Bilkent University

Department of Computer Engineering

Senior Design Project

Project short-name: Peer Review System

Final Report

group-1f

Berke Ceran, Güven Gergerli, Lara Merdol, Akmuhammet Ashyralyyev, Sıla Saraoğlu

Instructor: Eray Tüzün
Teaching Assistant(s): Elgun Jabrayilzade, Erdem Tuna

Contents

1 Implementation Process

The beginning of the implementation process started after the first iterations. We divide the tasks of implementing entity classes, front-end and connecting the database to different teams. First, Lara and Akmuhammet started from the front-end part by using Sublime Text. Güven and Sıla implemented the entity classes according to uml class diagram by using Intellij and Berke researched and implemented the connection between a database and a system by using Intellij and MongoDB Atlas. First, we worked on our own repositories in Github. Moreover, Lara created a repository called CS319-f-1 and uploaded our reports to the Github. In addition to Github, as a group we all worked and communicated with each other from the Zoom communication platform. Then, after the second iterations the implementation process has sped up. Then, we divide the tasks again for different parts of the application. Berke and Sıla were responsible for implementing the database and controller class. First, they worked individually and implemented several methods. However, since implementation requires a collaboration, they worked from Zoom on one computer. For deciding on several functionalities and to speed up, Akmuhammet helped them also. For the front-end part, Lara, Güven and Akmuhammet worked on the front-end while the other team worked on the back-end. Akmuhammet also worked on binding the front-end and back-end with help of the HTTP requests. In the final week of the implementation, we all worked from Zoom. Sometimes, teams are selected and divided for some tasks. We all adapt Gann chart logic for finishing several tasks in a given time.

Furthermore, IntelliJ is used for the implementation of Java classes and Sublime Text is used for writing the HTML/CSS/Javascript part of the project. While we are using IntelliJ, firstly MongoDB (a global database) is linked with Java (with creating a project as a Maven Project). Then, with the help of the Spring boot and HTTP requests all front-end and back-end were connected to each other.

We try to use Github as much as possible in our implementation process; nevertheless, since one of our team members uses IOS and the maven project differs and sends errors in IOS and Windows, the others all got errors and conflicts which were hard to solve. Moreover, since we were doing the project online, when we push and pull we face conflicts in our codes (that we made different assumptions for different features), thus we connected in Zoom as much as possible and go through the same code to not to have any conflicts.

We also tried to stick to Object Oriented Programming as much as possible by using Entity classes in our functions. Moreover, we used design patterns to enrich our system of the application. In future sections, the details of that will be given.

The youtube link of the trailer is present in the project's repository is given in the .readMe file and it is also given as follows https://youtu.be/ywxCzDxk-yM.

At the end of the implementation process, we upload our project as zip file in the Github because the Github does not provide more than 100 files in a project.

1.1 System Requirements

Peeray (peer review system application) is designed as a web application. The application now only runs in the local host thus anyone who has a JVM (Java Virtual Machine), Java SDK and a Spring Boot can run the program. However, since it is designed as a web application, when it is moved to a server, anyone who has an access to an internet connection can access and use the system.

1.2 User's Install Guide

- Enter to the link below to see our reports and project
  https://github.com/LaraMerdol/CS319-f-1

- Clone the project by push command or downloading the zip file and run it in a Java compiler (we used IntelliJ thus it is recommended.)

- Then, run the CS319Application class to initiate the system and run the welcome_page.html to see the system in the computer's local host.

1.3 Build Instructions

The project has no website address since the binding of the frontend(HTML,CSS,JS) and backend(Java and MongDB) is done in spring boot and intellij so the program runs in the local host on the computer.
The backend of the project is done in MongoDB and the data of the project is stored in MongoDB Atlas. Frontend is written in HTML,CSS and Javascript. The project is compact and all the classes of backend and frontend are in the same folder. The backend uses HTTP requests to reach the data in the database. Firstly, the javascript classes request java controller classes of the pages and these java controller classes get the data from MongoDB Atlas. Any user who has the code of the system can generate the system in his/her local host. Nevertheless, the system is designed for a website thus the system can be transferred to a server where everyone with an internet connection can reach and use the system.

2 Changes and Improvements

If you look at the iterations of our reports, you can see that our classes were not properly correct and required changes. In our final object design in the second iteration of our design report, logic is the same in our implementation as well.

We have three packages called Entity and Controller, Database and Front-End. Moreover, we used a three tier architectural style that these packages interacted with only one level below of them. In these packages, we have classes with the same purposes. In our implementation, the Front-End package(first tier) only interacts with the package Controller -that includes the Entity and Controller classes-. Moreover, this package interacts with Database class. We tried to stick to the design as much as possible, only the DataController class was divided into several controller classes in our system and these controllers have the functionality of both InputController and DataController classes.

- One of the improvements in our system is first the instructor or TAs were adding the sections and sections of the students. Now to alleviate the load of responsibilities of instructors and TAs, admin of the system now addes and assigns instructors and TAs to sections. Also, admin enters the students to section as well.
- To get the sensitive information to the user, before the admin adds the users with specific roles to the system, all users should sign in with their information. After this process, all users can enter the system.
- Now, to see and review the artifacts the groups will upload their github link to the system.
- Navigations for groups, sections and students are added to almost every page to be user-friendly and give flexibility to users.
- The forget password feature is updated as sending a forget password request to admin by the link and admin will interact with the user to renew the user’s password. Yes, it does load admin a work; however, we try to choose the most secure option for our users.
- Since we are a system that is based on review features, it is important for us to get our users’ reviews of our system. For this purpose, the Contact Us feature is added to get the reviews. Also, it is useful for us to renew our system for newer functionalities in future versions.
- The system is now designed for a single course in one semester. After the semester is finished to close the course the system will delete all the information in the database, to delete the sensitive information of the users’ of previous semester in order to secure our users.

2.1 Design Patterns Used in the System

2.1.1 Adapter Design Pattern

In our project, we implemented the adapter design pattern. Our project is based on Maven project structure and maven project structure and contents are declared in pom.xml in our project implementation. However, displaying the content of the data in front-end (HTML - CSS - Javascript) XML format is not appropriate. The javascript files only work with JSON formatted objects. The solution we found is creating Adapter classes to convert the object so both parts of our project can understand each other. We created controller classes to work as an XML-to-JSON adapter for every function. These are the only classes which interact with our front-end.When an adapter receives a call, it translates the incoming XML data into a JSON structure and passes the call to the appropriate methods of the Javascript classes.

2.1.2 Model View Controller Design Pattern

In our project, we used MVC design pattern by separating our classes by their purposes to models, views and controllers. The model classes can be divided into business and data models. The model is represented by the entity package of our project.. The view is represented by our HTML, CSS and Javascript files and the controller is represented by our classes in the controller package. In our project, we tried to use a component based MVC approach by using HTTP requests to communicate between views and data model.

2.1.3 Data Access Object Pattern

The main logic of the DAO pattern is separating business logic of the system with the data access logic. Our data access logic is done in our class called mongoDB which interacts with the database itself. Our business logic is done in our package called front-end which contains HTML, CSS and Javascript classes. Our project adapts a specific data resource's access API (HTTP Requests) to generate the client interface.

2.1.4 Template Design Pattern

In our project, we used template design pattern in our question classes. First, we provide an abstract class of a question which gives a template to execute its methods and 3 subclasses open-ended, multiple-choice and point question that overrides these methods.

2.2 Design Trade-Offs

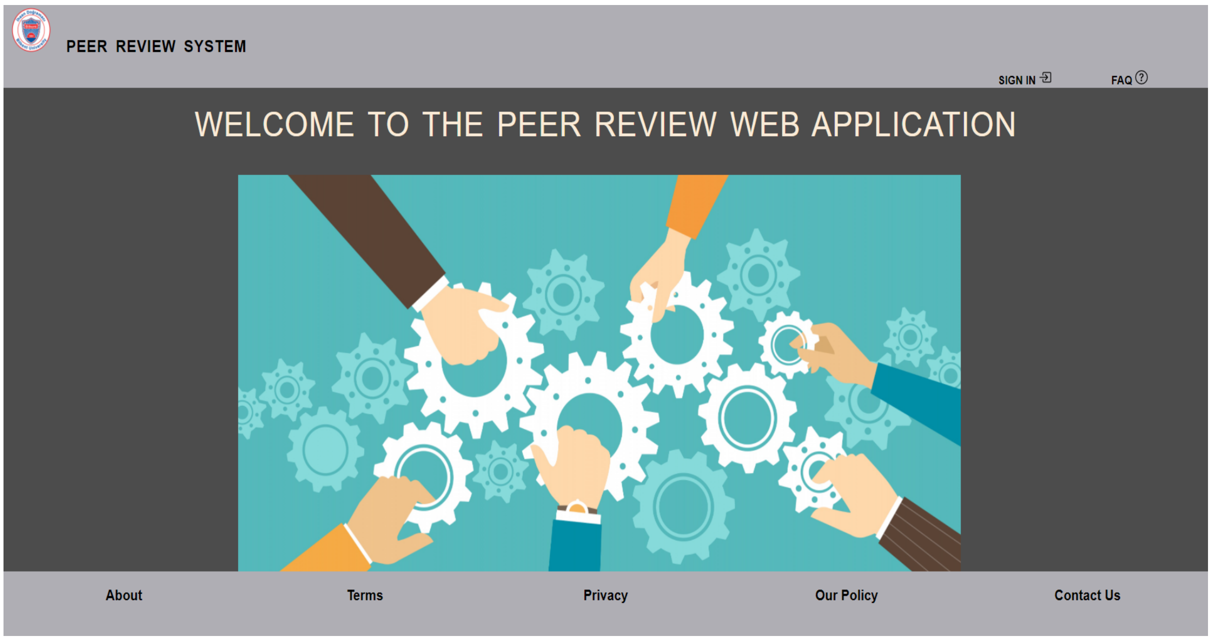
2.2.1 Number of Users Vs. Speed

The database in our project will always remain open. We have 60 methods in our database class that interact with the database and 16 different controller classes use these methods to connect the database. Therefore, the system can be used for fewer users with a lot more speed.

3 User's Manual

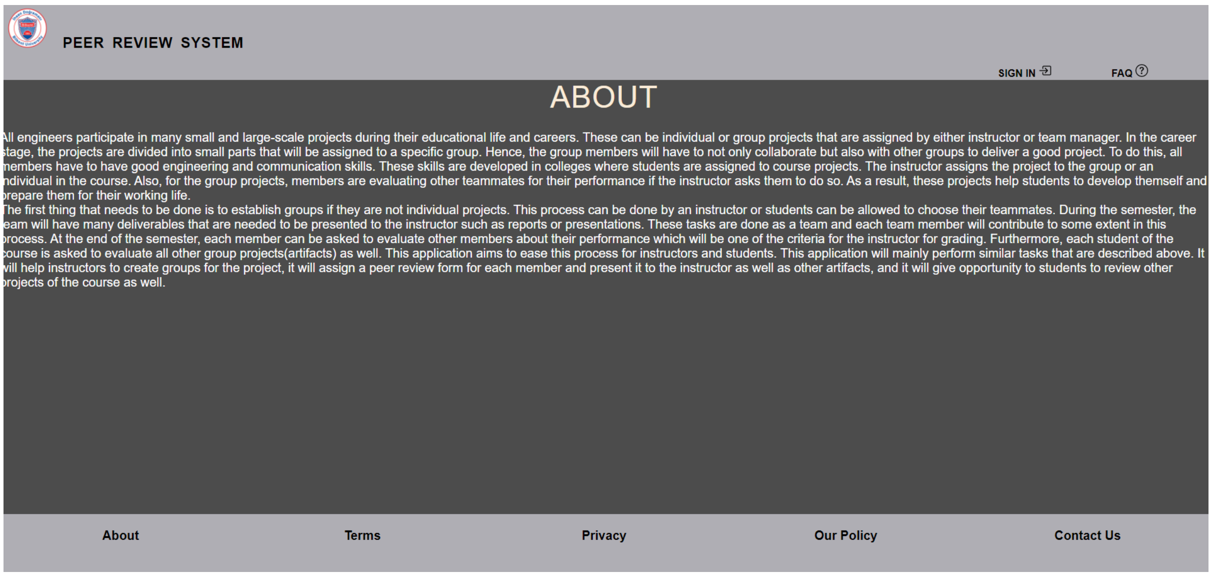
The system consists of 3 types of users that are Instructor , T.A and Student. The app generally focuses on group formations and review systems such as peer review, artifact review and course review.

3.1 Welcome Page



Welcome page is the start page of the Peer Review System Web Application, which consists of 7 clickable buttons and links; “About”, “Terms”, “Privacy”, “Our Policy”, “Contact Us”, and ”Sign In” which navigates the user to signup/login page and “FAQ button” which navigates to the frequently asked questions page which includes both a tutorial and answers to common questions about the application. The “About”, “Terms”, “Privacy” and “Our Policy” pages are similar to each other and contain different information about the application. Following About page is given as an example to visualize.

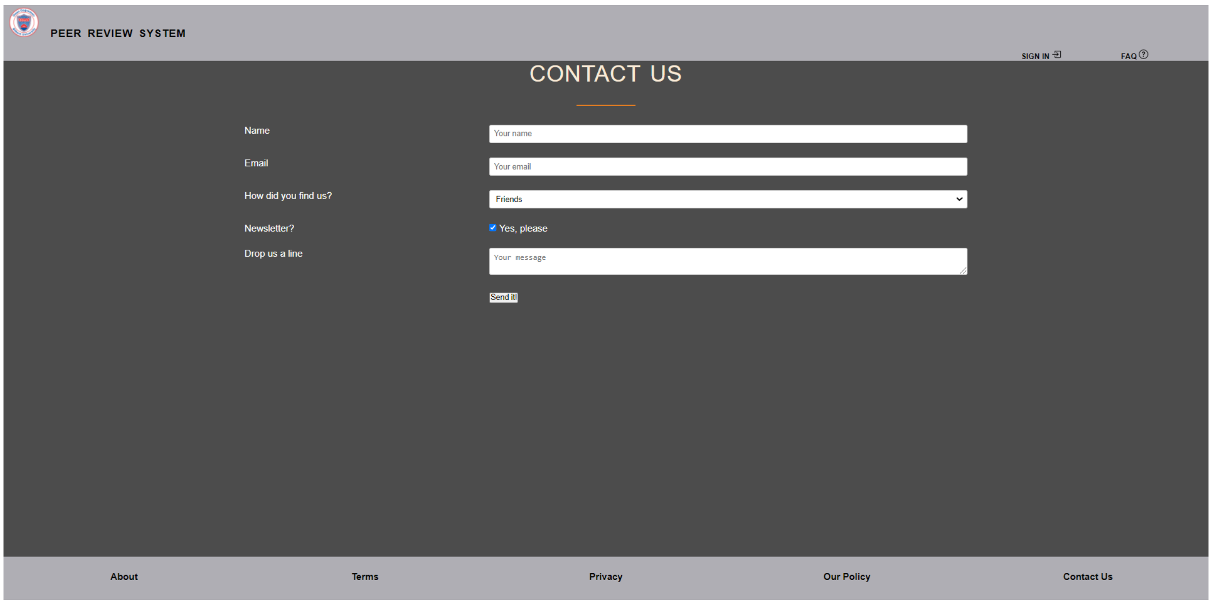
3.2 About Page



ll engineers participate in many small and large-scale projects during their educational life and careers. These can be individual or group projects that are assigned by either instructor or team manager. In the career tage, the projects are divided into small parts that will be assigned to a specific group. Hence, the group members will have to not only collaborate but also with other groups to deliver a good project. To do this, all nembers have to have good engineering and communication skills. These skills are developed in colleges where students are assigned to course projects. The instructor assigns the project to the group or an ndividual in the course. Also, for the group projects, members are evaluating other teammates for their performance if the instructor asks them to do so. As a result, these projects help students to develop themself and repare them for their working life.
he first thing that needs to be done is to establish groups if they are not individual projects. This process can be done by an instructor or students can be allowed to choose their teammates. During the semester, the eam will have many deliverables that are needed to be presented to the instructor such as reports or presentations. These tasks are done as a team and each team member will contribute to some extent in this rocess. At the end of the semester, each member can be asked to evaluate other members about their performance which will be one of the criteria for the instructor for grading. Furthermore, each student of the ourse is asked to evaluate all other group projects(artifacts) as well. This application aims to ease this process for instructors and students. This application will mainly perform similar tasks that are described above. It ill help instructors to create groups for the project, it will assign a peer review form for each member and present it to the instructor as well as other artifacts, and it will give opportunity to students to review other rojects of the course as well.

The About page briefly gives the description of the peer review system application.

3.3 Contact Us Page

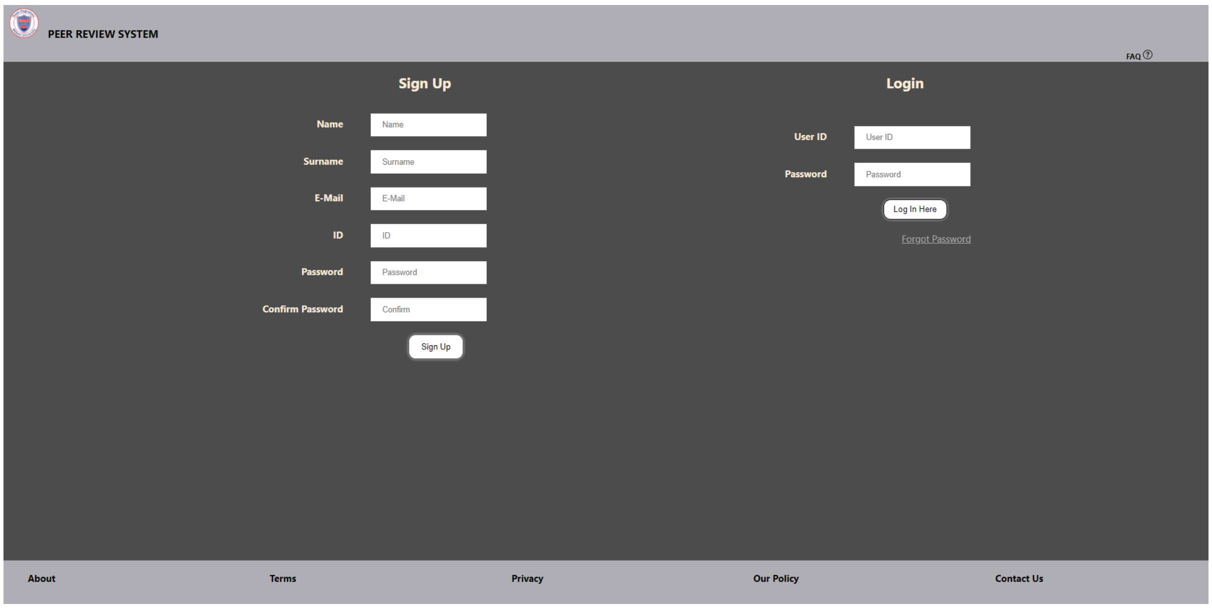


How did you find us? Friends

Newsletter? ✓ Yes, please

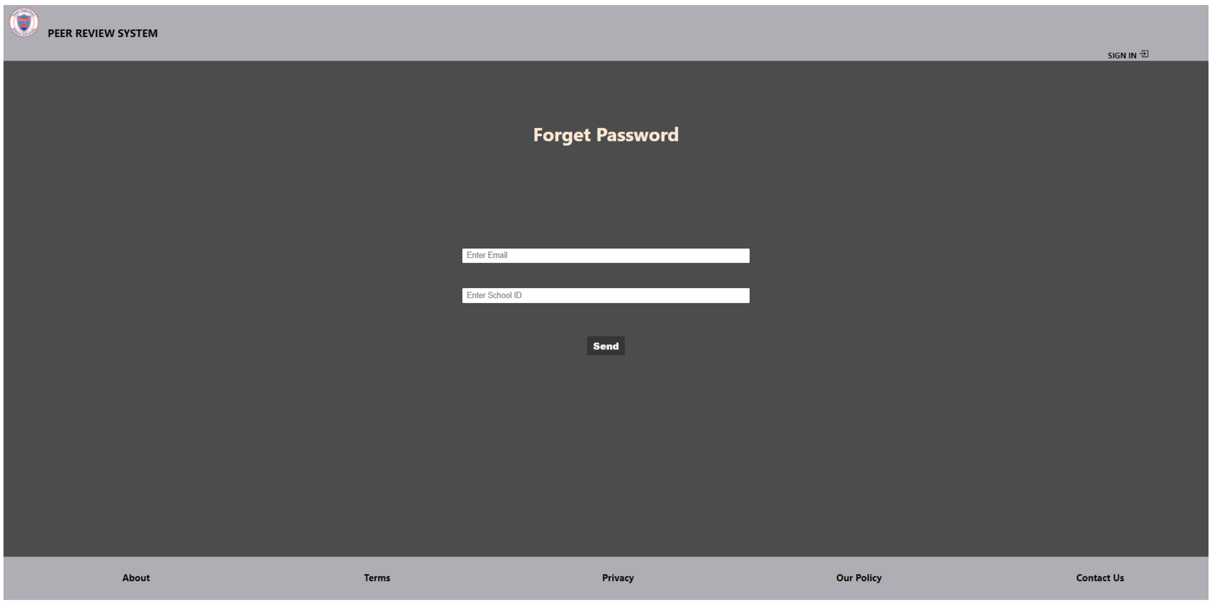
The contact us page is for getting users' reviews or to answer users' problems or questions about our system. It expects from the user: a name, email, the information about how the user finds us and the message from the user. The admin checks them from the system and contacts with the users individually.

3.4 Sign Up and Login Page



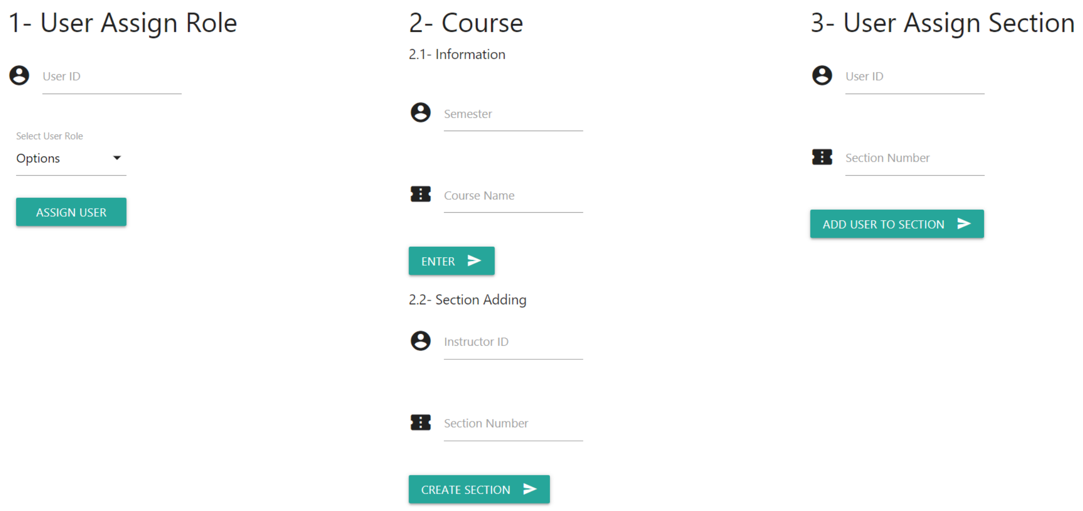
To enter the system first each user has to create his/her own profile / sign up in the left column of the page and enter his/her own sensitive information. To login to the page, each user has to be signed up before and give the sensitive information correct in the right column(login) of the page. If the user forgets his/her password, the user can click the link forgot password to get a new one.

3.5 Forget Password Page



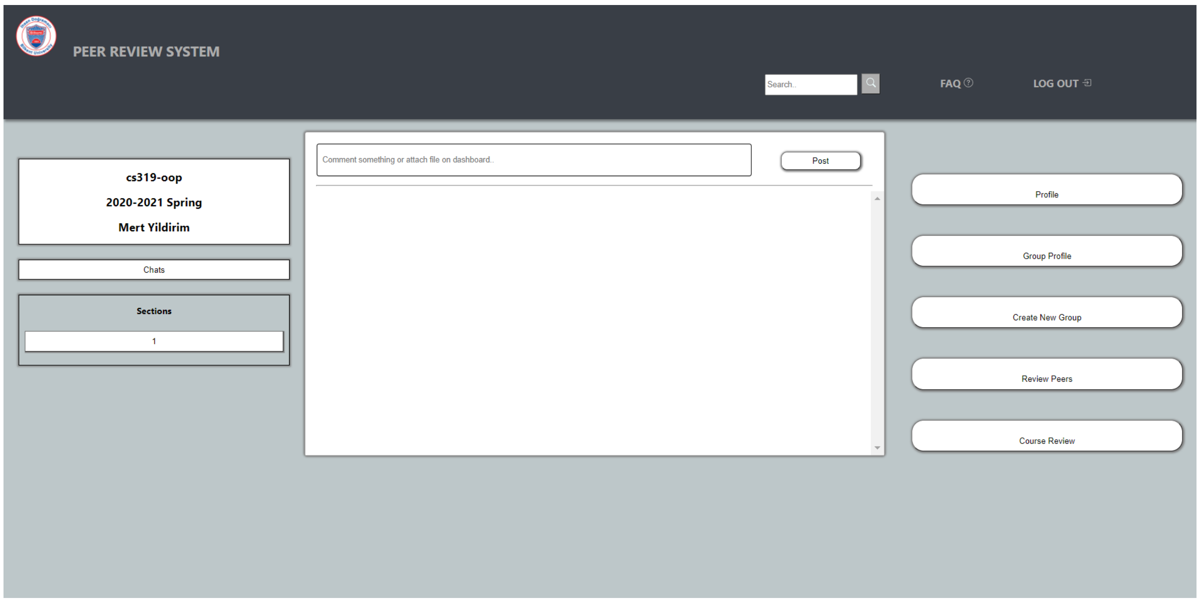
The forget password page accepts an id and an email from the user. After the validation of the user the admin contacts with the user to provide the most secure way to give a new password to the user.

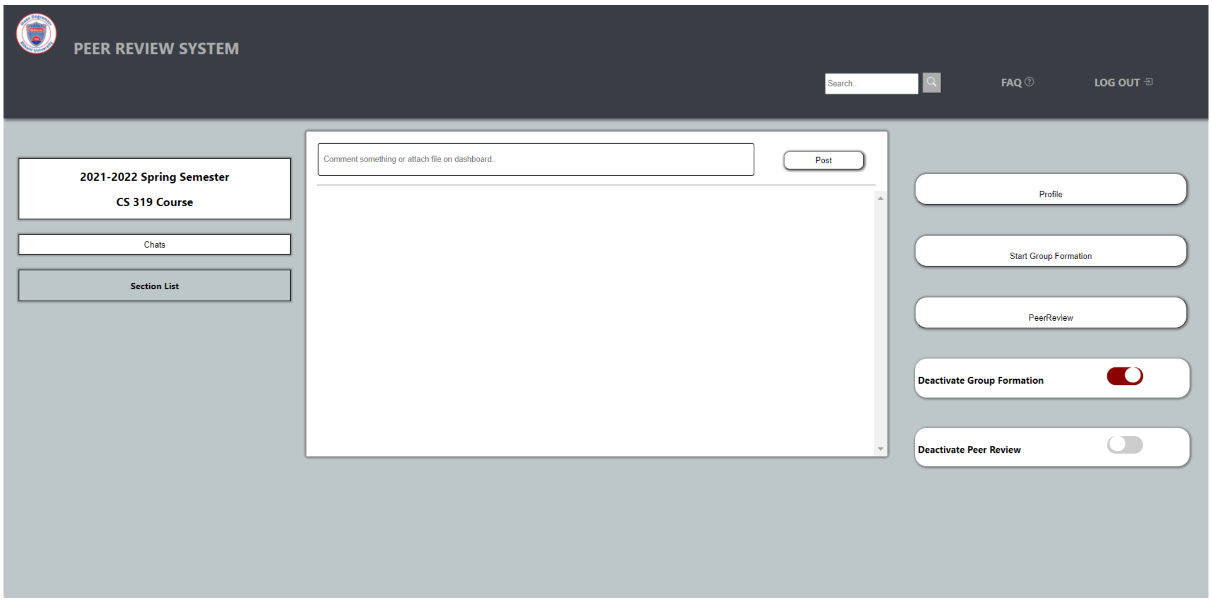
3.6 Admin Page



This page can only be seen by the admin of the system. This page loads when admin is logged to the system. The admin can assign a role such as instructor or TA to a user (1). The admin can enter the current semester and course information as well (2.1). The admin can add a section and assign the instructor to the section (2.2). The admin can assign users to the section as well then assigned users turn into students. The admin should enter this information after all users are signed up.

3.7 Dashboard Page

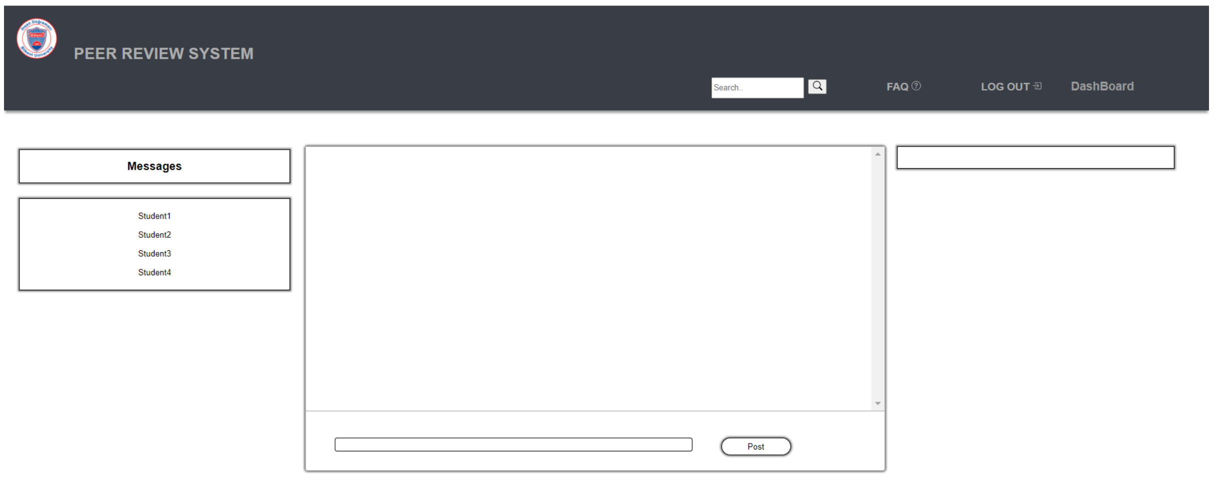




The dashboard page consists of 3 functional columns. The middle column in the dashboard is the main forum of the system where any user

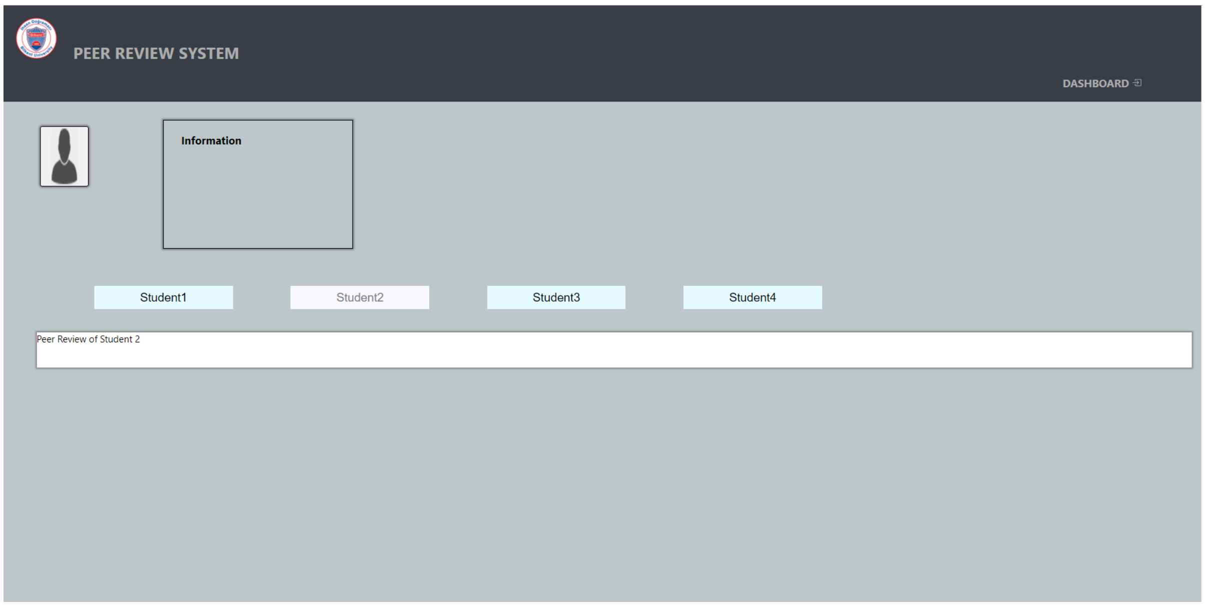
can post a message which can be seen by every user in the system and it is identical for every user in the system. The left column consists of the brief information of the user, the chats button that directs the user to the chats page, and the interactive section list where every section is a clickable button. When a section button is clicked, every group that the section has appears below the section list. The groups changes when the user clicks another section in the section list. Every group in the group list are also a clickable button that directs the user to the group profile of the group.
The left column differs by the type of the user due to different kinds of needed functionalities. A student has a “Profile” button, a “Group Profile” button, a “Create New Group” button which is visible after the instructor starts the group formation, and a “review peers” button which is visible after the instructor submits the peer review questions. An instructor/TA has a “Profile” button, a “Start Group Formation” button that starts the group formation period which can be deactivated with the slider below, and a “Peer Review” button which as well can be deactivated with the slider below.

3.8 Chat Page



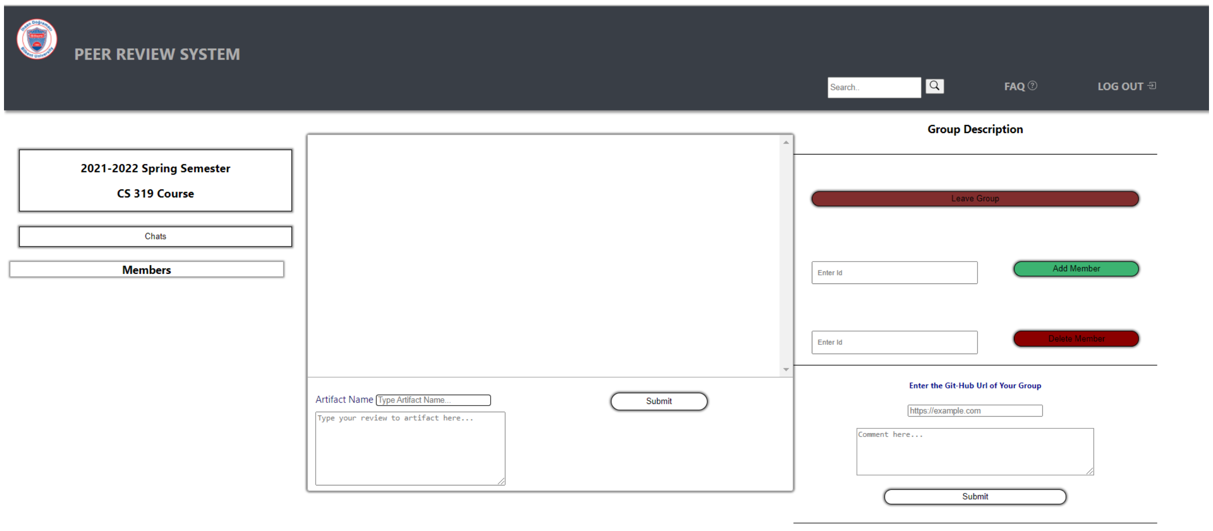
This page consists of the chats and messages between users. In the left column the user can pick another user to chat with and in the right column, the user can see brief information about the other user that is currently chatting with. In the middle column, the user sends and receives the messages with the selected user. With the dashboard button the user can leave the chat page whenever they want.

3.9 Personal Profile Page



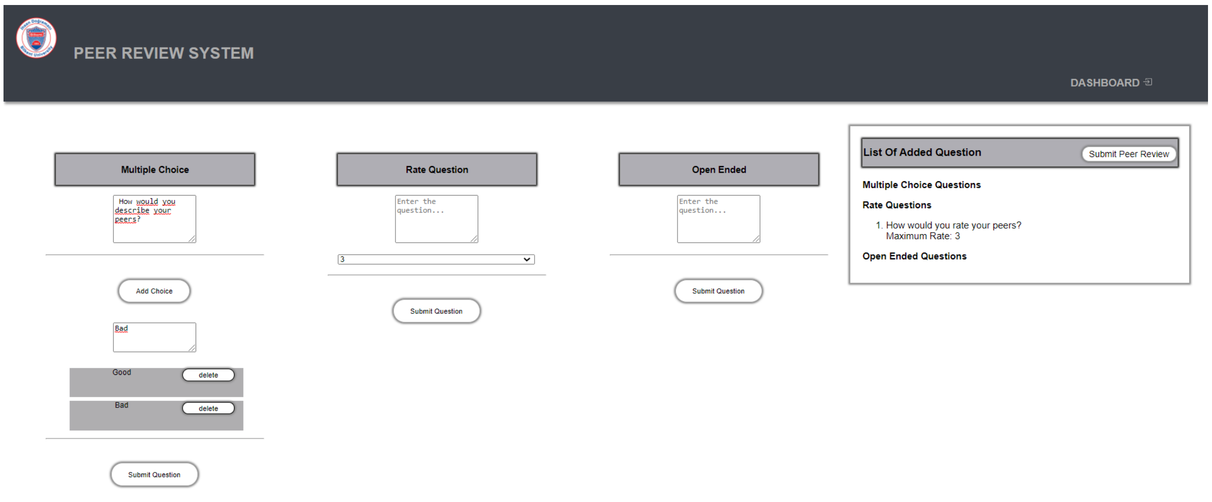
The personal profile page consists of several information of users such as: name, surname, e-mail, course name, semester, section number, and group name. If the user is an instructor only: name, surname, e-mail, course name and semester are shown. If an instructor or a TA visits a student’s profile, the information about peer review is shown below. The instructor can click any student in the peer reviews section to see what the filled peer review of the student. However, this sensitive information is not visible to other students.

3.10 Group Profile Page



This is the page of a group profile page that gives the information about the groups. First of all, this page can have 2 views: the view of a group member or the view of a student who is not a member of the group. For the first one, the members can load a github link to be seen by the other users to review their artifacts. The group members can also leave the group, add a member to the group or delete a member from the group also. For the second situation, the users can see the github link and review the artifacts at the bottom of the middle. On the left of the page, the group members will be shown.

3.11 Assign Peer Review Page



How would you describe your peers?

Enter the question...

3

Bad

Good

Bad

List Of Added Question

Multiple Choice Questions

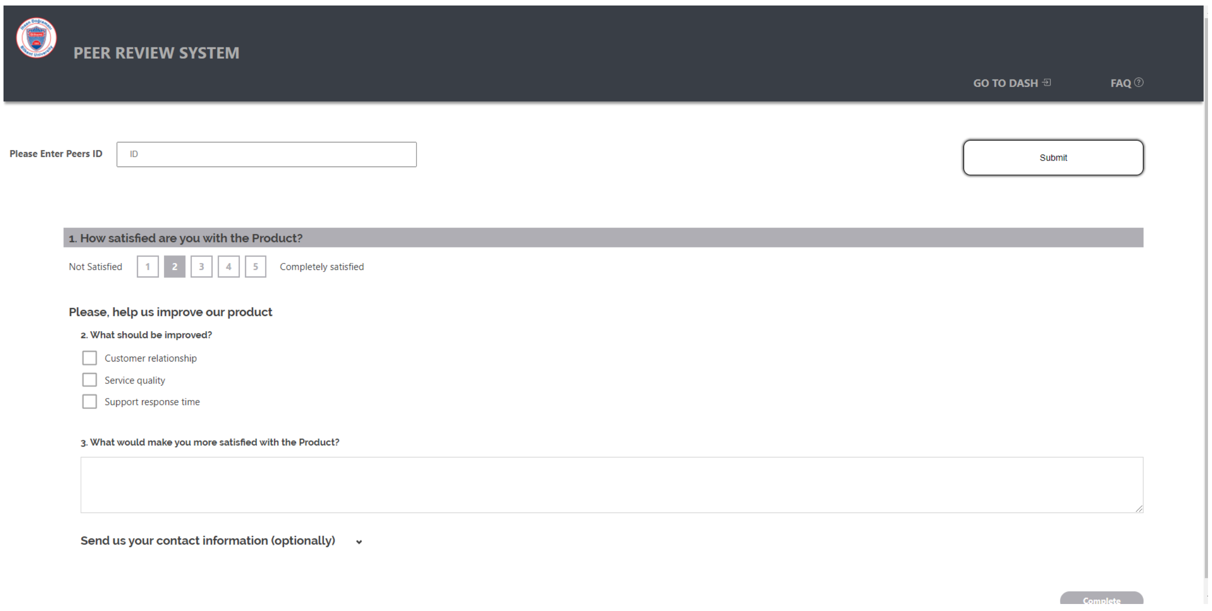
Rate Questions

1. How would you rate your peers?
   Maximum Rate: 3

Open Ended Questions

This is the page can only be seen by the instructor and the page functions for assigning peer review questions. The instructor can add 3 types of questions to the review questions. When the instructor, if the instructor clicks on the submit peer review button, the questions are sent to the students.

3.12 Peer Reviewing Page



Please Enter Peers ID: ID

1. How satisfied are you with the Product?

Not Satisfied 1 [✓ 2] 3 4 5 Completely satisfied

Please, help us improve our product

2. What should be improved?

- [ ] Customer relationship
- [ ] Service quality
- [ ] Support response time

3. What would make you more satisfied with the Product?

Send us your contact information (optionally)

The peer reviewing page is only accessible by student users. In order for a student to fill a peer review, first the student enters the peer’s student

ID. After the ID of the student is checked, the questions that are created and submitted by the instructor are displayed to the student. After the student fills the question, the student clicks the submit button and then submits the peer review through the submit button. With the dashboard button the student can leave the peer reviewing process anytime.

4 Project Process / Work Allocation

- Github Repository and ReadMe
    - Lara Merdol Initialized the Github Repository for our project.
    - All members of the group discussed and wrote the ReadMe file that contains the brief description.
- Analysis Report Iteration 1
    - Introduction and Proposed System are written by Akmuhammet, Lara and Güven. Requirements are written by Berke and Sıla.
    - Use case model is drawn by Berke and Sıla. Class diagram is drawn by Akmuhammed, Lara and Güven. All members participate in drawing the dynamic models.
    - Mainly Akmuhammed and Güven participated in drawing the Mock-ups.
    - All members go over the report at the end.
- Design Report Iteration 1
    - Introduction written by Lara.
    - System Architecture and Subsystem Services mainly written and diagrams drawn by Akmuhammed.
    - Final object design and the descriptions are made by Sıla and Güven, furthermore
    - Packages, Class interfaces and Entity Packages are written mainly by Lara and Berke.
    - All members go over the report at the end.
- Analysis Report Iteration 2
    - Lara, Güven and Sıla take part in the second iteration.
    - Some diagrams are modified and some diagrams are drawn from scratch, typos are corrected by Lara, Güven and Sıla.
- Design Report Iteration 2
    - Akmuhammed and Berke take part in the second iteration.
    - Akmuhammet revised the system decomposition and final object design. Berke added and revised information about the report.
- Final Report
    - Lara, Güven and Sıla wrote the Final Report.
- Implementation of Back-End
    - Akmuhammet, Sıla and Berke write entity and controller classes for backend in Java.
    - Berke took part in implementation of the database by using MongoDB and Sıla and Berke wrote the class of the Database.
- Implementation of Front-End
    - Güven and Lara take part in the implementation of the frontend by using Javascript, HTML and CSS. Akmuhammet binded the front-end with the back-end.