Bilkent University
Department of Computer Engineering

2022-2023 Fall Semester

CS 319: Object-Oriented Software Engineering

ERASMUS PROJECT

Group 2H

FINAL REPORT

Group Members	Student Numbers
Bora Yılmaz	22003359
Cengizhan Terzioğlu	22003014
Emirkan Derköken	22002693
Göktuğ Kuşcu	22002867
Murat Güney Kemal	22002692
Onur Asım İlhan	21903375

Instructor: Eray Tüzün

Teaching Assistants: Metehan Saçakçı, Emre Sülün, Muhammad Umair Ahmed, İdil Hanhan, Mert Kara

Table of Contents

1. Introduction

In this project, Bilkent University's Erasmus web page was redesigned according to the requests of the Erasmus coordinators. This web application's aim was to make the whole application process easier for both the students and the coordinators. We also digitalized the Erasmus application forms in order to reduce the number of papers and working hours wasted.

1.1. Changes in the Implementation

1.1.1. Services and Controllers

We have separated the services and controllers to different packages.

1.1.2. Add & Update Universities

Erasmus coordinators can add & update universities as an additional feature.

1.1.3. Application Period

Application period entity is removed as it just complicates things.

1.1.4. University Department

University department is added as a new entity since one university can have multiple departments that are available for Erasmus.

1.1.5. Bilkent Courses

Bilkent course is added as a new entity as it makes it easier to match the courses from other universities with Bilkent's courses and to control which course coordinators can see which course proposals.

1.1.6. Form

Form entity is removed because we realized keeping the forms in the applications entity is simpler.

1.1.7. International Office Home Page

There is no separate Erasmus points page for the international office. We implemented related functionalities in the home page.

2. Lessons Learned

2.1 Database Architecture Consistency

Constantly changing the database structure as needed wasn't a good idea. After each merge, we had to create our data from scratch which made us lose lots of time. We should have finalized the database tables at the start of the project to prevent this issue.

2.2 Multiple People Modifying the Same Part

Sometimes some parts of the project needed multiple changes and they were done by different people. This resulted in various conflicts since one person's change was not yet there in the others. If we could make the work allocation better, we could have avoided many of the conflicts.

3. User's Guide

Common pages

Common pages offer information about the application features, data present in the database, people to contact for further/unmentioned information and FAQ. These pages can be seen by anyone routed to the website.

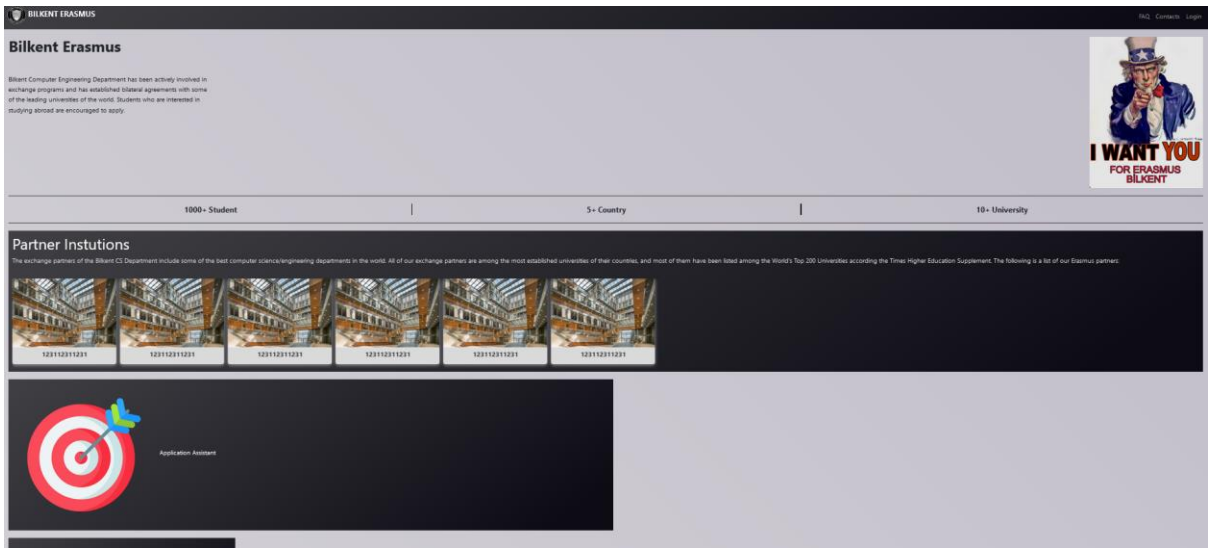

Figure 1: Main Page

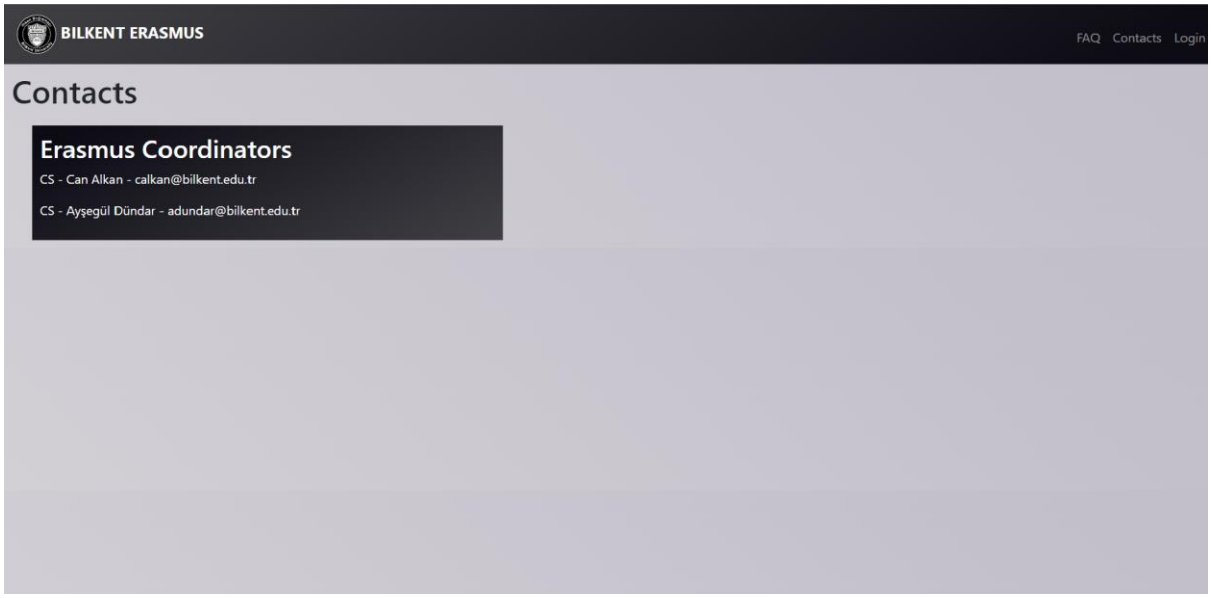

Figure 2: Contacts Page

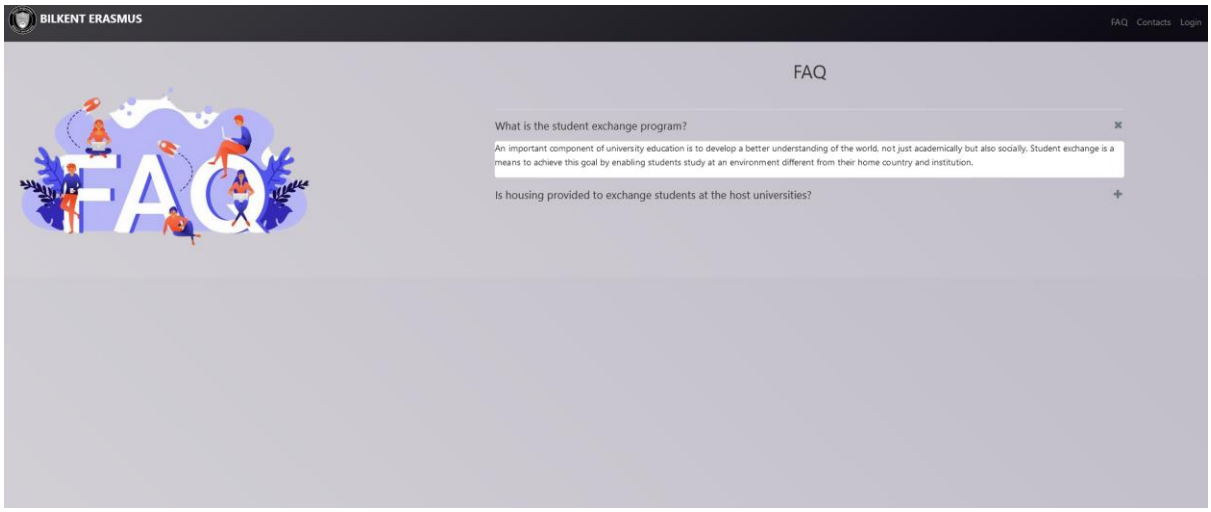

Figure 3: FAQ Page

Logging in

Users log in using their Bilkent ID and their SRS password. Successful logins direct the user to their home page. If the user has multiple roles, they are instead directed to the role selection page. Failed login attempts redirect the user to the login screen.

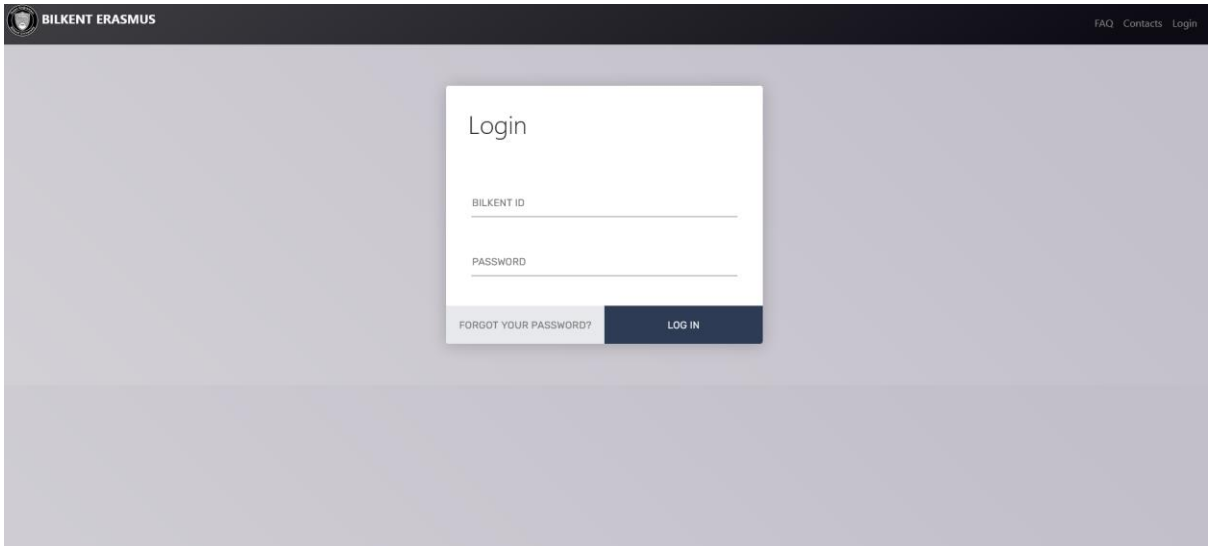

Figure 4: Login page

Role selection page

If a user has multiple roles (Erasmus coordinator may also be a course coordinator), the user is directed to the role selection page to decide the session role

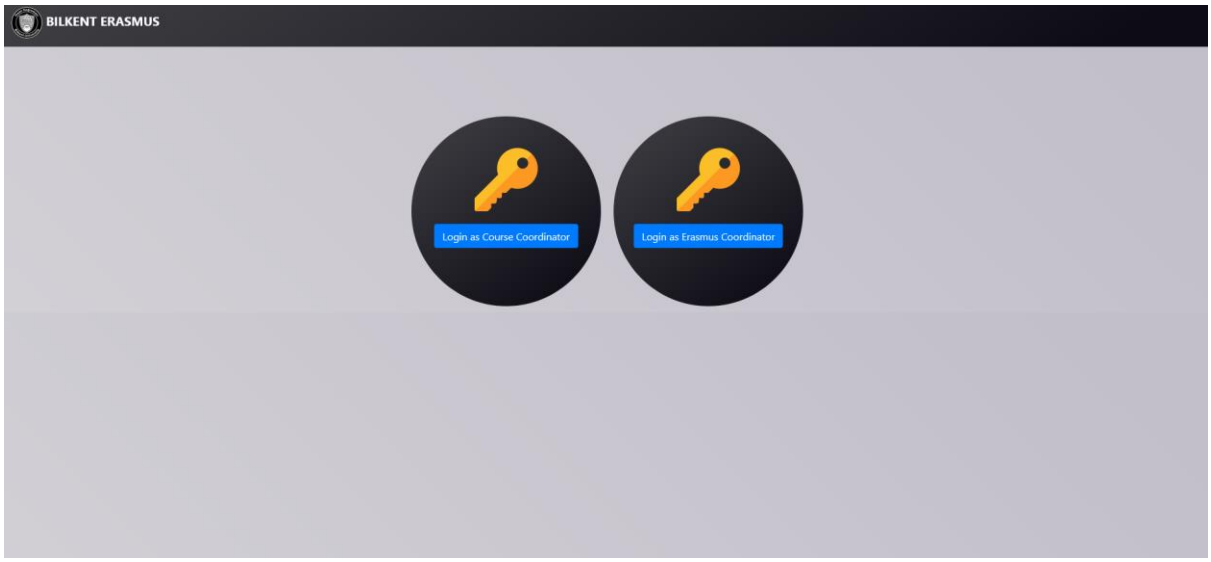

Figure 5: Role selection page

Home pages

After logging in, the user is directed to their role's home page. Each role and each user have their own home page.

Course coordinator home page

This page allows course coordinators to see the proposed mandatory courses related to their department, download the course's syllabus, go to its website, and approve or reject the mandatory course proposal.

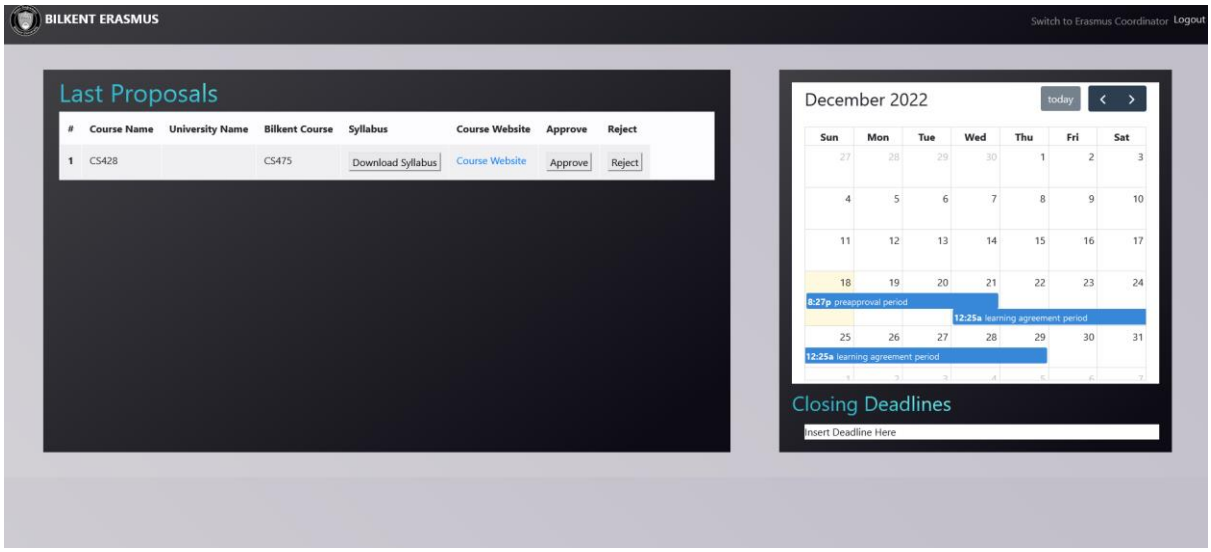

Figure 6: Course coordinator home page

International office home page

This page allows the international office to download a department's unranked applications in excel format and upload a department's placement table (in excel format) for the system to automatically place the applications to universities or to put them in the waiting list.

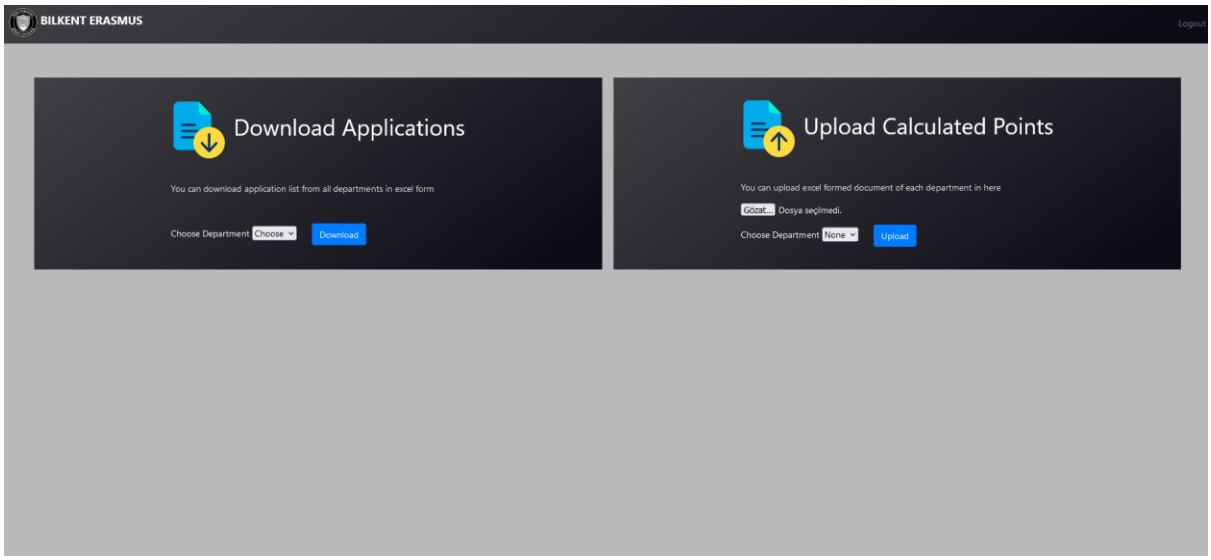

Figure 7: International office home page

Erasmus coordinator upload final form page

This page lets Erasmus coordinators see the applications that are suitable for uploading the course transition form and the application status after the course transfer form is uploaded. Coordinators can upload the form for a chosen application, update the form and delete it.

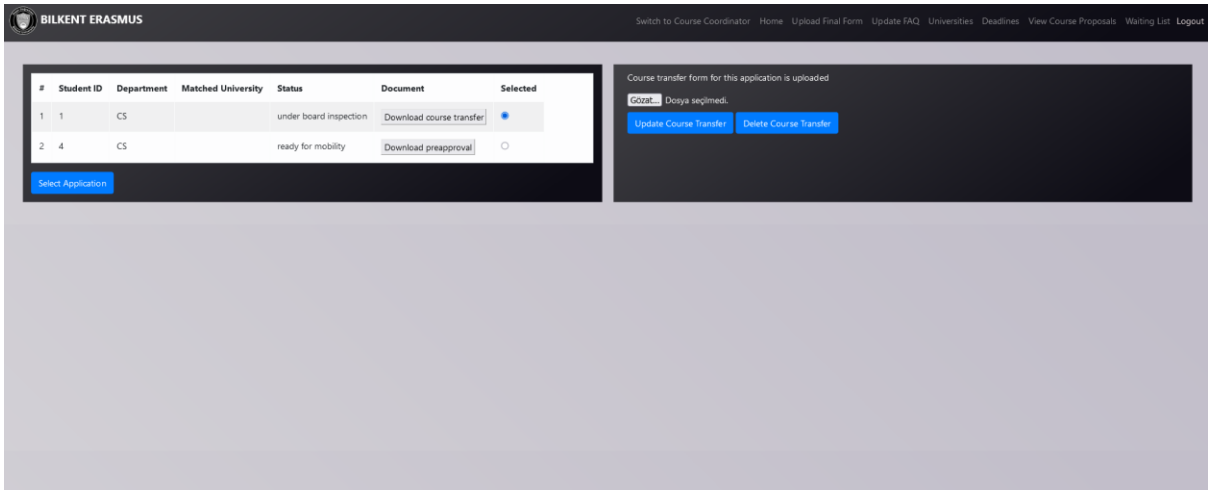

Figure 8: Erasmus coordinator upload final form page

Erasmus coordinator FAQ update page

This page allows Erasmus coordinators to see their department-related FAQ, add a new FAQ as well as update or delete the existing FAQ.

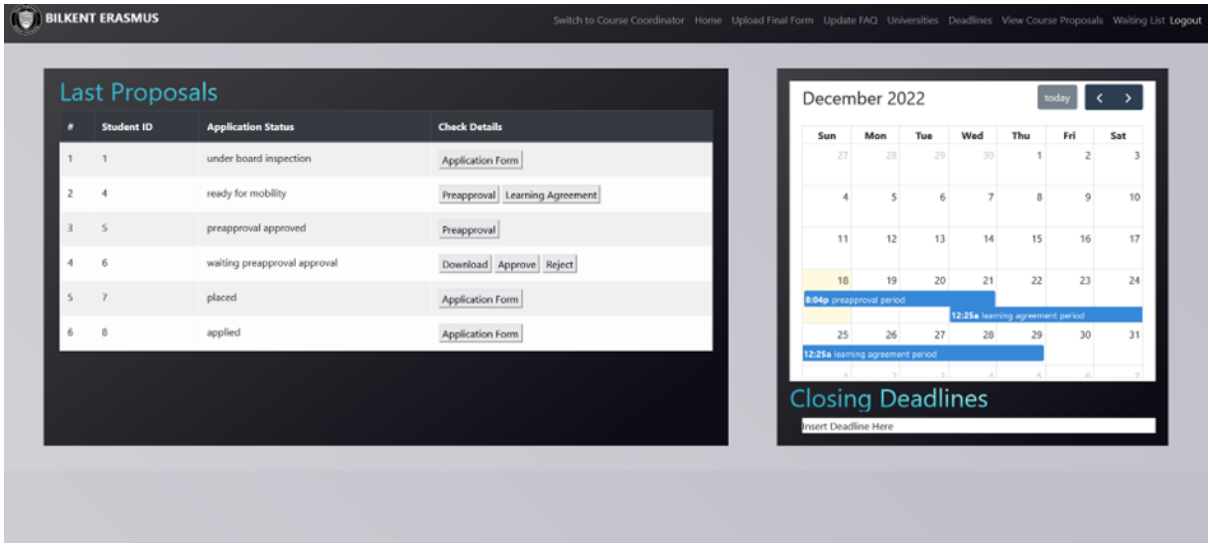

Figure 9: Erasmus coordinator update FAQ page

Erasmus coordinator universities page

This page allows Erasmus coordinators to update the existing universities' information and add a new university for their department.

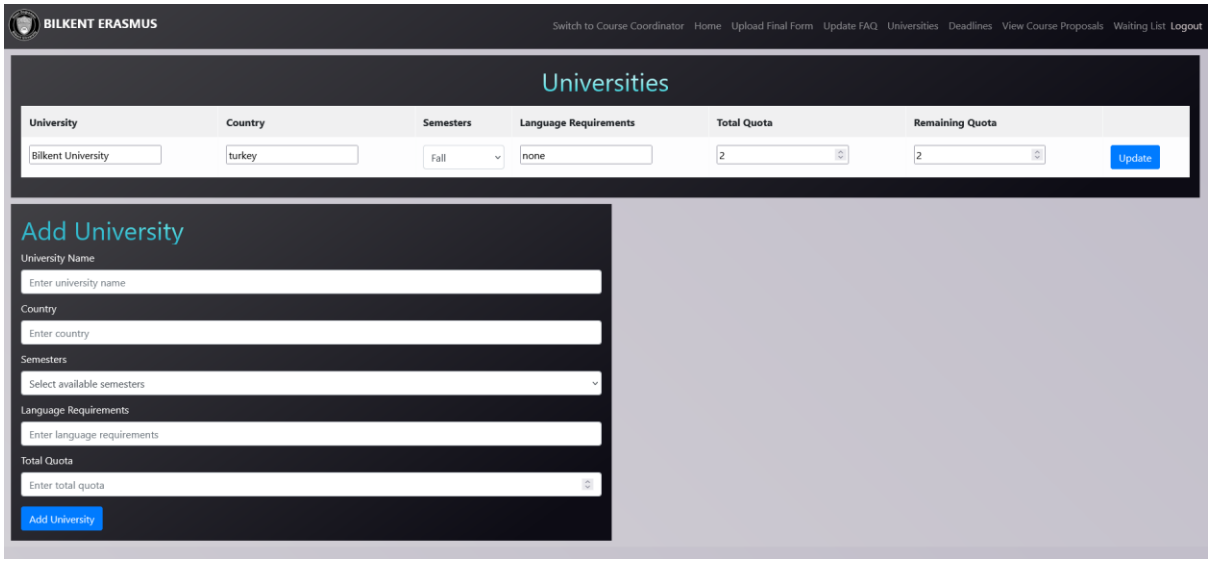

Figure 10: Erasmus coordinator universities page

Erasmus coordinator update deadline page

This page lets Erasmus coordinators update a pre-existing deadline (as the deadline types are constrained) on the global scale.

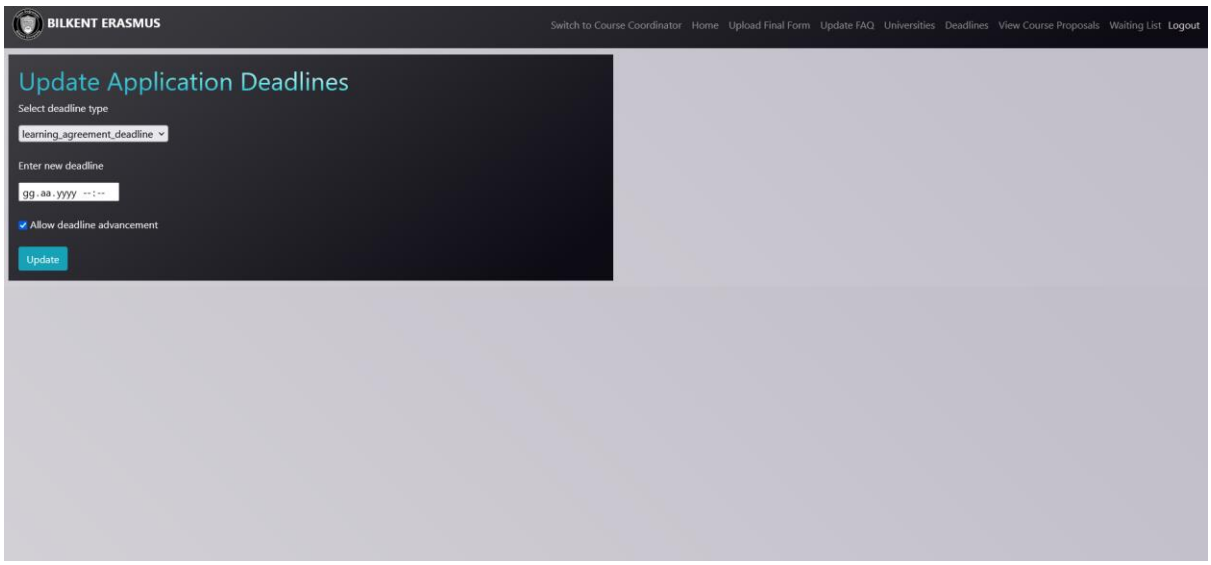

Figure 11: Erasmus coordinator update deadline page

Erasmus coordinator view course proposals page

This page allows Erasmus coordinators to see the proposed elective courses, download the course's syllabus, go to its website, and approve or reject the elective course proposal.

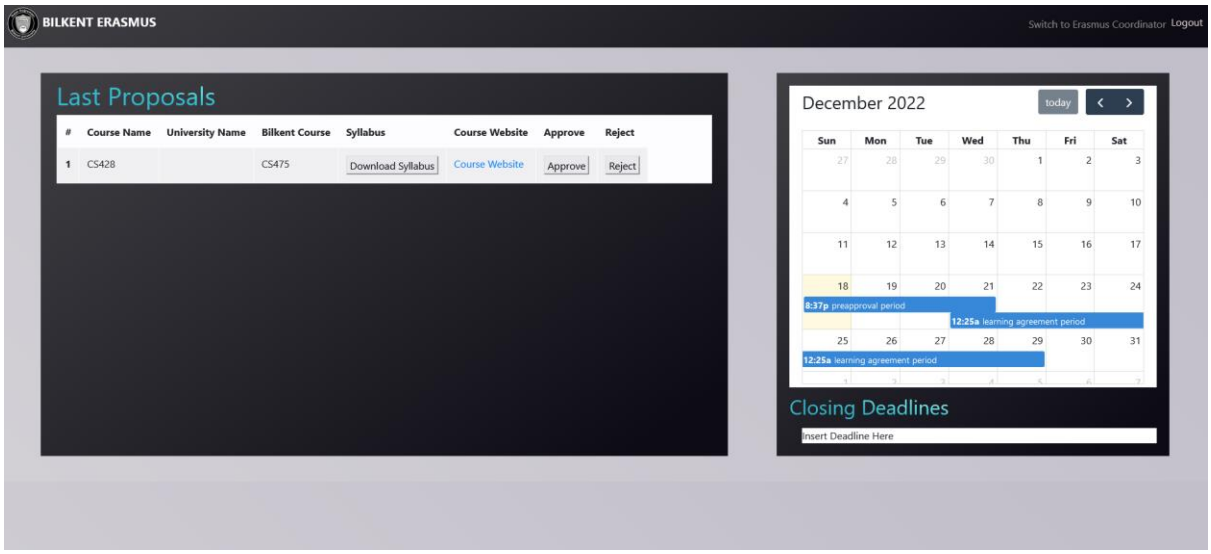

Figure 12: Erasmus coordinator view course proposals page

Erasmus coordinator waiting list page

This page allows Erasmus coordinators to see the applications in the waiting list, export the waiting list as an excel file, and manually place applications to universities.

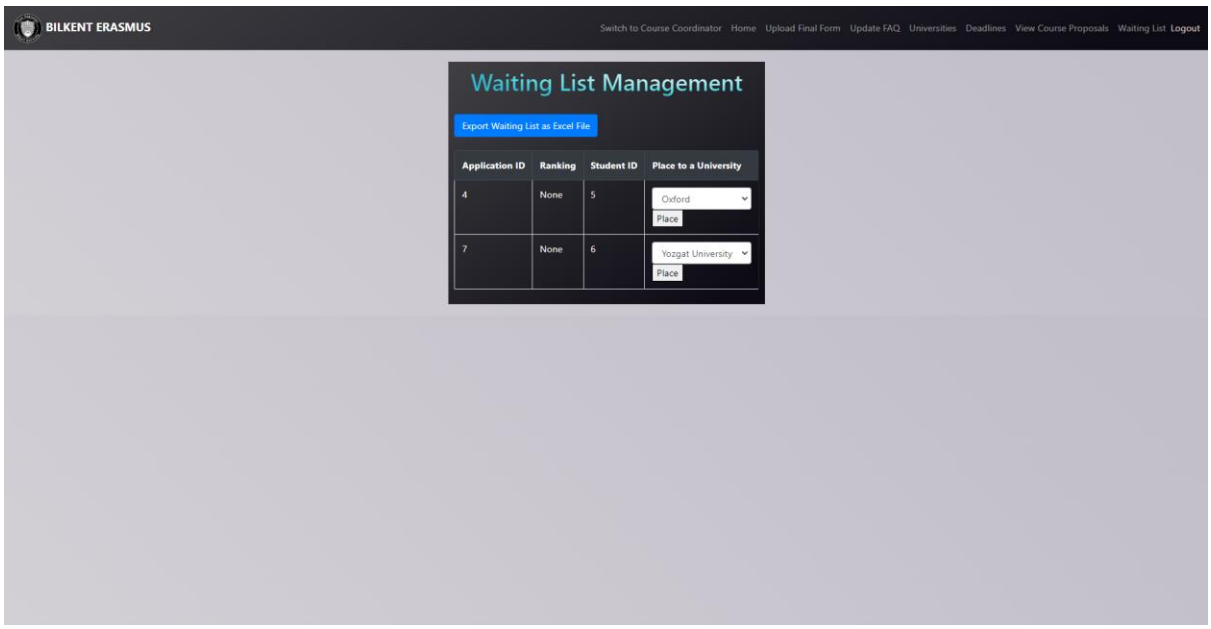

Figure 13: Erasmus coordinator waiting list page

Student home page

This page shows the student's current application status. The page changes dynamically according to which deadlines have passed. There is also a to-do list showing the steps the student has not completed yet, and a calendar showing the deadlines for the required forms.

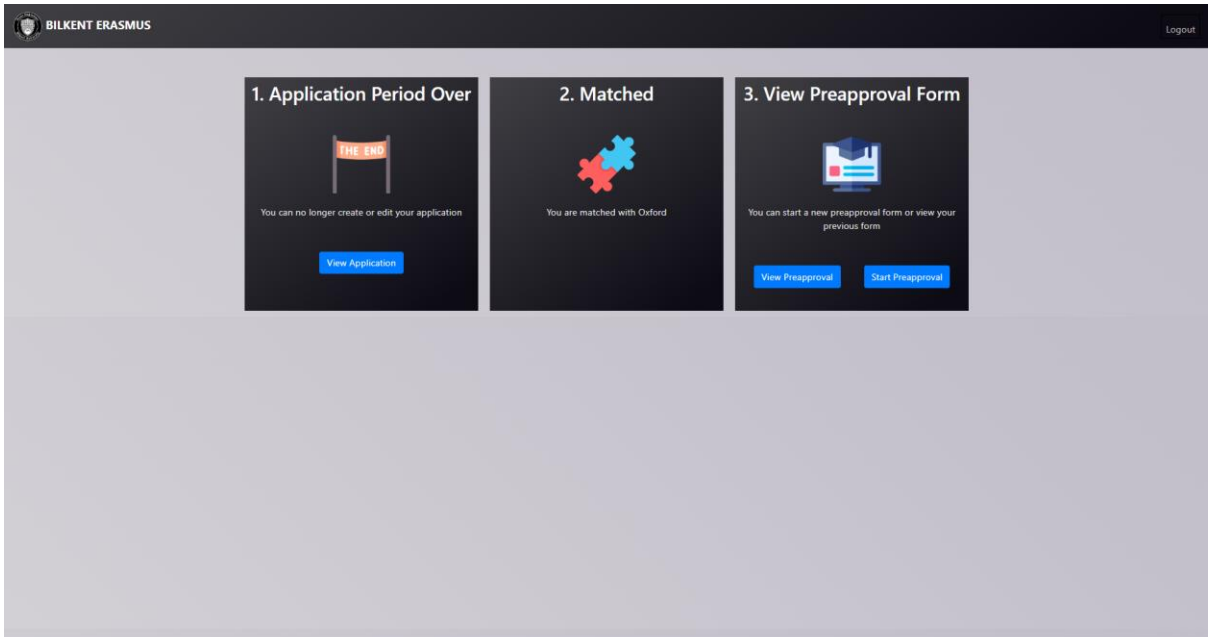

Figure 14: Student home page, learning agreement deadline has not passed.

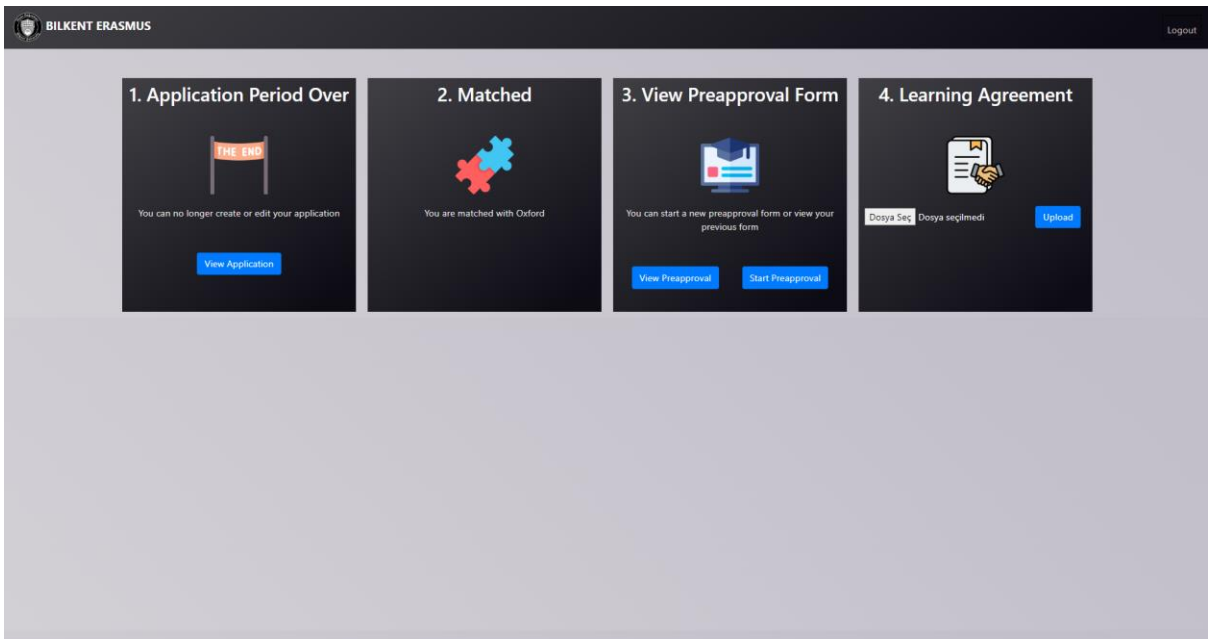

Figure 15 Student home page, learning agreement deadline has passed.

Student preapproval page

The student creates their preapproval form on this page. They can choose courses from their matched university. They can download the form in pdf form to be signed, and then upload the signed form.

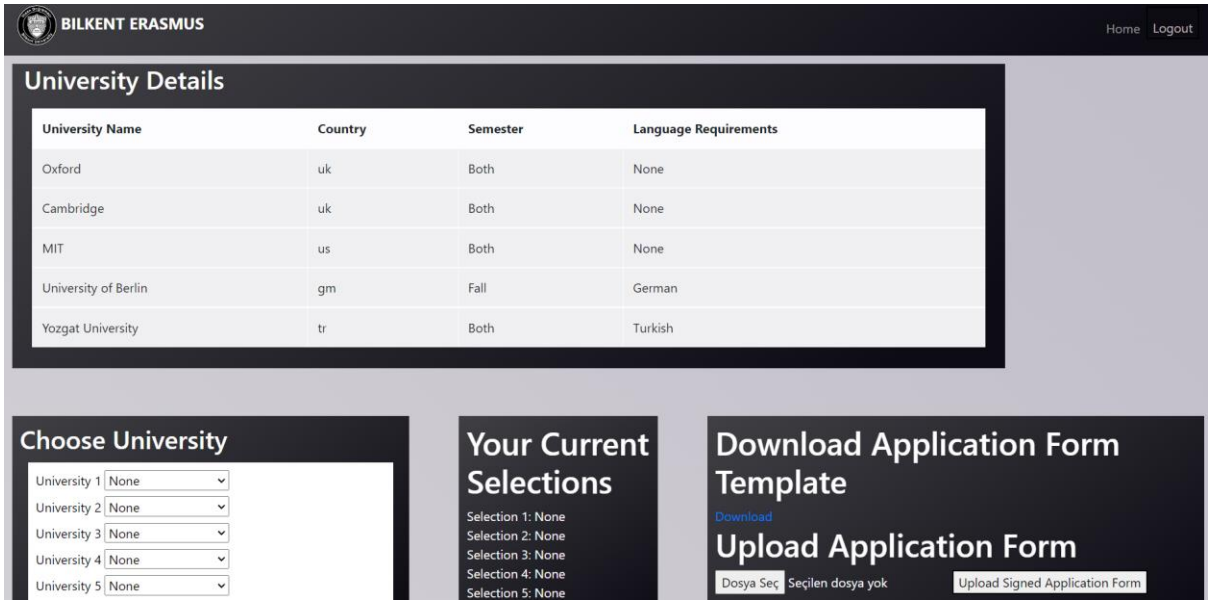

Figure 16: Erasmus Application Page

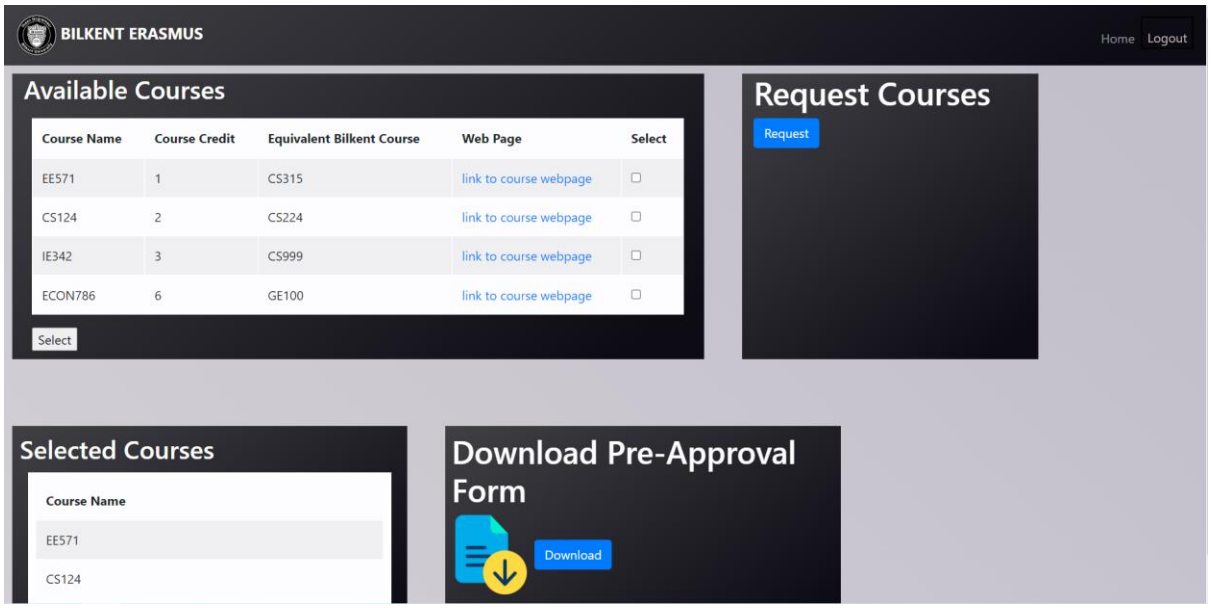

Figure 17: Preapproval form creation

4. Build Instructions

4.1 System Requirements

Minimum System Requirements:

Operating System: Windows® 7 64 Bit | Windows® 8.1 64 Bit | Windows® 10 Home 64 Bit
Processor: Intel® Core™ I7 930 | AMD® FX 6350
Memory: 4 GB RAM
Storage: 5 GB

Recommended System Requirements:

Operating System: Windows® 11 Home 64 Bit
Processor: Intel® Core™ I7 10750H | AMD® Ryzen 7 5800H
Memory: 16 GB RAM
Storage: 128 GB

4.2 Installation

Our Erasmus application project is implemented using Python. Hence Python should be installed on the computer to be able to run the server. Then the “pip install -r requirements.txt” command should be called from the terminal to install the libraries used in the implementation. After the libraries listed in requirements.txt are installed, run_server.py should be executed to run the server. This will generate a link to our project’s main page. After this point, opening the link in a web browser will be enough to reach our page. The HTTP server currently being used is Flask’s built-in development server, hence it is not suitable for production due to crucial security and performance reasons. In order to prepare the application for production deployment, a production quality HTTP server (such as Apache or nginx) and a Web Server Gateway Interface (such as Python’s waitress framework) are also needed to be installed and configured.

5. Work Allocation

5.1 Bora Yılmaz

- Wrote user types in analysis report iteration 1
- Drew the object design diagram in analysis report iteration 1
- Changed the object design diagram in analysis report iteration 2
- Reviewed additional features in analysis report iteration 2
- Drew the web service layer diagrams in design report iteration 1
- Redrew subsystem decomposition diagram and reviewed authorization matrix in design report iteration 2
- Redrew the web service layer diagrams in design report iteration 2
- Implemented Erasmus coordinator home (with Cengizhan), final report upload, FAQ management, deadline management pages
- Implemented calendar feature

5.2 Murat Güney Kemal

- Created state diagrams in analysis report iteration 1.
- Created state, sequence, activity diagrams in analysis report iteration 2.
- Wrote user types in analysis report iteration 1.
- Wrote some of the non-functional requirements in analysis report iteration 1.
- Wrote the introduction section of the design report iteration 1.
- Wrote the data management layer section of the design report iteration 1&2.
- Created entity diagram for data management layer.
- Wrote Data Management Layer Class Interfaces section for design report iteration 2.
- Implemented Course Coordinator homepage.
- Implemented Course service and course coordinator controller.
- Implemented FAQ service.
- Implemented Erasmus Coordinator’s elective course evaluation process.
- Implemented International Office service and controller.

5.3 Onur Asım İlhan

- Wrote introduction part of analysis report 1 and 2
- Activity Diagram and Sequence Diagram of analysis document of analysis report 1
- Created mockup pages on analysis report 1 and 2
- Created final object design part of design report 1 and 2
- Created boundary, controller and services of all design with Emirkan
- I mainly focused on the front end of our project, creating and editing almost all of the html files. Prepared templates for other group members to fill in from the database. Worked concurrently with almost all group members to create a well-designed UI.

5.4 Emirkan Derköken

- Designed the overall architecture of the Requirement Analysis Report and allocated the works
- Wrote the introduction in the Requirements Analysis Report in iterations 1 and 2
- Wrote the authentication and authorization section in the Requirements Analysis Report in iterations 1 and 2
- Wrote the student & faculty administration part in the user types section of the Requirements Analysis Report in iterations 1 and 2
- Sketched the Object and Class Model diagram in iteration 1
- Cleaned up, refactored and proofread Requirements Analysis Report's first iteration
- Designed the overall architecture of the Design Report and allocated the works
- Designed the 4 Layer Architecture (boundary-controller-service-model) used in the application and is followed throughout the Design Report
- Wrote the Object Design Trade-offs section of the Design Report in iterations 1 and 2
- Wrote the External Packages section of the Design Report in iterations 1 and 2
- Wrote the explanations of the class interfaces that were assigned to me
- Decided on the tech stack that is used in the implementation considering facts that the group was short of frontend developers, performance was not a big concern for the application and our time was limited, so production speed was very important
- Designed the initial architecture of the application's database (all tables and their relations) and continued to improve the database throughout the way
- Implemented the first iteration of the 4 Layer Architecture and built the initial blueprint of the application
- Implemented the authentication and authorization functionalities of the application which includes the login page, user logins & logouts, sessions, roles and certain other services.
- Since the authentication and authorization functionalities were the first building blocks of the application, their implementations also act a guideline for the other team members
- Implemented the backend part of the student homepage which includes several visibility controls mechanisms
- Implemented the Erasmus application functionality of the student users.
- Implemented Pre-Approval Form related functionalities such as course selection and course proposal
- Implemented Learning Agreement Form related features such as file uploads and downloads which are also used by the other forms.
- Implemented the PDFService class which automatically generates Application and Pre-Approval Forms by filling in the required information such as course and university selections and user & university credentials for the student users
- The PDFService class is also capable of automatically signing submitted Pre-Approval and Learning Agreement Forms and filing in the date & coordinator name information with the click of a button if the Erasmus Coordinator user provides his/her signature in .png/.jpeg format.
- Implemented Singleton Design Pattern with multi-thread safe to improve performance and increase functionality
- Implemented and helped with the implementation of many services.

- Throughout the semester, I actively used git and GitHub. I performed all of the merge operations, resolved conflicts, fixed bugs and guided my teammates through git's feature branch workflow.

5.5 Cengizhan Terzioğlu

- Wrote the parts about to do list and calendar features for analysis report iteration 1
- Sketched the use case diagrams of exchange coordinator, dean/director and chair and explained all of their use cases for analysis report iteration 1
- Sketched the revised version of use case diagrams of all users and explained their use cases for analysis report iteration 2
- Wrote high-level software section for design report iteration 1
- Sketched subsystem decomposition for design report iteration 1
- Explained class interfaces from FormController to CalendarService for design report iteration 2
- Implemented the universities page for Erasmus coordinators
- Implemented the waiting bin management page for Erasmus coordinators
- Implemented the last proposals panel in the home page of Erasmus coordinators

5.6 Göktuğ Kuşcu

- Created the use-case diagrams of student, international office and exchange coordinator for analysis report iteration 1
- Created the user interface management layer for design report iteration 1
- Wrote parts of high-level software section for design report iteration 1
- Wrote high-level software section for design report iteration 2
- Updated the user interface management layer for design report iteration 2
- Wrote class interfaces of user interface management layer for design report iteration 2
- Wrote functional requirements section for analysis report iteration 2
- Implemented home page for international office
- Implemented final placements page for international office
- Implemented student preapproval page

6. Implementation Status

What has been implemented:

- UI navigation
- Login process
- Student application process
- Application approval
- View application status
- Course proposal
- Course evaluation by course coordinator
- Elective course evaluation by Erasmus coordinator
- Automatic placement for International Office
- Calendar
- Support for multiple forms
- Deadline management
- FAQ management
- Manual placement from waiting bin
- Automatic PDF (form) generation
- Automatic signature insertion to PDF forms

We could not manage to finish:

- Messanger
- Administrative Coordinator