



Bilkent University

Department of Computer Engineering

---

# CS319 Object-Oriented Software Engineering

*BILTERN: Summer Training Course Management Application*

## Project Final Report

- Cenk Merih Olcay 22002408
- Arshia Bakhshayesh 22001468
- Enes Bektaş 22002401
- Faruk Uçgun 22003016
- Faaiz Khan 22001476

Instructor: Eray Tüzün

Teaching Assistant(s): Muhammad Umair Ahmed and Yahya Elnouby

# **Contents**

<b>1 Introduction</b>	<b>3</b>
<b>2 Lessons learned</b>	<b>3</b>
<b>3 User's Guide (User manual)</b>	<b>5</b>
<b>4 Build Instructions</b>	<b>43</b>
<b>5 Work Allocation</b>	<b>43</b>

## **1 Introduction**

The final version of Biltern provides different UIs and functionalities for students, teaching assistants, graders, coordinators, secretaries, and BCC admins. The key features and functionalities of the system mentioned in the requirements and design reports were implemented. Users are registered onto the system by the secretaries who are registered by the BCC admins. After the registration, the users can log in to Biltern and perform their required tasks.

The main feature of the system is report handling and how students can upload these reports onto the system for the teaching assistants to assess and the graders to grade. Furthermore, there is an interactive grading form that the graders will use to provide the final grading of a student. Another feature we added was notifications where users get information that affects them regarding the current state of the system.

### **1.1 The state of the system**

All of the required functionalities and features in the backend and frontend are implemented and connected, except:

- The functionality to automatically assign graders to students
- Getting reports from the database can be a bit slow sometimes, as they are saved in a blob data type that takes a while to transfer
- Grader Lists for the secretaries and coordinators
- Course completion element
- Third-party systems like Turnitin to check for plagiarism and some security system to implement 2-factor authentication.

## **2 Lessons learned**

As a group, working on this project, we have learned some valuable lessons that are sure to help us in our careers in the future. There was definitely a sense of accountability present in all the group members, which we learned was very important to be able to finish the project on time, by being accountable for the work allocated to each of us. Another lesson that we learned was about the importance of communication in a project like. The group members working on the frontend needed to be constantly communicative about what their requirements were and what endpoints they needed, whereas the backend group members needed to respond to these requests and communicate with the frontend members to provide them with the best possible solution to this. Another aspect of the importance of communication we learned was to not be afraid to ask for help and, conversely, not being afraid to provide the required help to your group members and teammates. There were plenty of instances in our project where someone got stuck, but through proper communication and willingness to help, we managed to complete the work in time.

In this course, as a group, we felt the emphasis put on the requirements report and the design report. Whilst we were skeptical of the importance of these at first, we gradually came to learn how useful it was to have written these reports. The reports outlined exactly what we needed to do as a group, and they gave us a framework to design our project on. The complexity of the project made it so these were absolutely necessary and without the reports, our jobs would have been a lot more difficult.

As far as the technologies and languages go, we used React with Javascript for the

frontend as well as HTML and CSS; and Java, Spring Boot, and MySQL for the backend. While almost all of us were familiar with the technologies we used, we still needed to brush up on them to make use of them efficiently. We learned how to use our time efficiently and not spend a long time on absorbing the knowledge.

### 3 User's Guide (User manual)

#### 3.1 Login Page for All Users

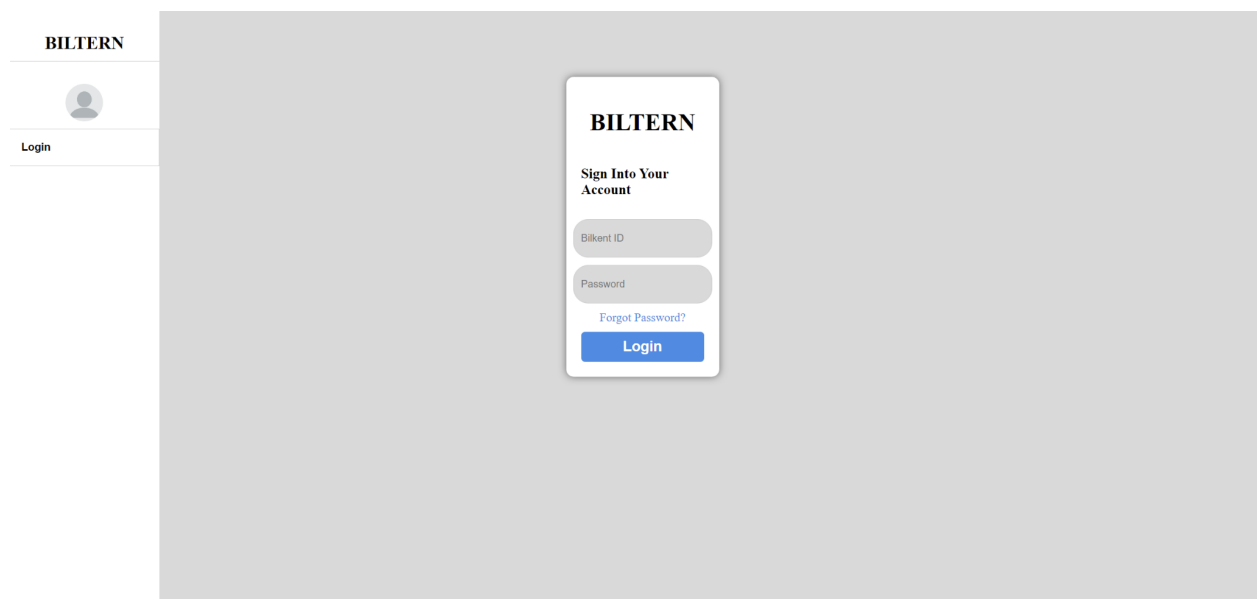


Fig. 1: Login Page

This is the login page that users are met with when they open a site for the first time in a session. From here, users can log in using their Bilkent ID and password for the system. If the credentials are entered incorrectly, the system will provide an error message as shown in the image below.

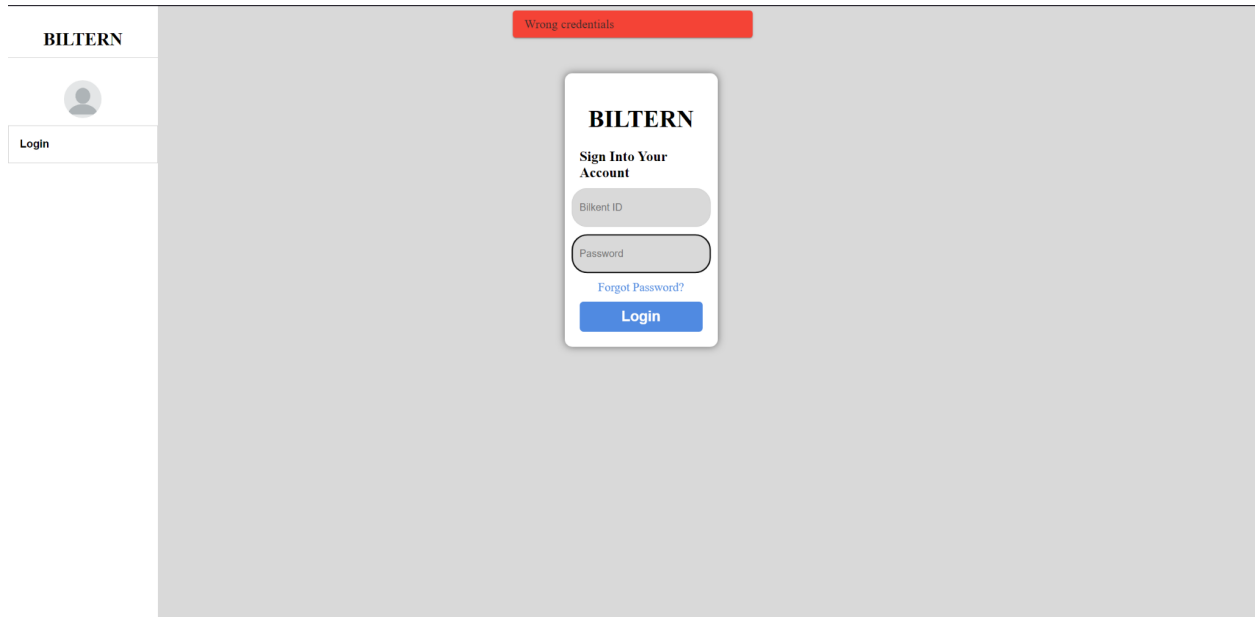


Fig. 2: Login Page Showing Error Message Due to Wrong Credentials

In case a user forgets their password, they are able to click the blue “Forgot password?” link present above the “Login” button.

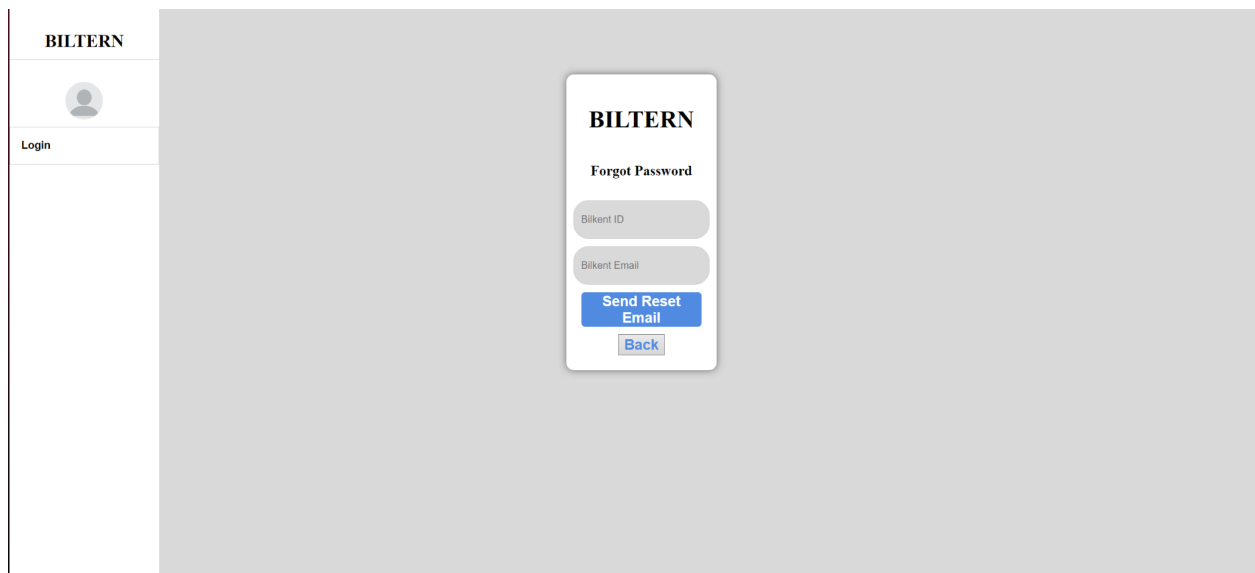


Fig. 3: Forgot Password Page

Once clicked, the user is redirected to the screen above, where they are prompted to

enter their Bilkent ID and Bilkent email address, after which a reset email is sent to the user containing their new password.

### 3.2 Dashboard for All Users

Once a user logs in, they are redirected to the dashboard. These dashboards contain different pieces of information depending on the user type. There is also a navigation bar present on every page that allows the user to navigate the pages of the system that allow them to perform their tasks.

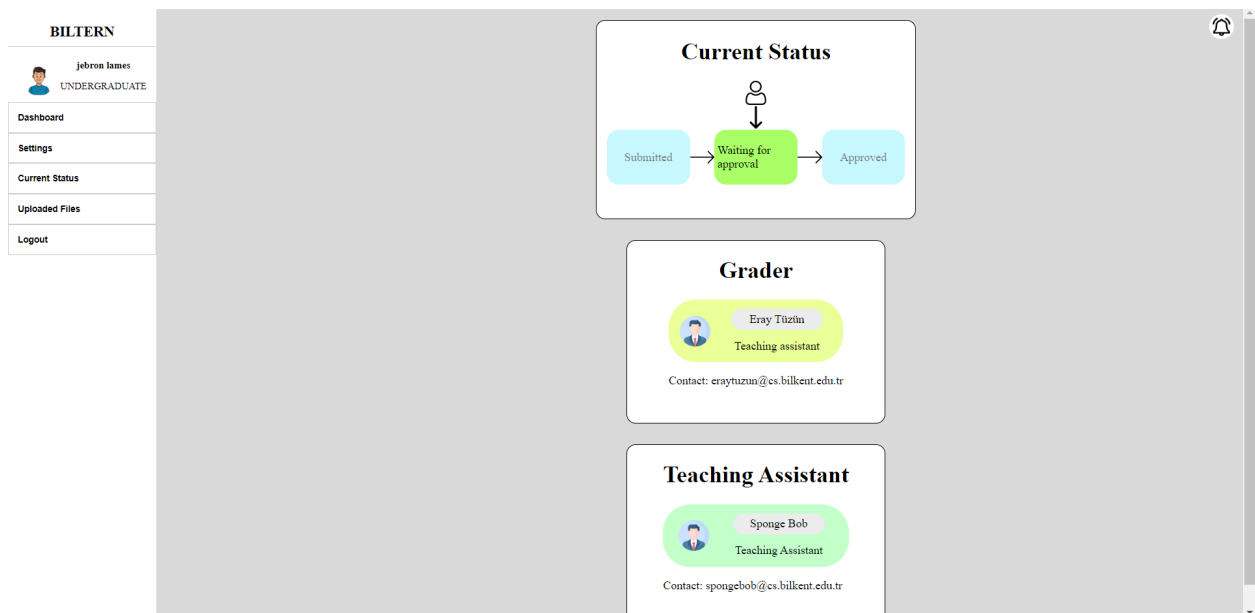


Fig. 4: Student Dashboard

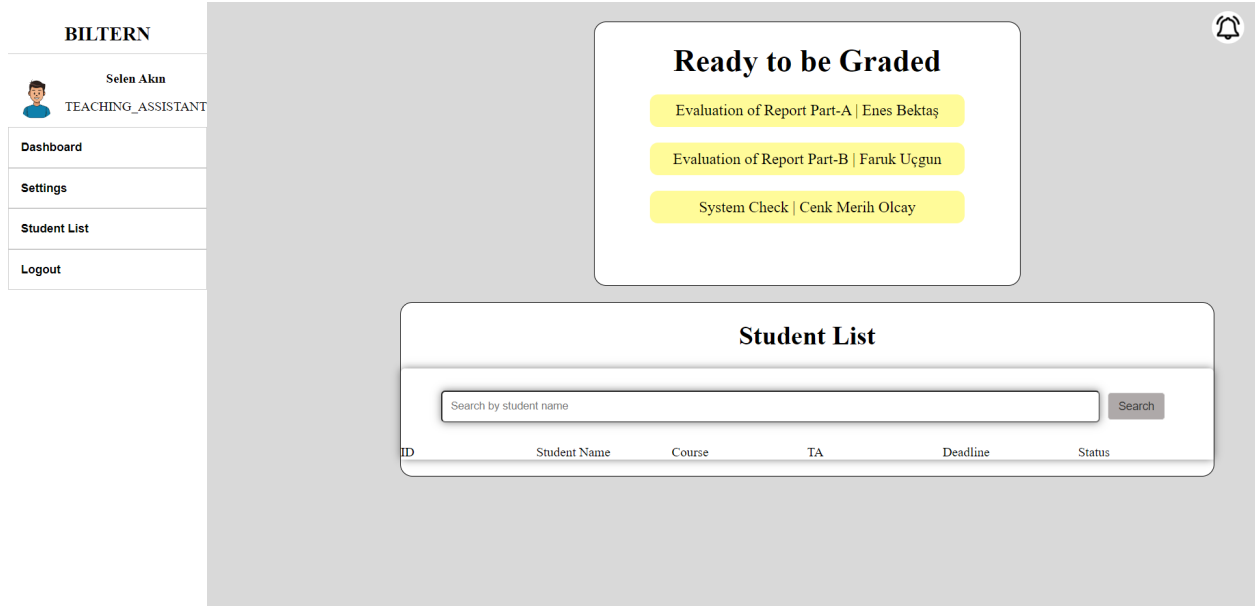


Fig. 5: Teaching Assistant Dashboard

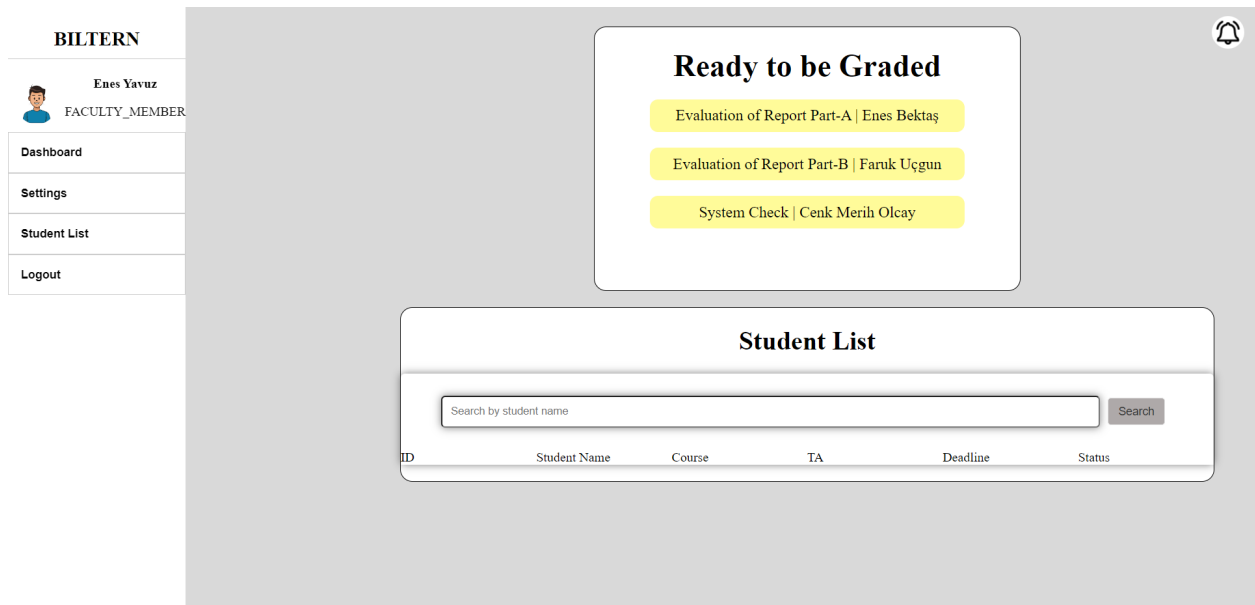


Fig. 6: Grader Dashboard



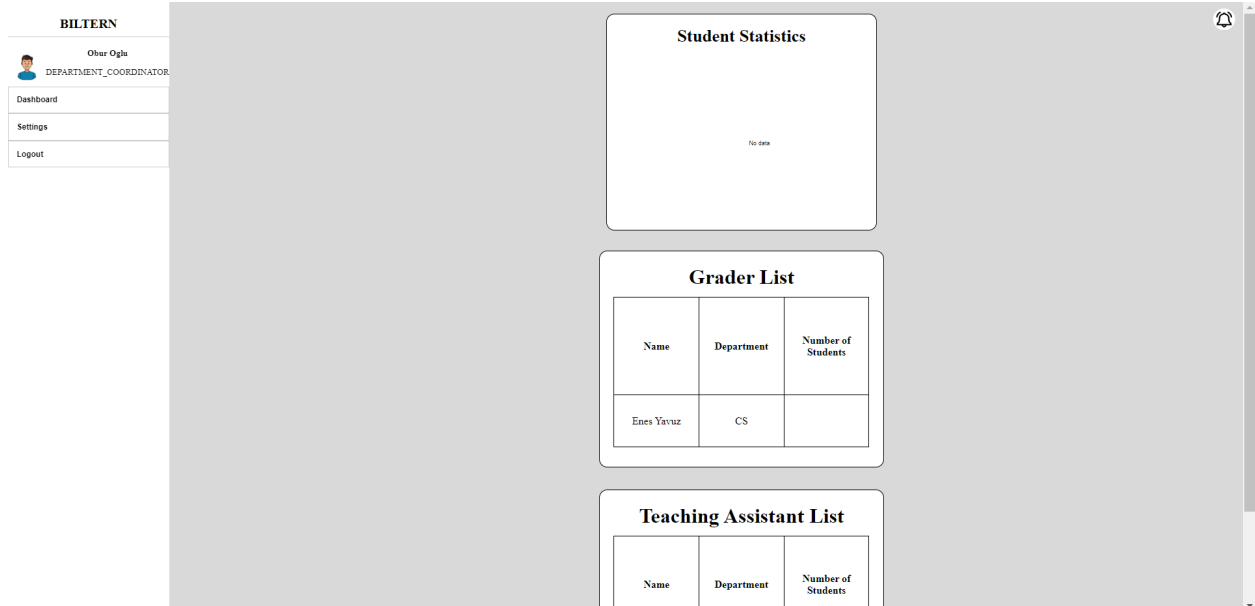


Fig. 7: Department Coordinator Dashboard

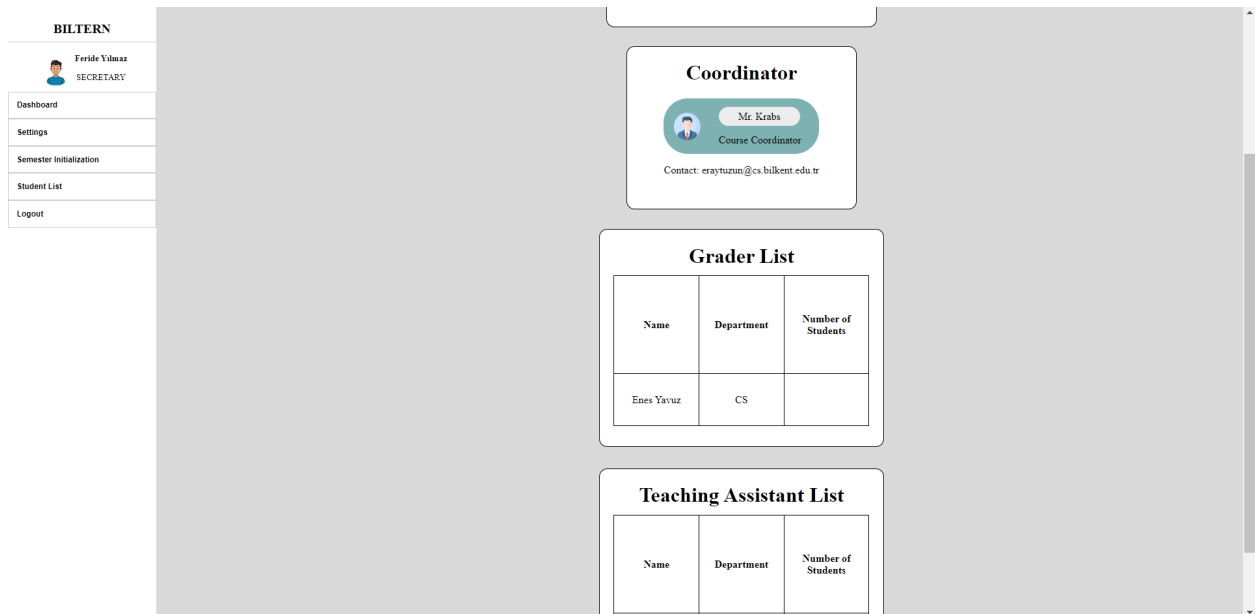


Fig. 8: Secretary Dashboard

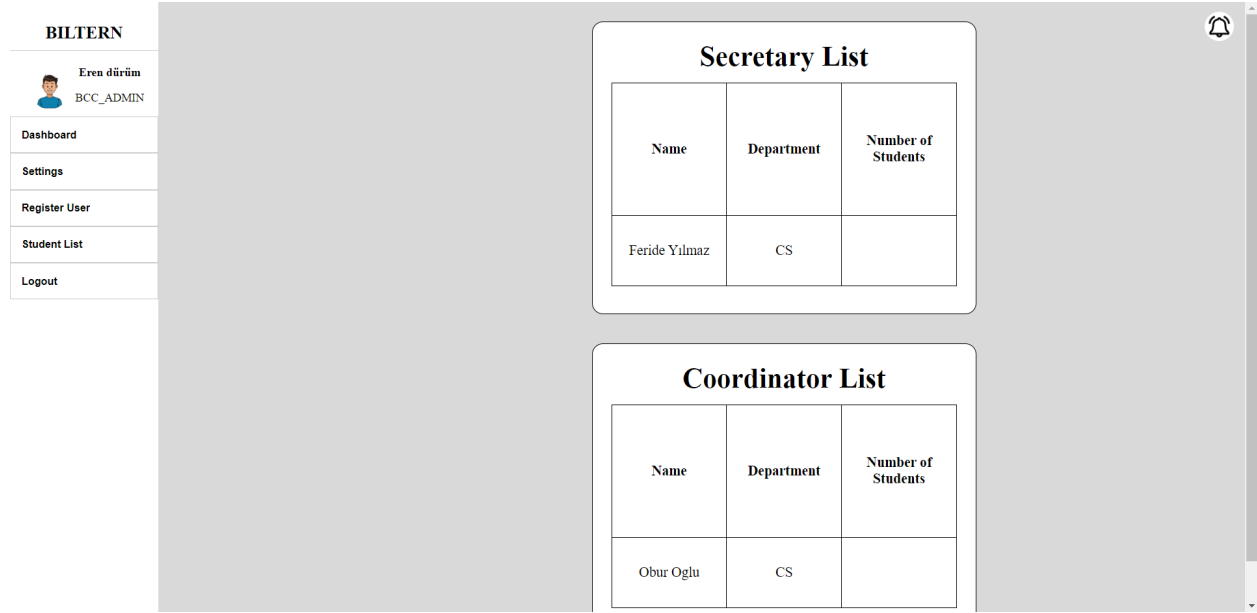


Fig. 9: BCC Admin Dashboard

### 3.3 Settings Page for All Users

From the navigation bar, a user can go to the settings page where they can change their password if they want to. They do this by filling a form containing fields for ID and current password. Once the fields are filled correctly, the user may send a request to get their password changed.

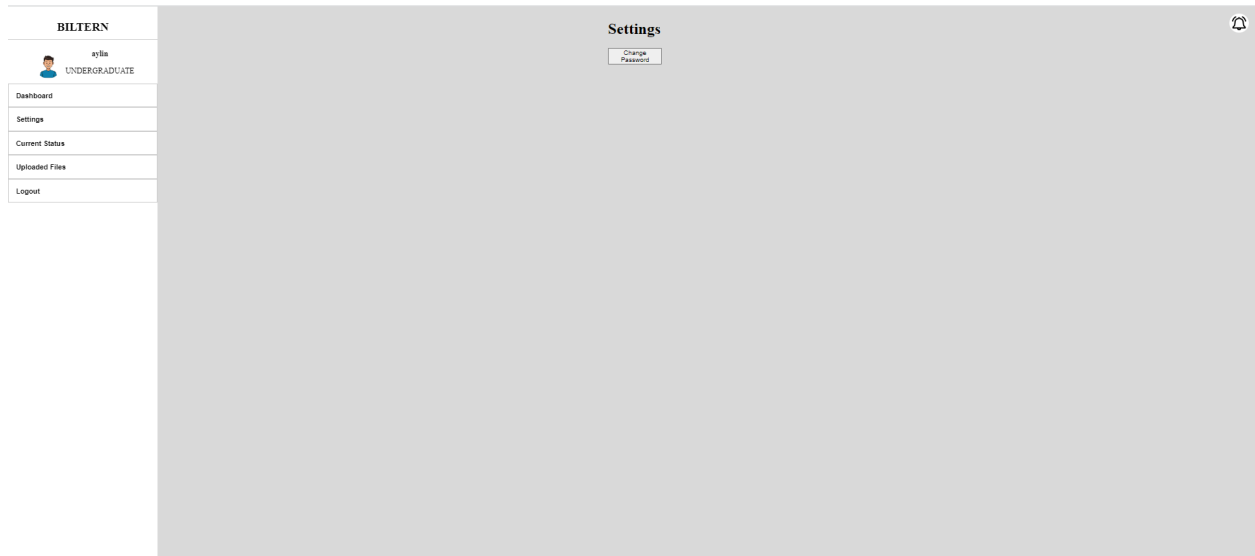


Fig. 10: Settings Page

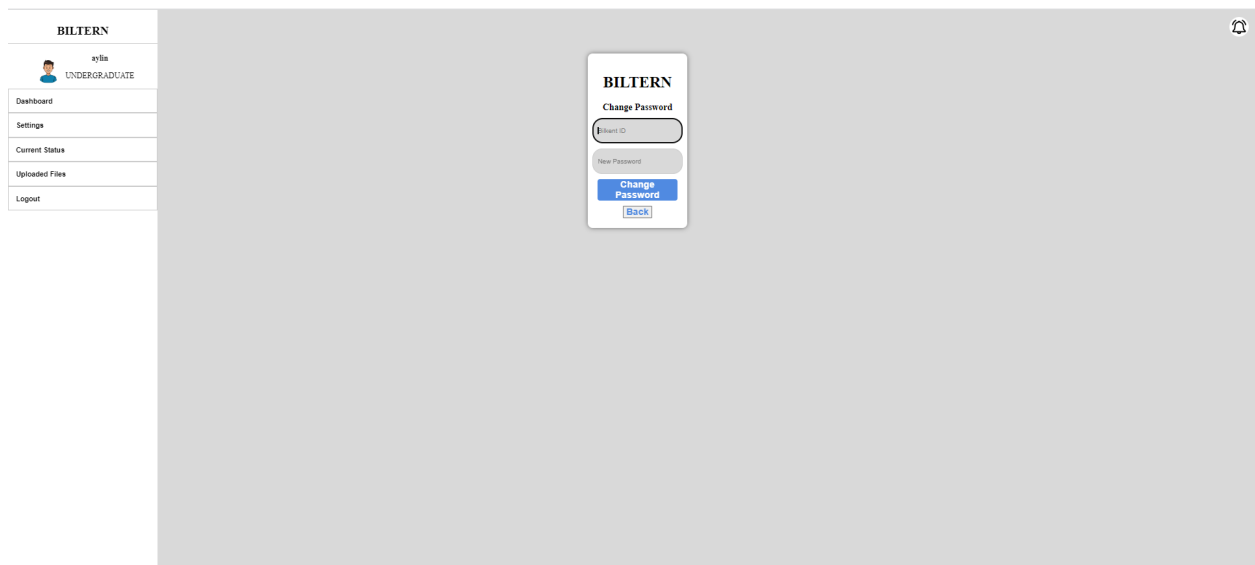
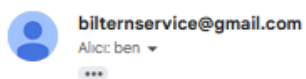


Fig. 11: Change Password Functionality



Your password change link is below

[Click](#)

Fig. 12: Sample forgot password mail

### 3.4 BCC Admin

The admin's credentials are hard-coded into the database, and these will be used to log in to the system. Due to this, there is no functionality to create admin accounts on the system.

From the dashboard, the admin can go to two pages using the navigation bar on the left: "Register User" and "Student List".

On the Register User Page, the admin can create a user account and add it to the system by filling out a form containing the required information. This in turn will send the user an email, informing them that they have been registered to the system.

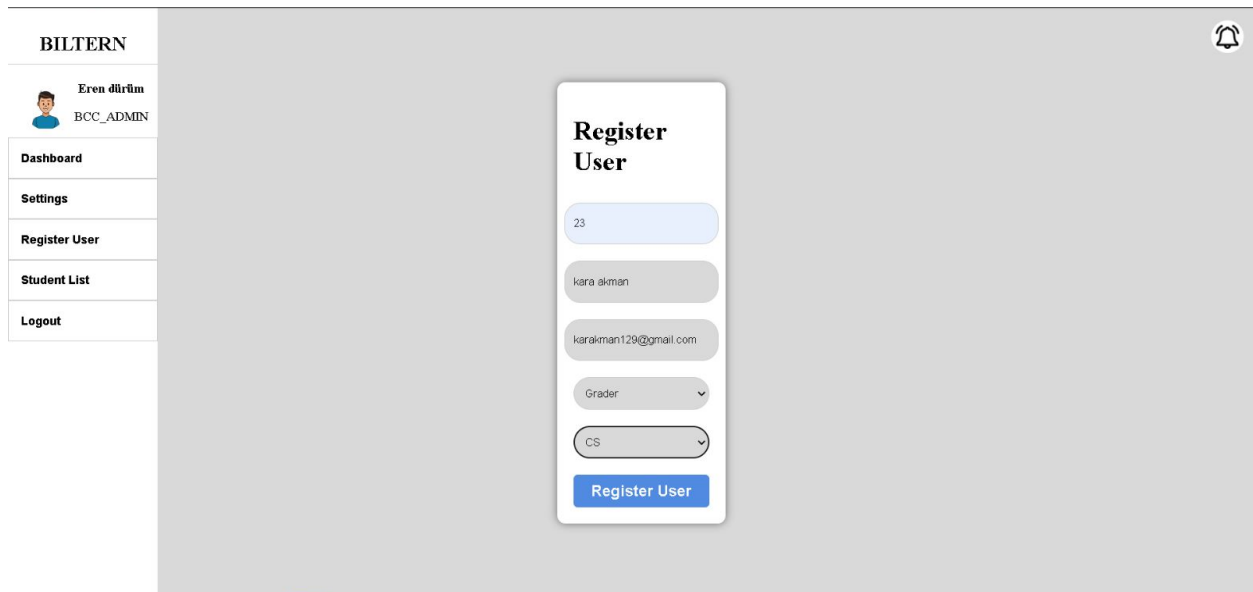


Fig. 13: Register User Page

On the Student List Page, the admin can view a list of the current students registered to Biltern. The rows of students on this page are interactable and will direct the admin to the Current Status Page for the student (see Section 3.9).

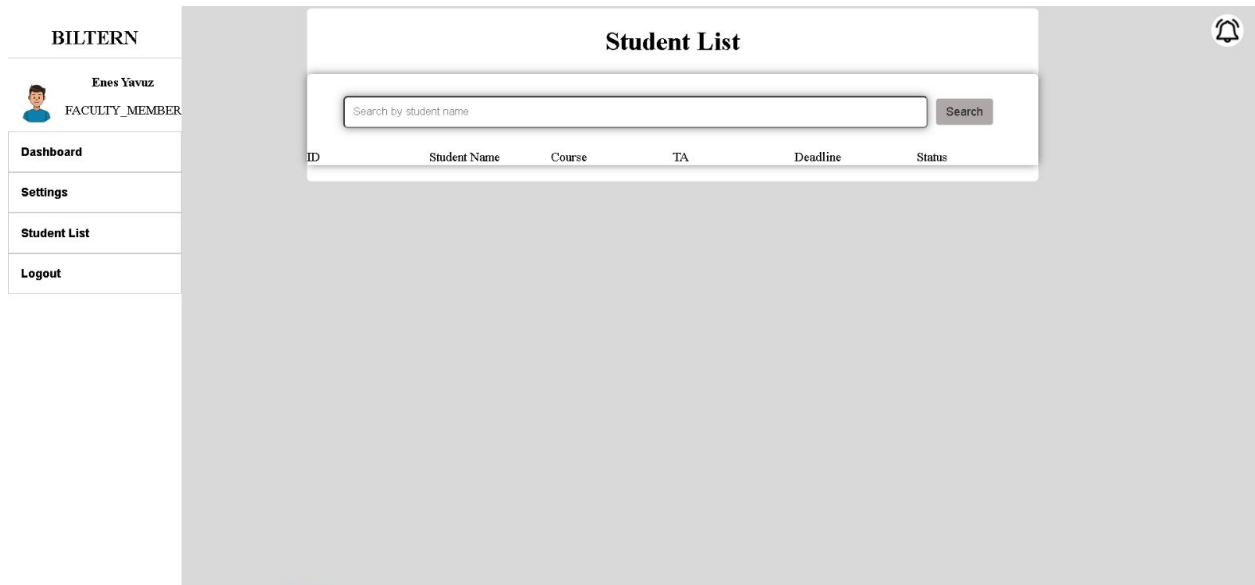


Fig. 14: Student List Page

### 3.5 Secretary

The secretary accounts are created by the admin. Once the account is created, a secretary can login and go about their tasks as required.

From the dashboard, the secretary can go to three pages using the navigation bar on the left: "Semester Initialization", "Student List", and "Register User".

The Semester Initialization page prompts the secretary to provide an excel file of users and their correct information, through which the secretary can register multiple users using one excel file.

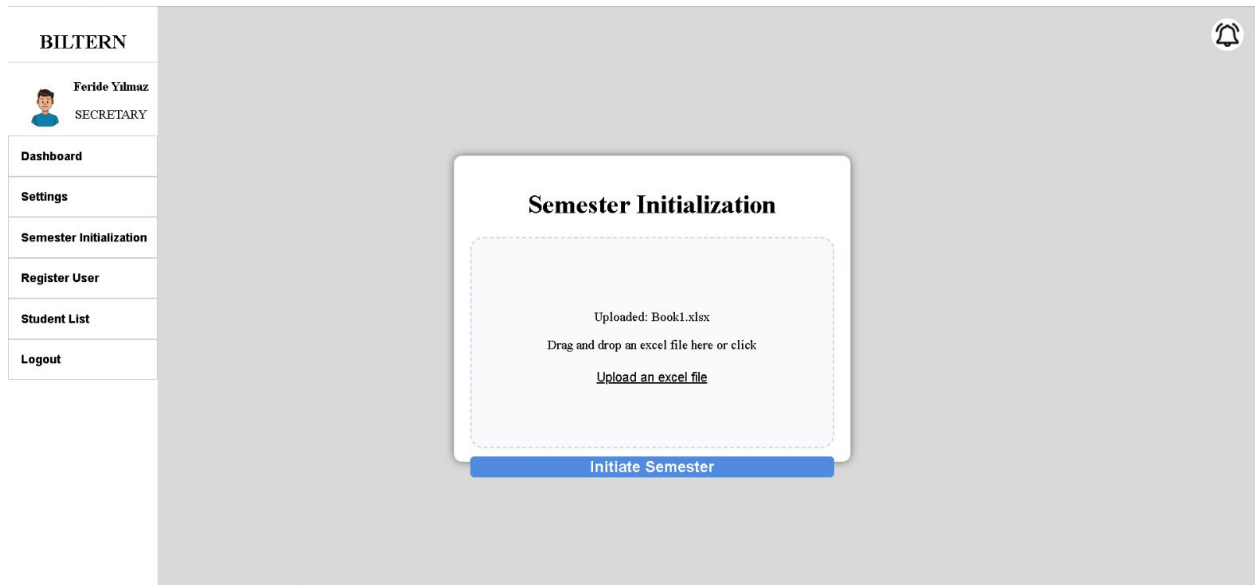


Fig. 15: Semester Initialization Page

On the Student List page, the secretary can view a list of the current students registered to Biltern. The rows of students on this page are interactable and will direct the secretary to the Current Status Page for the student where they can view and download the reports. (see Sections 3.4 and 3.9).

On the Register User page, the secretary can create a user account and add it to the system by filling out a form containing the required information. This in turn will send the user an email, informing them that they have been registered to the system (see Section 3.4).

### 3.6 Coordinator

The coordinator accounts are created by the secretary. Once the account is created, a student can login and go about their tasks as required.

From the dashboard, the coordinator can go to the "Student List" Page using the navigation bar on the left.

On the Student List page, the coordinator can view a list of the current students registered to Biltern. The rows of students on this page are interactable and will direct the coordinator to the Current Status Page for the student where they can view download the reports (see Section 3.4 and 3.9).

### **3.7 Grader**

The grader accounts are created by the secretary. Once the account is created, a student can login and go about their tasks as required.

From the dashboard, the grader can go to the "Student List" Page using the navigation bar on the left.

On the Student List page, the grader can view a list of the current students registered to Biltern. The rows of students on this page are interactable and will direct the grader to the Current Status Page for the student where they can download the reports and provide feedback if there have been uploaded reports (see Sections 3.4 and 3.9).

Additionally, graders are allowed to upload grading forms to provide a final grade to the student, when the time comes.

### **3.8 Teaching Assistant**

The teaching assistant accounts are created by the secretary. Once the account is created, a student can login and go about their tasks as required.

From the dashboard, the teaching assistant can go to the "Student List" Page using the navigation bar on the left.

On the Student List page, the TA can view a list of the current students registered to Biltern. The rows of students on this page are interactable and will direct the TA to the

Current Status Page for the student where they can download the reports and provide feedback if there have been uploaded reports (see Sections 3.4 and 3.9).

### 3.9 Student

The student accounts are created by the secretary. Once the account is created, a student can login and go about their tasks as required.

From the dashboard, the student can go to two pages using the navigation bar on the left: "Current Status" and "Uploaded Files".

The Current Status page allows the user to view the current status of their course as well as allow them to upload report pdfs, if the status requires it



Fig. 16: Current Status Page

The Uploaded Files page provides a list of all the files uploaded report pdfs by that student.



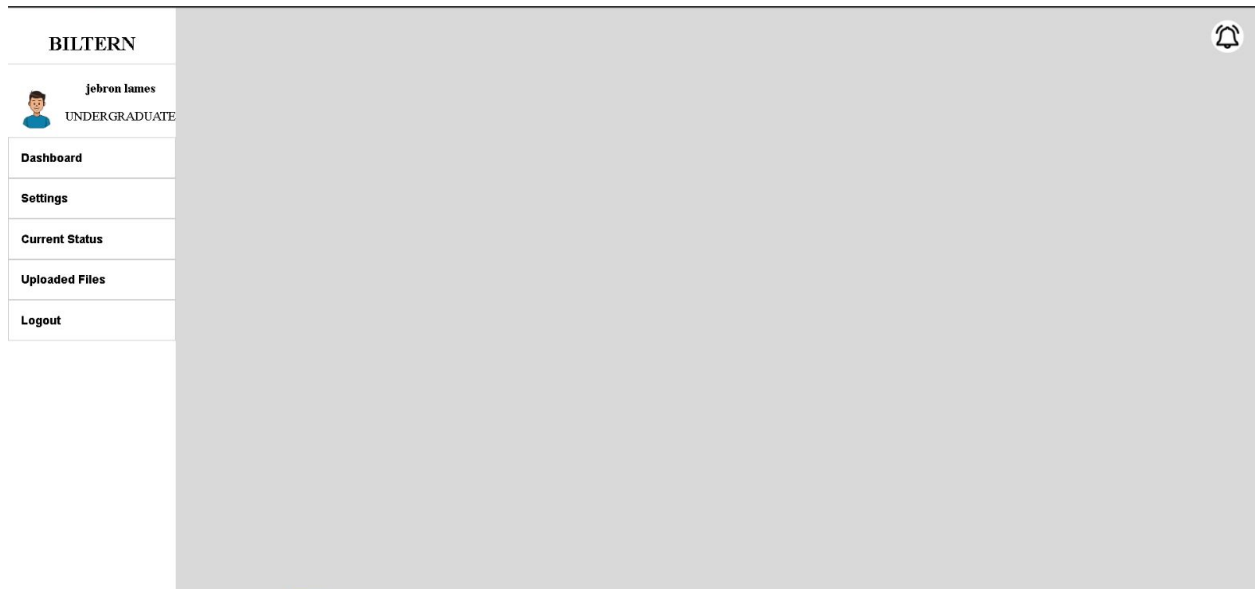


Fig. 17: Uploaded Files Page

## 4 Build Instructions

To build the project, first the Biltern repository should be cloned to local. The main branch must be used for deployment.

### Building back-end:

Oracle JDK Java 17 is required for the deployment of the back-end.

Opening the Biltern-Server folder as a project in IntelliJ IDEA(any version higher than 2021.1.1 in Community or Ultimate Versions) ). The SDK must be set to Following "application.yml" file should be added inside the resources folder to the exact relative path  
"

```
spring:
  mvc:
    format:
```

```
    date: yyyy-MM-dd HH:mm:ss

    date-time: yyyy-MM-dd HH:mm:ss

    time: HH:mm:ss

mail:

  host:

  username:

  password:

  port:

  properties:

    mail:

      smtp:

        auth: true

        starttls:

          enable: true

datasource:

  url:

  username:

  password:

  driver-class-name: com.mysql.cj.jdbc.Driver

jpa:

  hibernate:

    ddl-auto: update
```

```
show-sql: true

properties:

  hibernate:

    jdbc:

      time_zone: Europe/Istanbul

client:

  domain: http://127.0.0.1:5173

jwt:

  jwtHeader: Authorization

  idHeader: BilkentID

  jwtSecretKey:

  refreshTokenExpiration: 60

  accessTokenExpiration: 10

  onceUseTokenExpiration: 5
```

Note that fields for database, mail service and JWT secret key must be filled with custom information.

### **Building front-end:**

Node js and npm are required for the deployment of the front-end.

**Node js - v17.4.0 or higher**

**Npm - 8.11.0 or higher**

**Both can be installed through only installing Node js which is available here:**

<https://nodejs.org/en/download>

Opening the Biltern-Client folder in a terminal(VSCoDe, Git-bash, etc.), the first npm install command must be executed to install required npm packages. After the installment of npm packages is completed successfully, the front-end application can be started in localhost's port 5173 with the npm **run dev** command.

## **5 Work Allocation**

- **Cenk Merih Olcay 22002408**

- a. Analysis Report:**

- Wrote the current system part

- Drew Activity diagrams

- Drew State diagrams

- b. Design Report:**

- Created access control and security matrix

- Wrote and design Application layer part

- Wrote the second part in Object design trade-offs

- c. Implementation:**

- Implemented Spring Security logic required for method access security, Role

- based authorization and JWT generation, verification

- Created notification package

- Created role administration package and registration logic

- **Arshia Bakhshayesh 22001468**

- a. Analysis Report:** both Iteration use case diagram and explanations.
- b. Design Report:** First Iteration entity diagram, boundary conditions, one of design patterns, one section of trade-offs. Second iteration, general small fixes based on the feedback in hardware software mapping, design goals, boundary control, and design patterns
- c. Implementation:** The whole (backend) report package(feedback, gradingform) assignment package, course package, and general debugging and small changes in user package to match the usages. Final report screenshots and making sample data

- **Enes Bektaş 22002401**

- a. Analysis Report:**

Wrote the non-functional and pseudo requirements.

Designed dashboard pages for all users, notifications screen, uploaded files page, display file page and grading form page.

- b. Design Report:**

Designed Interface Layer in the Subsystem Decomposition with Faruk.

Helped creation of Application Layer in the Subsystem Decomposition.

Helped create access control and security matrix.

Designed boundary conditions diagram.

- c. Implementation:**

Implemented dashboard page and panels for all users.

Implemented UploadedFiles Page and Display File Page.

Implemented GradingFormPage.

- **Faruk Uçgun 22003016**

- a. Analysis Report:**

- Worked on non-functional and pseudo requirements.

- Designed the login, settings, semester initialization, role assignment, student list for different roles and current status for different roles.

- b. Design Report:**

- Designed Interface Layer in the Subsystem Decomposition with Enes.

- Wrote the explanations for subsystem decomposition, user interface layer

- Worked on access control table and object design trade-offs

- Wrote user interface layer class interfaces

- c. Implementation:**

- Implemented authentication including login, logout, change password, forgot password functionalities

- Implemented current status, notifications, navbar, role assignment, semester initialization, settings, student list pages depending on the role type.

- Wrote many other common files throughout the frontend part.

- **Faaiz Khan 22001476**

- a. Analysis Report:**

- Wrote the introductory section for the analysis report

- Wrote the functional requirements in the first iteration

- Drew the class diagram in the first iteration and edited it in the second iteration

Wrote the explanations of all the classes in the class diagram in the first iteration and changed them accordingly in the second

**b. Design Report:**

Wrote the entire introductory section: including the purpose of the system and the design goals

Worked on the subsystem decomposition diagram

Wrote the hardware/software mapping section

Drew the deployment diagram

Wrote the persistent data management section

Wrote the object design trade-offs section

Wrote the design patterns section

**c. Implementation:**

Wrote the entity classes for the users: Student, TeachingAssistant, Grader, Coordinator, Secretary, and BCC Admin

Wrote the repository interfaces for all user classes

Wrote the service layers for all user classes

Wrote the controllers for all user classes

Wrote the DTO classes for TeachingAssistant and Grader

Wrote the Course entity, its repository interfaces, its service and controller

