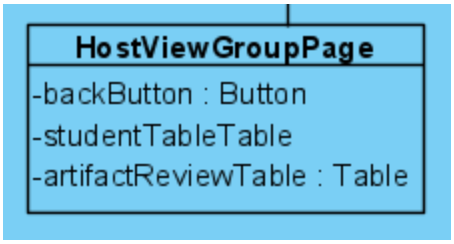
Class Name HostMenuPage
Attributes
• private Button viewStudentListButton: directs host to ViewStudentListPage • private Button artifactReviewButton: directs host to ArtifactReviewPage • private Button addStuToGroupButton: directs host to AddStuToGroupButton • private Button settingsButton: directs host to SettingsPage • private Button helpButton: directs host to HelpPage • private Button logOutButton: directs host to log out from the system
Methods



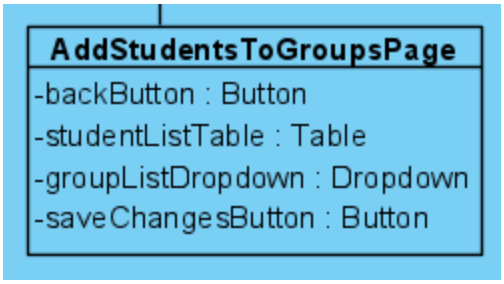
Class Name ViewStudentListPage
Attributes
• private Button backButton: directs host to HostMenuPage • private Button hostViewInfoCardButton: directs host to hostViewInfoCardPage • private Button hostViewGroupButton: directs host to hostViewGroupPage • private Button sortByGroupButton: sorts students by their groups • private Button sortByGradeButton: sorts students by their grades • private Button sortByNameButton: sorts students by their names • private Button sortByReviewedButton: sorts students by their reviewed/non reviewed status • private Table studentList: lists all of the students
Methods



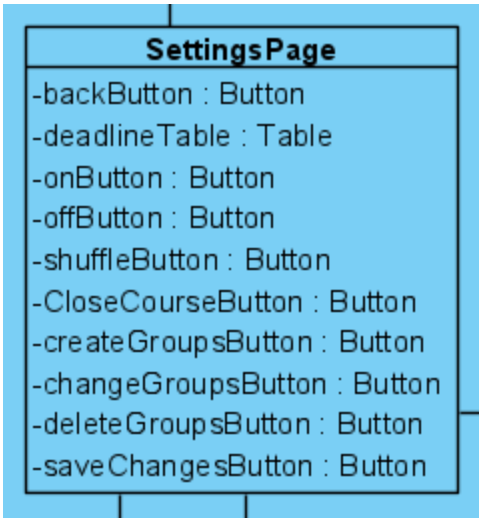
Class Name HostViewInformationCardPage
Attributes
• private Button backButton: directs host to ViewStudentListPage • private Table ScaleTable: table that will display the scales about students • private Table CommentTable: Table that will display comments • private StudentView StudentInfo: Student info object that will get the info from the database
Methods
• public void displayInformation(Student): displays student's relevant information



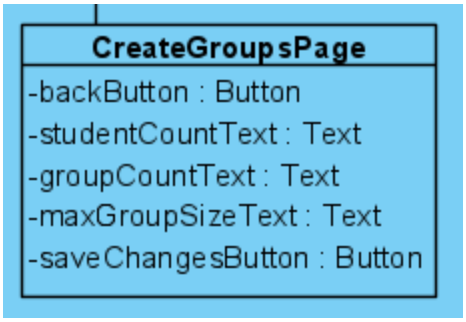
Class Name HostViewGroupPage
Attributes
• private Button backButton: directs host to ViewStudentListPage • private Table StudentTable: lists students in the particular group • private Table ArtifactReviewTable: lists artifact reviews in the particular group
Methods



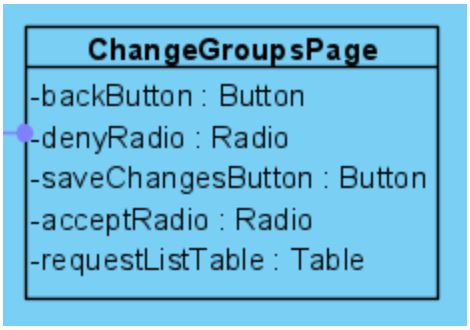
Class Name AddStudentsToGroup
Attributes
• private Button backButton: directs host to HostMenuPage • private Button saveChangesButton: save changes • private Table StudentListTable: lists students • private DropDown groupListDropDown: drop downs the group list
Methods



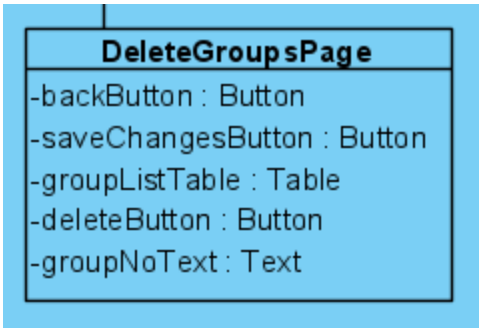
Class Name SettingsPage
Attributes
• private Button backButton: directs host to HostMenuPage • private Button saveChangesButton: save changes • private Button shuffleButton: shuffles students with no groups to remaining groups • private Button onButton: review scales and comments are hidden • private Button offButton: review scales and comments are not hidden • private Button closeCourseButton: closes the course • private Button createGroupsButton: directs host to CreateGroupsPage • private Button changeGroupsButton: directs host to ChangeGroupsPage • private Button deleteGroupsButton: directs host to DeleteGroupsPage
Methods



Class Name CreateGroupsPage
Attributes
• private Button backButton: directs host to SettingsPage • private Button saveChangesButton: save changes • private Text studentCountText: text for number of students • private Text groupCountText: text for number of groups • private Text maxGroupSize: text for maximum number of students in a group
Methods

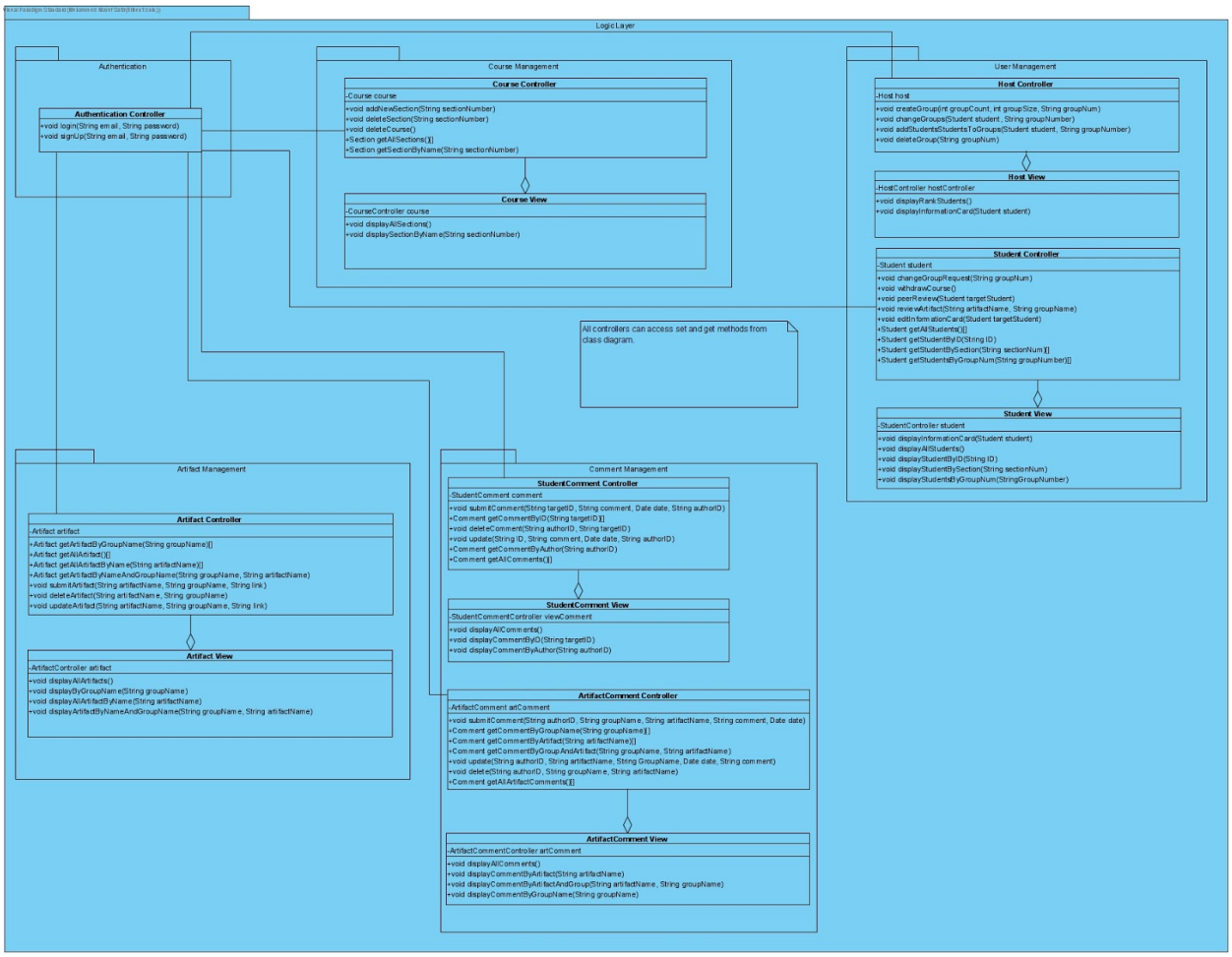


Class Name ChangeGroupsPage
Attributes
• private Button backButton: directs host to SettingsPage • private Button saveChangesButton: save changes • private Radio denyRadio: denying students for changing group • private Radio acceptRadio: accepting students for changing group • private Table requestListTable: table of students request for group change
Methods

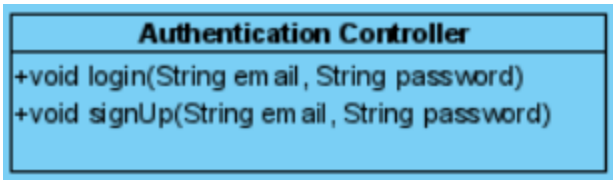


Class Name DeleteGroupsPage
Attributes
• private Button backButton: directs host to SettingsPage • private Button saveChangesButton: save changes • private Button deleteGroup: deleting a particular groups • private Text groupNoText: text for group number • private Table groupListTable: table of group lists
Methods

3.1.2 Logic Layer

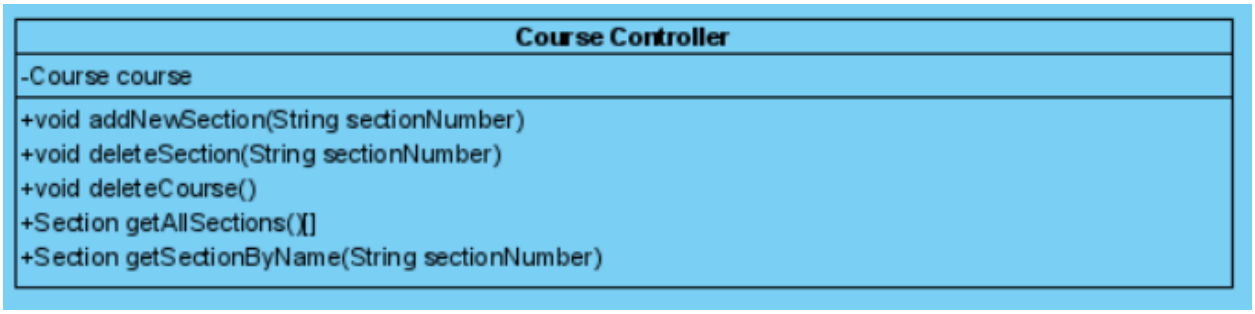


3.1.2.1 Authentication

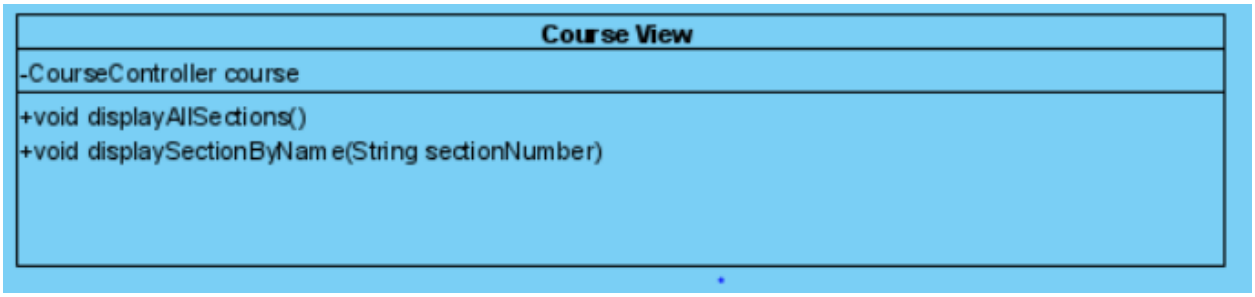


Class Name StudentCommentController
Attributes
Methods
• public void login(String email, String password): Checks the email and password and whether the user is a host or student, it directs to their menu page. • public void signUp(String email, String password) Only students can sign up and email and password will be stored in the database.

3.1.2.2 Course Management

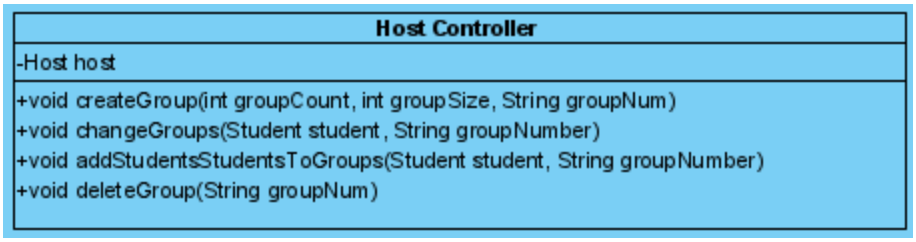


Class Name CourseController
Attributes
• private Course course: course object so that the controller can use course class functions
Methods
• public void addNewSection(String sectionNumber): adds section by its number, sectionNumber • public void deleteSection(String sectionNumber): deletes section by its number, sectionNumber • public void deleteCourse(): deletes the course • public Section getAllSections(): gets all the sections • public Section getSectionByName(String groupName): gets section by their name, groupName

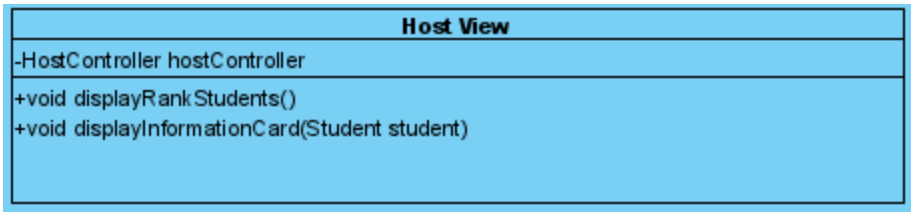


Class Name CourseView
Attributes
• private CourseController course: course controller object so that the course view can access controller functions
Methods
• public void displayAllSections(): displays all sections • public void displaySectionByName(String sectionNumber): displays section by sectionNumber

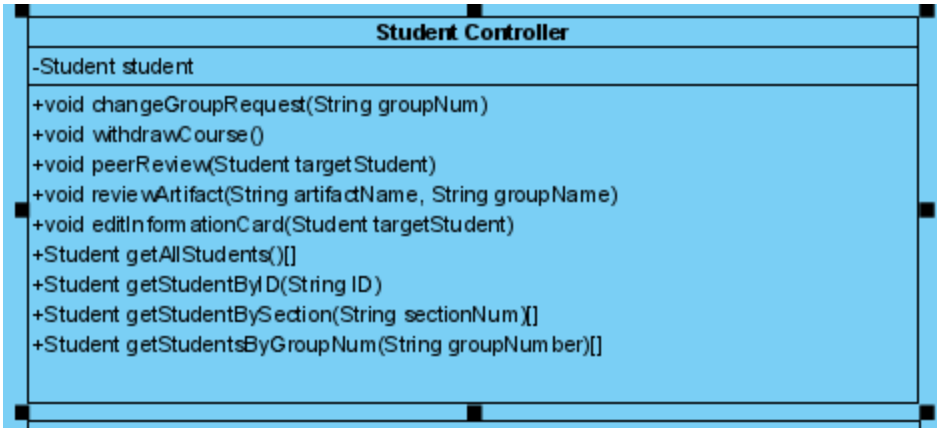
3.1.2.3 User Management



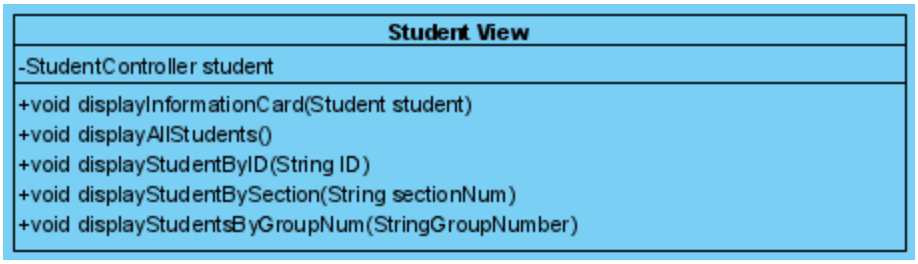
Class Name Host Controller
Attributes
• private Host host: Holds the object of Host so the controller can access the host functions and attributes.
Methods
• void createGroup(int groupCount, int groupSize, String groupNum): Creates groups with indicating its number of students. • void changeGroup(Student student, String groupNumber): Changing student group who is already in group. • void addStudentsToGroups(Student student,String groupNumber): Adding student who is not in group to specific group. • void deleteGroup(String groupNum): Deletes the group indicating by the group number.



Class Name Host View
Attributes
• private HostContoller hostController: Holds the HostController object so that it can access the attributes and functions of HostController class.
Methods
• public void displayRankStudents(): displays a list of students by rank • public void displayInformationCard(Student student): displays all information about single student

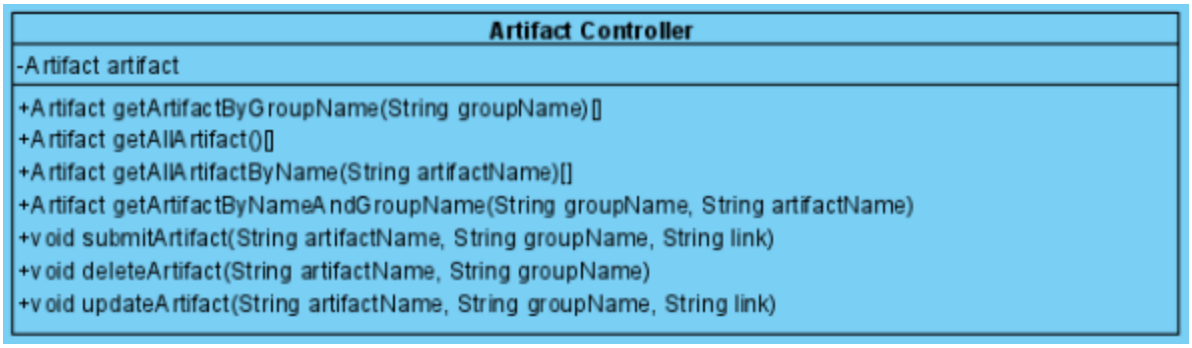


Class Name StudentController
Attributes
• private Student student: Holds the object of the student so that it can access the attributes and functions of Student class.
Methods
• public void changeGroupRequest(String groupNum): Student sends the request for change group indicating it is target group number. • public void withdrawCourse(): Student withdraws the course this causes all of the data for student in database to be deleted. • public void peerReview(Student targetStudent): Student reviews the his/her peer. • public void reviewArtifact(String artifactName, String groupName): Student reviews the artifact of the other groups. • public void editInformation(Student targetStudent): Student can edit his information card. • public Student getAllStudents()[]: returns all the students objects. • public Student getStudentByID(String ID): returns the student object by searching its ID. • public Student getStudentBySection(String sectionNum): Returns the student objects who are in the section number indicated in parameter. • public Student getStudentsByGroupNum(String groupNumber): Returns the student objects which are in the specified group number.

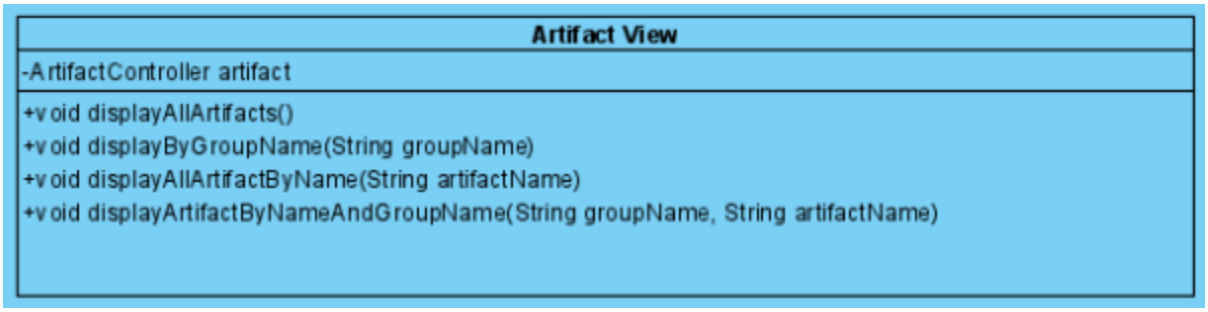


Class Name StudentView
Attributes
• private StudentController student: Holds the StudentController object so that it can reach the attributes and functions of StudentController class.
Methods
• public void displayInformationCard(Student student): displays information about student • public void displayAllStudents(): displays all of the student in the course • public void displayStudentByID(String ID): finds student with ID and displays • public void displayStudentBySection(String sectionNum): displays all of the students in specified section

3.1.2.4 Artifact Management

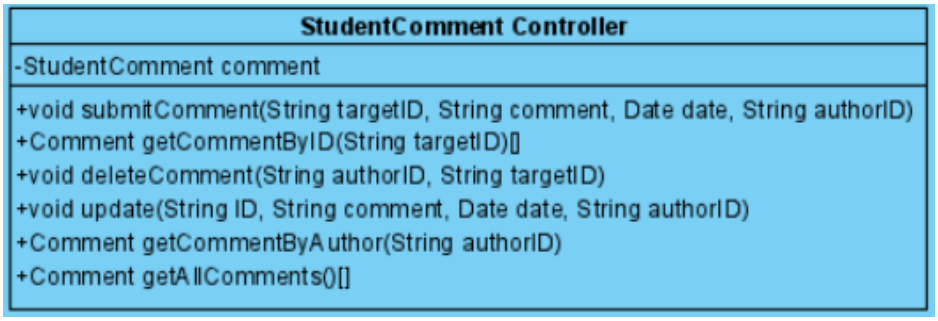


Class Name Artifact Controller
Attributes
• private Artifact artifact: an artifact object so that the controller can access the functions inside the artifact class
Methods
• public Artifact getArtifactByGroupName(String groupName): gets artifact by their group name, groupName • public Artifact getAllArtifact(): gets all the artifacts • public Artifact getAllArtifactByName(String artifactName): gets all artifacts by their name, artifactName • public Artifact getArtifactByNameAndGroupName(String groupName, String artifactName): gets all artifacts by their name and group name, artifactName and groupName • public void submitArtifact(String artifactName, String groupName, String link): submits artifact with its attributes • public void deleteArtifact(String artifactName, String groupName): deletes the specified Artifact • public void updateArtifact(String artifactName, String groupName, String link): updates the specific artifact

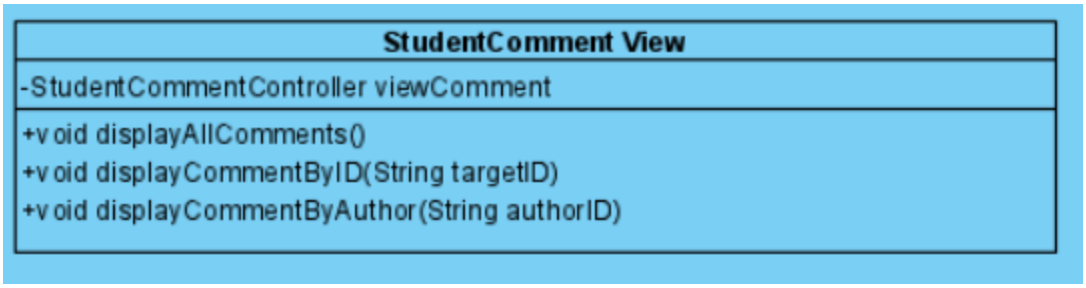


Attributes
• private ArtifactController artifact: artifact controller object so that the view class can access the controller functions
Methods
• public void displayAllArtifacts(): displays all of the artifacts • public void displayByGroupName(String groupName):display all artifacts of a specific group • public void displayAllArtifactByName(String artifactName): displays all artifacts in the same category • public void diplayArtifactByNameGroupName(String groupName, String artifactName): displays a specific artifact of a single group

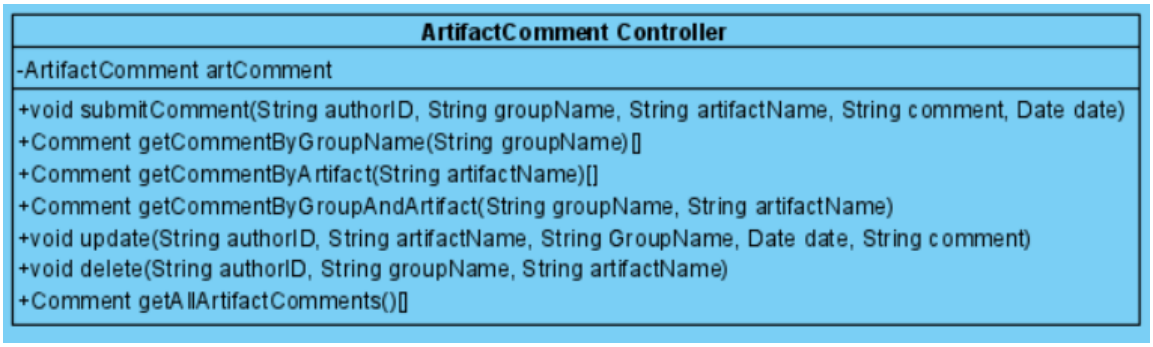
3.1.2.5 Comment Management



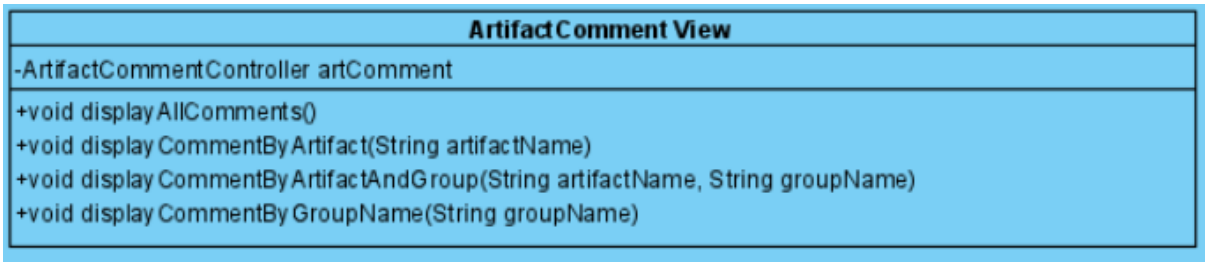
Class Name StudentCommentController
Attributes
• private StudentComment comment: student comment object so that the controller can access student comment class functions
Methods
• public void submitComment(String targetID, String comment, Date date, String authorID): submits comment with its attributes • public Comment getCommentByID(String targetID): gets the comment according to targetID • public void deleteComment(String authorID, String targetID): deletes comment • public void update(String ID, String comment, Date date, String authorID): updates comment with its attributes • public Comment getCommentByAuthor(String authorID): gets the comment according to authorID • public Comment getAllComments: gets all the comments



Class Name StudentCommentView
Attributes
• private StudentCommentController viewComment: controller object so that the view class can access controller functions
Methods
• public void displayAllComments: displays all of the comments • public void displayCommentByID(String targetID): displays targetID's comments • public void displayCommentByAuthor(String authorID): displays authorID's comments

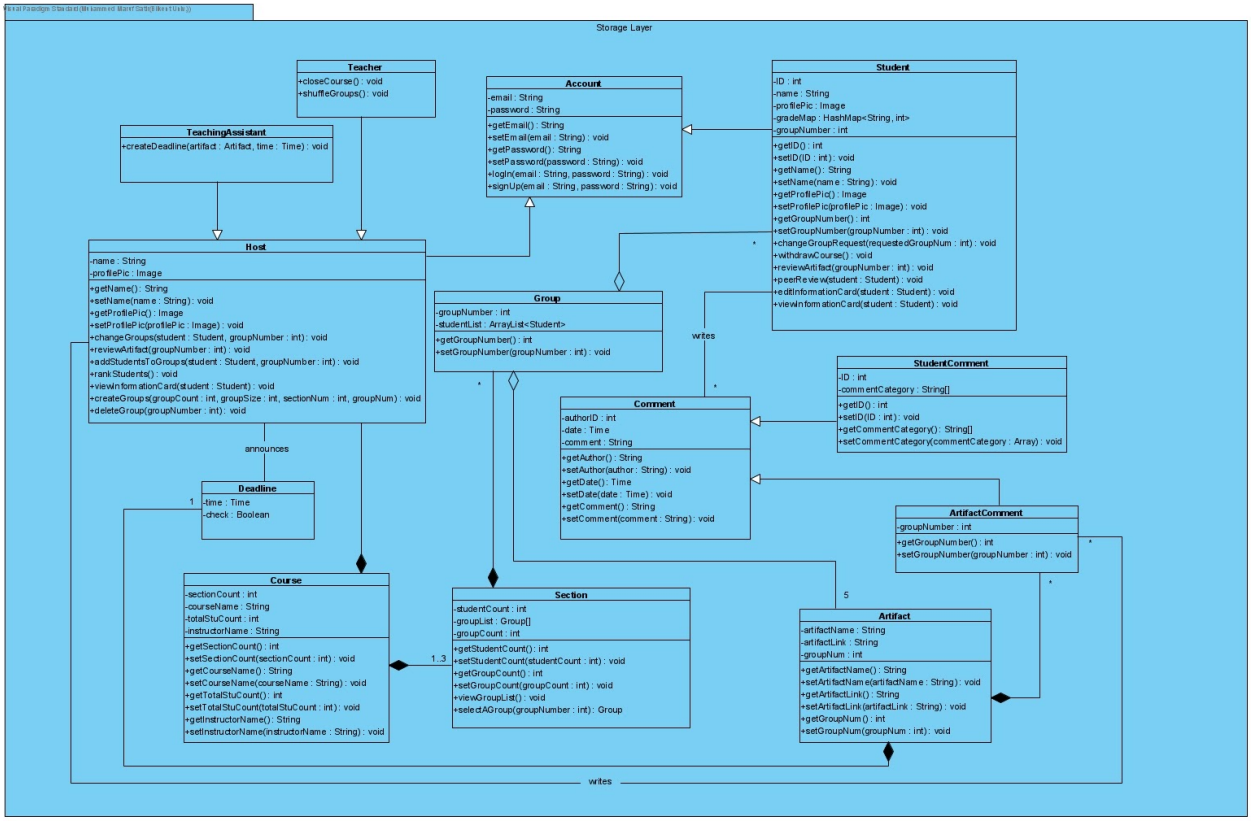


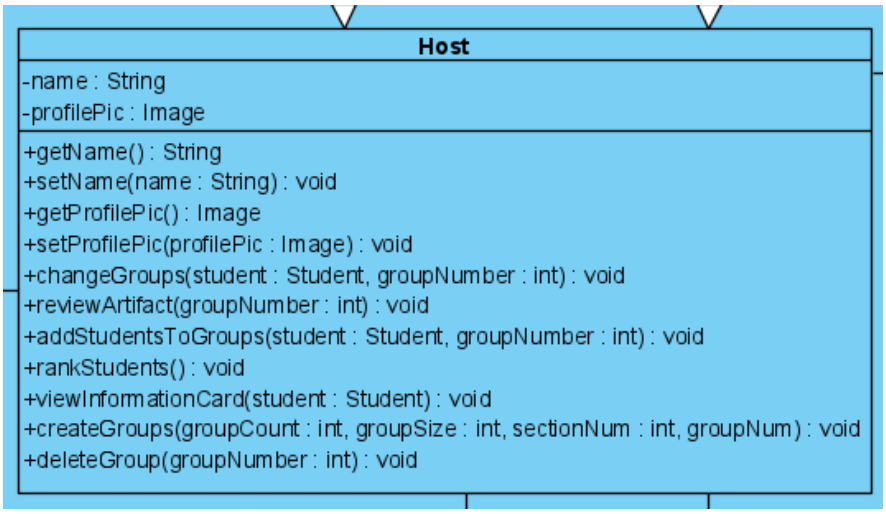
Class Name ArtifactCommentController
Attributes
• private ArtifactComment artComment: artifact comment object so that the controller can access artifact comment functions
Methods
• public void submitComment(String authorID, String groupName, String artifactName, String comment, Date date): submitting the new comment with parameters author's ID, name of the group, name of the artifact, date and the comments itself • public Comment[] getCommentByGroupName(String groupName): returns to the array of comments of a specific group • public Comment[] getCommentByArtifact(String groupName): returns the array of comments for single artifact • public Comment[] getCommentByGroupAndArtifact(String artifactName): returns the array of comments for single artifact of a specific group • void update(String authorID, String artifactName, String GroupName, Date date, String comment): updates the database with new comment • void delete(String authorID, String groupName, String artifactName): deletes the specific single comment. • public Comment[] getAllArtifactComments(): returns to the array of artifact comments



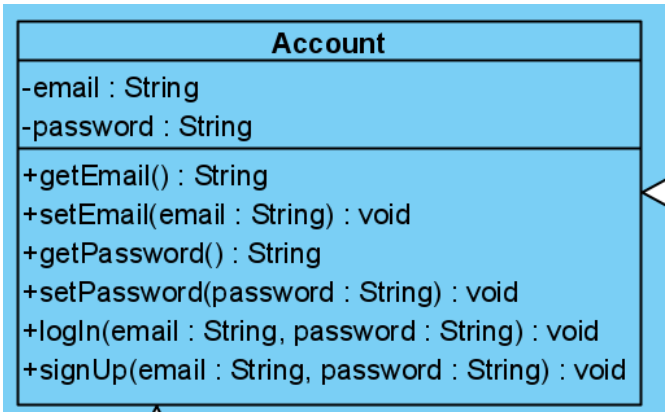
Class Name ArtifactCommentView
Attributes
• private ArtifactCommentController artComment : artifact controller object to communicate with the controller
Methods
• public void displayAllComments(): displays all comments of artifact • public void displayCommentByArtifact(String artifactName): display comments of the artifact • public void displayCommentsByArtifactAndGroup(String artifactName, String groupName): display comments of the artifact for a specific group • public void displayCommentsByGroupName(String groupName): displays comments for a specific group

3.1.3 Storage Layer

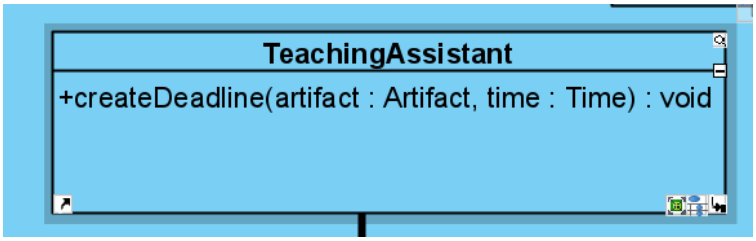




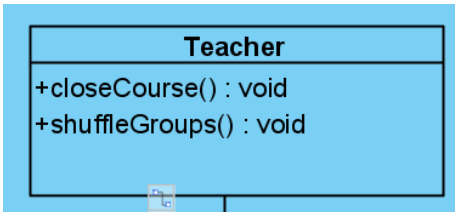
Class Name Host
Attributes
● private String name: holds the name of the instructor/TA ● private Image profilePic: holds the photo of the instructor/TA
Methods
● public String getName(): returns the name of the instructor/TA ● public void setName(String name): sets the name of the instructor/TA to the parameter ● public Image getProfilePic(): returns the photo of the instructor/TA ● public void setProfilePic(Image profilePic): sets the photo of the instructor/TA to the public parameter ● public void closeCourse(): terminates the course, the records of the groups and the students ● public void deleteGroup(int groupNumber): removes the group with the index of the parameter from the course ● public void createGroups(int groupCount, int groupSize): creates student groups as much as the parameter and sets the maximum capacity of each group to the second parameter ● public void shuffleGroups(): randomly assigns the students who aren't distributed in any groups yet ● public void changeGroups(Student student, int groupNumber): changes the group of the student in the parameter into the group in the parameter ● public void reviewArtifact(int groupNumber): displays the work done by the group of the number in the parameter ● public void addStudentToGroup(Student student, int groupNumber): adds the student in the parameter to the group whose index is in the second parameter unless the group size is full



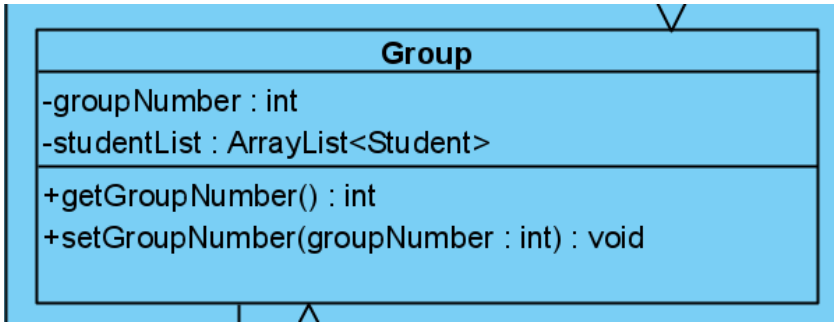
Class Name Account
Attributes
• private String email: holds the email address of the account • private String password: hold the password information of the account
Methods
• public String getEmail (): returns the email address of the account • public void setEmail (String email): sets the email address of the account • public String getPassword (): returns the password of the account • public void setPassword (String password): sets the password of the account



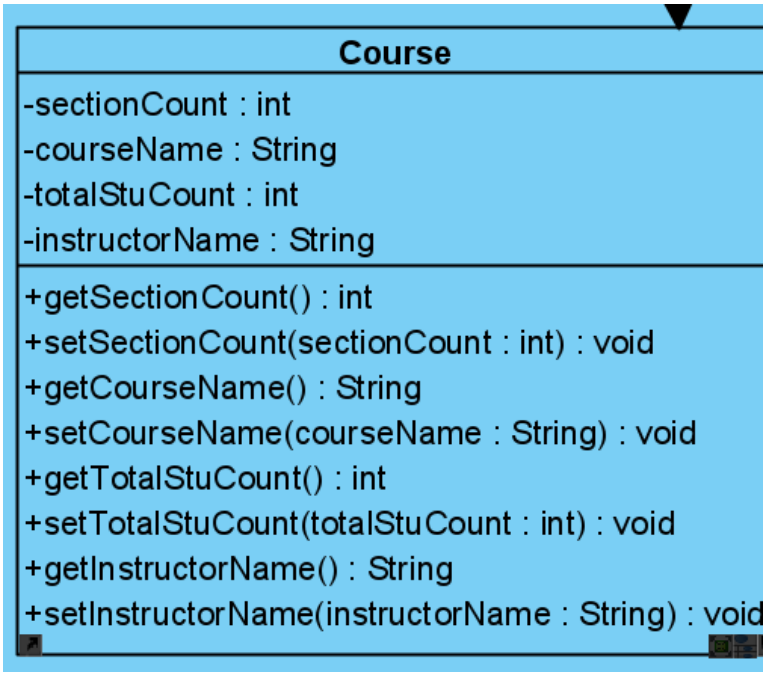
Class Name TeachingAssistant
Attributes
inherited from host
Methods
• public void createDeadline(Artifact artifact, Time time): create a deadline for a given artifact and given time



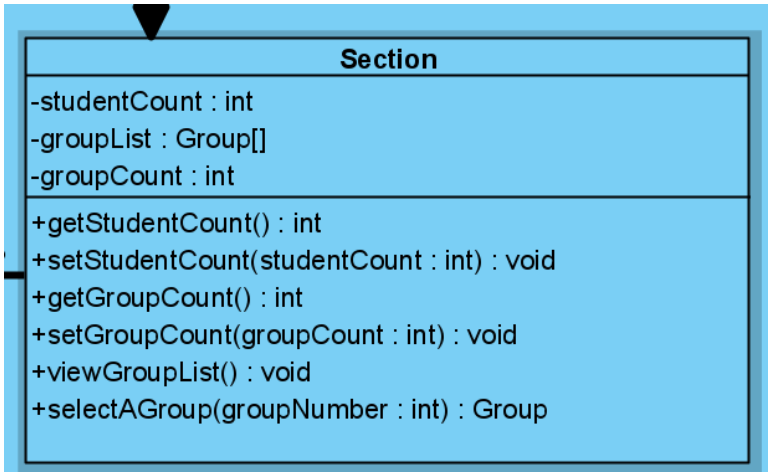
Class Name Teacher
Attributes
inherited from host
Methods
• public void closeCourse(): deletes all the information inside the database • shuffleGroups(): shuffle the students without groups to random groups



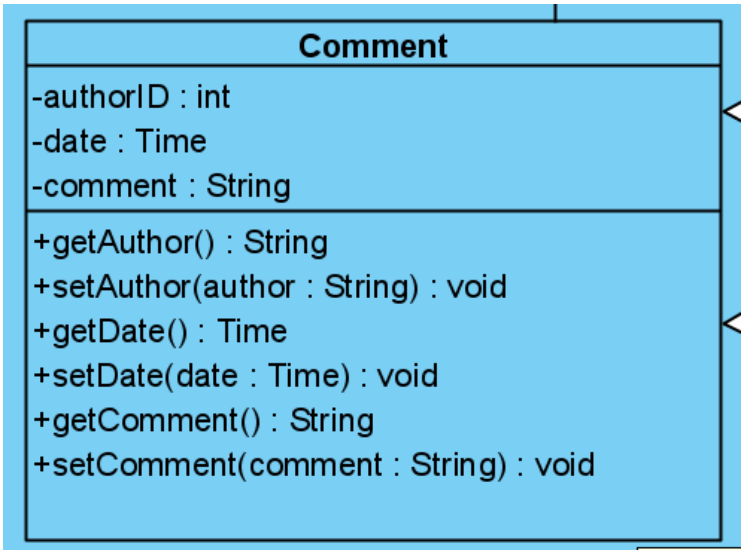
Class Name Group
Attributes
• private int groupNumber: the group number of the group object • private Array studentList: the students in the group as an array
Methods
• public int getGroupNumber (): returns the groupNumber parameter of the Group object • public void setGroupNumber (int groupNumber): sets the group number of Group object



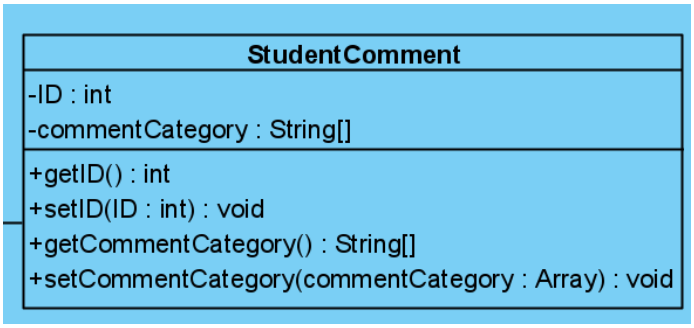
Class Name Course
Attributes
• private int sectionCount: number of the sections in the course • private string coursename: name of the course • private int totalStuCount: the total number of student in the course • private instructorName: the name of the instructor
Methods
set and get methods for the attributes.



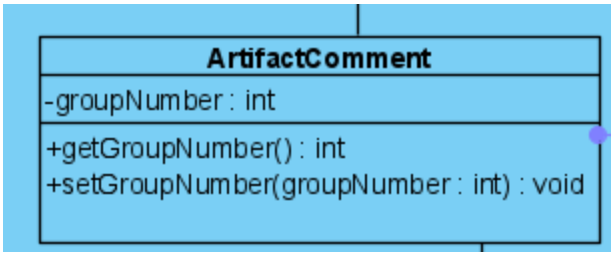
Class Name Section
Attributes
• private int studentCount: the number of students in the course • private groupList[]: the array of groups in the sections • private int groupCount: the number of groups in the section
Methods
• public Group viewGroupList(): returns the list of all groups • public void selectAGroup(int groupNumber): allows the user select a group depending on the parameter



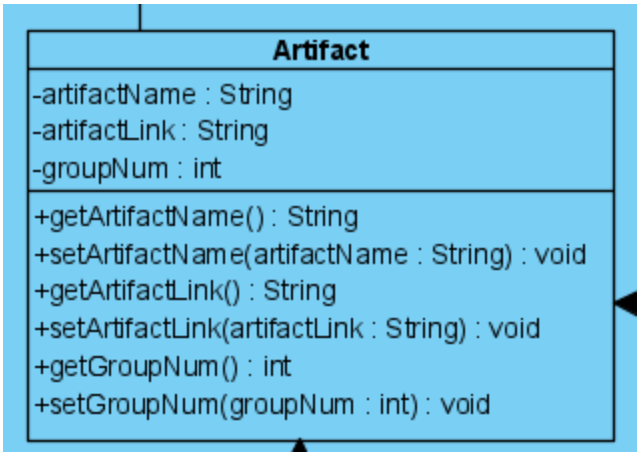
Class Name Comment
Attributes
• private String comment: holds the comment part of the Comment object • private Time date: holds the date when the comment is submitted • private int authorID: holds the author of the comment
Methods
• public String getComment (): returns the comment part of the Comment object • public void setComment (String comment): sets the comment part of the Comment object • public Time getDate (): returns the submission date of the Comment object • public void setdate (Time date): sets the submission date of the Comment object • public String getAuthor (): returns the author of the Comment object • public void setAuthor (String author): sets the author of the Comment object



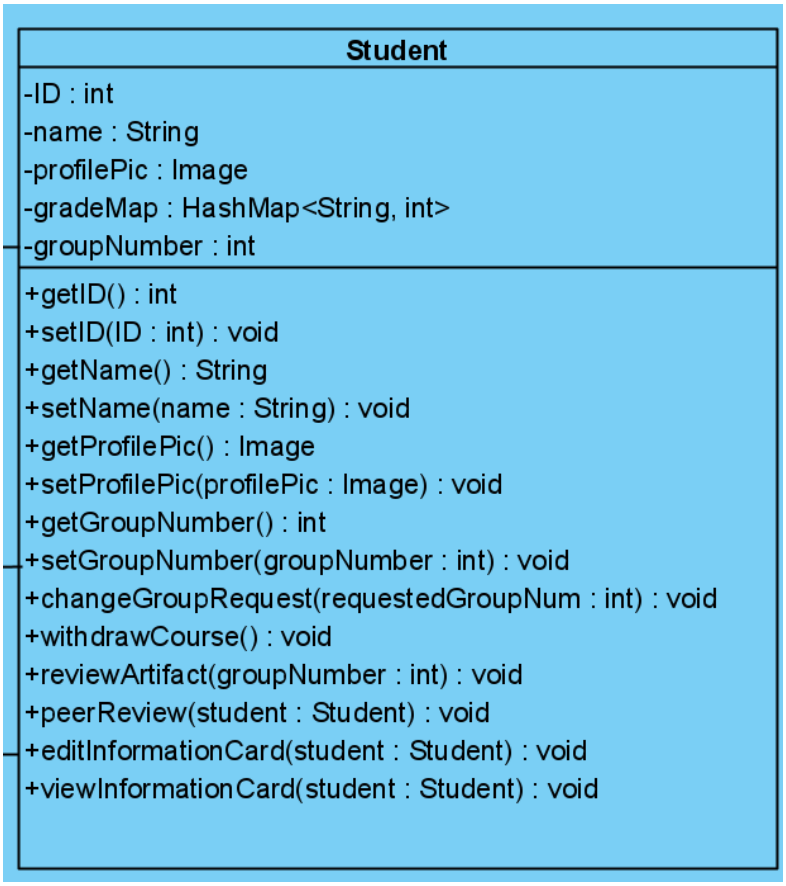
Class Name StudentComment
Attributes
• private int ID: holds the ID of the student who is target of the comment • private String[] commentCategory: holds the points given to the student by the peers (points given for communication, contribution etc.)
Methods
• public int getID(): returns the ID of the student • public void setID (int ID): sets the email address of the account • public String[] getCommentCategory (): gets the commentCategory of the studentComment • public void setCommentCateogry (String[] commentCategory): sets the commentCategory of the studentComment



Class Name ArtifactComment
Attributes
• private int ID: holds the ID of the student or host who writes the comment • private String[] commentCategory: holds the points given to the student by the peers (points given for communication, contribution etc.)
Methods
Attributes
• public int ID: holds the ID of the student or host who writes the comment • public String[] commentCategory: holds the points given to the student by the peers (points given for communication, contribution etc.)

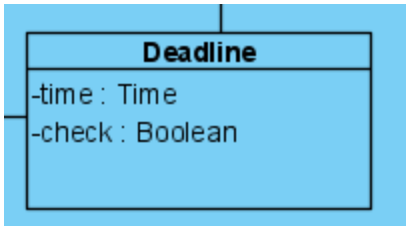


Class Name Artifact
Attributes
• private String artifactName: holds the name of the artifact. • private String artifactLink: holds the link of the artifact. • private int groupNum: holds the group number of the artifact.
Methods
• public String getArtifactName(): return the name of artifact • public void setArtifactName(String artifactName): sets the name of artifact • public String getArtifactLink(): return the link of artifact link • public void setArtifactLink(String artifactLink): sets the link of artifact • public int getGroupNum(): return the number of group • public void setGroupNum(int groupNum): set the number of group



Class Name Student
Attributes
• private ID: Holds the ID number of a student. • private String name: Holds the name of the student. • Private Image profilePic: Holds the image of the profile picture of the student. • Hashmap<String, int> gradeMap: Holds the grade of a particular student on the student. • int groupNumber: Holds the group number that the student is in.
Methods
• public int getID(): returns the ID of the student.

- public void setID(int ID): sets the ID of the player.
- public String getName(): returns the name of the student.
- public void setName(String name): sets the name of the student.
- public Image getProfilePic(): returns the image of the student.
- public void setProfilePic(Image image): sets the image of the student.
- public int getGroupNumber(): return the group number that student had been included.
- public void setGroupNumber(int groupNumber): sets the group number of the student.
- public void changeGroupRequest(int requestedGroupNum): student sends request to the host by indicating the requested group's number.
- public void withdrawCourse(): student requests the withdrawal of the system and system deletes his/her account from the system.
- public void reviewArtifact(int groupNumber): student review artifact of the particular other group.
- public void peerReview(Student student): Student gives other group member grades and comments on that grade.
- public void getGroupMembers(): returns the group members of the student who calls this method.



Class Name Deadline
Attributes
• private Time time: the time property of the deadline • private Boolean check: holds the condition whether or not the deadline passed
Methods

4. Improvement Summary

- Minor typo’s are fixed.
- Group number is added to the cover page.
- Unnecessary design goals are deleted.
- Trade-off part is taken to the object design phase.

Access Control Matrix: Access control matrix table is added to the “access control and security” section.

Low-level Design: Object Design section is added. There are three layers which are mentioned in the low-level design, UI Layer, Logic Layer, Storage Layer. All of the layers are added as a figure. Moreover, detailed descriptions of every class in the figures are added with class figures.