Bilkent University

Department of Computer Engineering

CS319 Term Project

Section 3

Group 3A

Bilasmus

Final Report

Group Members:

Alp Afyonluoğlu (22003229)
Emre Karataş (22001641)
Erkin Aydın (22002956)
Parsa Keihan (22001422)
Selin Bahar Gündoğar (22001514)

Instructor: Asst. Professor Eray Tüzün
Teaching Assistant(s): Muhammad Umair Ahmed, Emre Sülün, Mert Kara, İdil Hanhan

Contents

Final Report

Bilasmus

1. Introduction

Bilasmus application includes several different user interfaces for its different user types: outgoing student, coordinator, incoming student, international student office, admin, department secretary, and instructor. The main functionalities have been implemented in the application; though, some changes had to be made to our design ideas which will be discussed in the further paragraphs. Besides, all users of Bilasmus must be registered by the admin in which the users receive an email consisting of the link to go to the Bilasmus application and change his/her password. All notifications of the application are sent through email; furthermore, notifications are sent whenever a coordinator, instructor, or department secretary approves or rejects a form that the student had uploaded or whenever the international student office had posted a language proficiency exam date announcement. Once the user logs in, there are different functionalities based on what the user type is and only certain users can access the documents of other users.

There were some functionalities that were foreseen to be included in the project; though, due to time limitations, they could not be implemented. For instance, an in-app messaging mode was going to be uploaded to allow users to communicate within the app. Moreover, a to-do list was to be included in all user types' side bar to be aware of what should be done next by the user. Furthermore, an in-app notification system was to be implemented; though; as a team it was decided to move the notifications part of the application to sending emails as the notifications have a higher chance of being seen through email. Additionally, an e-signature system was to be implemented; though, due to time constraints it could not be implemented and instead the user that had to sign a document has to upload the hand-signed version of the document to the application.

As mentioned before, there are different user interfaces for different user types. For instance, the outgoing student user has access to uploading course request forms, Pre-Approval forms, Learning Agreement forms, and a language proficiency exam upload selection. These forms, except for the language proficiency exam, go to the coordinator. The course request form may be sent to the instructor if the wished course is a mandatory course. Additionally, the coordinator user is capable of seeing and evaluating uploaded course requests, Pre-Approval forms, and Learning Agreement Forms. Furthermore, the coordinator also uploads the Course Transfer forms, and can see the transcripts of all outgoing student users who returned from Erasmus/Exchange. The uploaded Course Transfer Form and the approved Pre-Approval Forms are sent to the Faculty Committee Executive Board. The incoming students can send a course request to coordinator which is later sent to the department secretary automatically by the system. International student office announces language proficiency exam dates and receives and evaluates language proficiency exam results. The admin uploads the transcripts of Bilkent students returning from Erasmus/Exchange.

The coordinator has 10 pages: main page, course request page, detailed course request page, pre-approval form page, pre-approval form upload page, learning agreement page, learning agreement upload page, course transfer form page, student wait list page, and transcript page. The home page displays the upload status of outgoing students for the forms course request, pre-approval, and learning agreement. The Course Request Page lists all the course request forms sent by outgoing students. The coordinator can click on the button View to view the list of the courses. Furthermore, the user can click on the button Reject to reject all the courses of the student or the user can click on Approve to approve all courses listed for that student. Once the user clicks on View, the user will be redirected to the Course Requests detailed page. On this page, the user can see the course names, course types, status of instructor evaluated, course link, course syllabus, and the final decision of the coordinator regarding the course. By clicking on the course link and syllabus, the coordinator will be redirected to the specific course link or the syllabus. Moreover, by clicking on the Approve and Reject buttons the coordinator can approve or reject the individual courses. The Pre-Approval Form page lists all uploaded pre-approval files by outgoing students. The coordinator can download the form by clicking on the view button or reject and approve by clicking on the reject and Upload Signed Form buttons. The Upload Signed Form button takes the coordinator to the Upload Pre-Approval Form page where the coordinator can upload the signed Pre-Approval form and download the file to check if the correct file was uploaded. Once the coordinator submits the pre-approval file, the file is sent to the Faculty Executive Committee Board. The Learning Agreement page lists all learning agreement files uploaded by outgoing student. The coordinator can download the form by clicking on the view button, and approve or reject the form by clicking on the Upload Signed Form button and the Reject button. By clicking on the Upload Signed Form button, the user is taken to the Upload Learning Agreement page where the coordinator can upload the signed version of the file and also download the uploaded file by clicking on the Download File button. Moreover, on the Course Transfer Form, the user can search the name of a student through the search engine and click on the search button the have all the matching names be listed on the data table. The coordinator can select any user by clicking on the Select button of the outgoing student's row. Once the user is selected, the coordinator can upload the course transfer form and click on the Upload button to send the form to the Faculty Executive Committee Board. The user can also download the file by clicking on the Download File button. Additionally, the Student Wait List page displays the Excel sheet of all students who are in the wait list. The coordinator can update the Excel sheet through the application. The Transcripts page includes a search bar that allows the coordinator to see a list of all uploaded transcripts of outgoing students who returned from Erasmus. Once the user searches a student name by clicking on the Search button, the outgoing students with matching names will be listed in the data table and the coordinator can choose to download the specific transcript by clicking on the Download button situated in the same row as the student's name.

The instructor user has 1 page: Waiting Course Requests page. On this page, the mandatory courses that an outgoing student wishes to take is listed if that course's coordinator is the instructor. The information such as the student's name, course name, course type, course link, and course syllabus are listed on the page. Moreover, the instructor user can visit the course link by clicking on the button Course Link and can view the syllabus by clicking on the syllabus button. Furthermore, the instructor can make the final decision whether the course is approved or not by clicking on the Approve and Reject buttons found on each row of courses.

The department secretary has 1 page: course requests. The course requests of incoming students are received by the coordinator at first. Then, once the coordinator approves the courses, the courses are sent to the department secretary to register to courses on SRS. The department secretary can manually click on the Approve or Reject buttons to indicate if the courses have been registered on SRS.

The admin user has 2 pages: Main page and transcript page. The main page has four significant functionalities: adding a user, updating email of a user, deleting a user, and maintenance of the system. Adding a user is done by entering or selecting 5 attributes: name, surname, bilkent ID, e-mail, and user type. When clicked on “add user” button, the new user is added to the system. For deleting the user there is only a need of bilkent ID and “delete user” button removes the specified user from system. In order to update a user’s email address, the bilkent ID and the new email of a user is needed, and “update user” button is sufficient when clicked. Last but not least, maintenance is available by letting the admin provide needed information about the date and time of the maintenance. Next, transcript page has two primary functionalities: downloading or uploading the transcript of a specified student. The student can be searched and found on a search bar provided, and if it exists it can be selected. After selection, if there is already a transcript existed, the file can be downloaded. Otherwise, admin can brows or drag a transcript file in PDF format to the section provided and upload it by clicking on “upload” button.

The outgoing student user has 6 pages: main page, course request page, preapproval upload page, language proficiency upload page, learning agreement upload page, and profile page. The main page has three main functionalities: 1)it shows the application status, meaning how many of the actions of an outgoing student done, 2)it gives the current course list of the student, and 3)there are links that can come handy to an outgoing student time to time. The course request page is designed to prepare a list of requested courses to take in ERASMUS and send it to the system. In the preapproval upload page, language proficiency upload page, and learning agreement upload page, the outgoing student can upload, update, and delete their corresponding forms. The profile page gives the general information of the outgoing student in the system, such as name, surname, and Bilkent ID. It also has a button for sending a request to admin for information change.

The international student office user has 2 pages: main page and evaluate language proficiency page. In the main page, setting an exam to a language with a date can be done. Additionally, a registration link can be sent using the id of a student. In evaluate language proficiency page, approving and rejecting students’ proficiency exams can be done. After rejection or approval, the list of rejected and approved students is updated and displayed.

The incoming student user has 3 pages: main page, course request page, and a profile page. On the main page, the incoming students can see the list of the courses they are taking in Bilkent. On the course request page, incoming students can prepare a course request list and send it to the system, just like outgoing students do. On the profile page, incoming students can see their information on the system.

The faculty committee board user has 5 pages: the main page, preapproval page, preapproval sign page, transfer form page, and transfer form sign page. The main page displays waiting preapproval and course transfer forms’ lists. Using buttons of elements of these lists is the same as using navbar buttons. In the preapproval list page, buttons for viewing, approving, and rejecting preapproval form exist for each preapproval form. The course transfer form is the same as preapproval list page. Only the form type differs.

The login page is necessary and provided to all user types. It collects two information: an email address and a password to enter the system. If the information provided doesn’t match, then an alert is shown and reenterance is needed. There is also a reset password button wich directs the user to the reset password with email page.

The reset password directory consists of two pages: one for providing an email and one for changing the password. At first an email has to be entered which is already registered in the system. After that, a link of password changing page is sent to that email and user can enter and re-enter a new password for his/her account.

The register page is provided only after admin creates a new user. A link is sent to the e-mail provided by admin while adding a new user to the system. The added user can click on that link and direct to a new page called register. In that page he/she can only provide a password for their account with the rules specified.

2. Lessons Learned

During the project, we focused a lot of our time on analysis and design, which greatly helped us during the implementation stage. We were able to add features and meet requirements efficiently because we had a clear understanding of what we were building and how to build it. Before this project, we did not have prioritized requirement elicitation and design phases, and we would often jump straight into implementation in our other projects before, leading to difficulties when trying to make changes or add new features.

In this project we learned how to manage responsibilities in the group. We were divided into two groups; two of us were working for backend and three of us were working for frontend. These two different groups contacted and met each other regularly during the project, mostly in the implementation phase. In this project,we learned how important the communication in the group is. We worked with Node.js for backend code, PostgreSQL for database, HTML, CSS, and JavaScript for the frontend. Even though we were divided into groups, at the end of implementation all of us knew the full project structure and we were dealing with all of the bugs and errors together. We also had to help each other out and instead of working only in either backend or frontend, we all also did full-stack.

This project taught us the importance of thorough analysis and design phases. However, we faced challenges with the implementation due to time constraints, which may have resulted in missed edge cases and the inability to fully implement certain features. We learned that it is important to allocate enough time for planning and to not wait until the last minute to begin working on a project to produce a better outcome and make the implementation process smoother.

3. User’s Guide

Users can enter the system using their email and password. If they missenter one of the credentials, they get an error according to the problem with their credentials. There are different error messages, only one is shown below.

3.1. Logging In

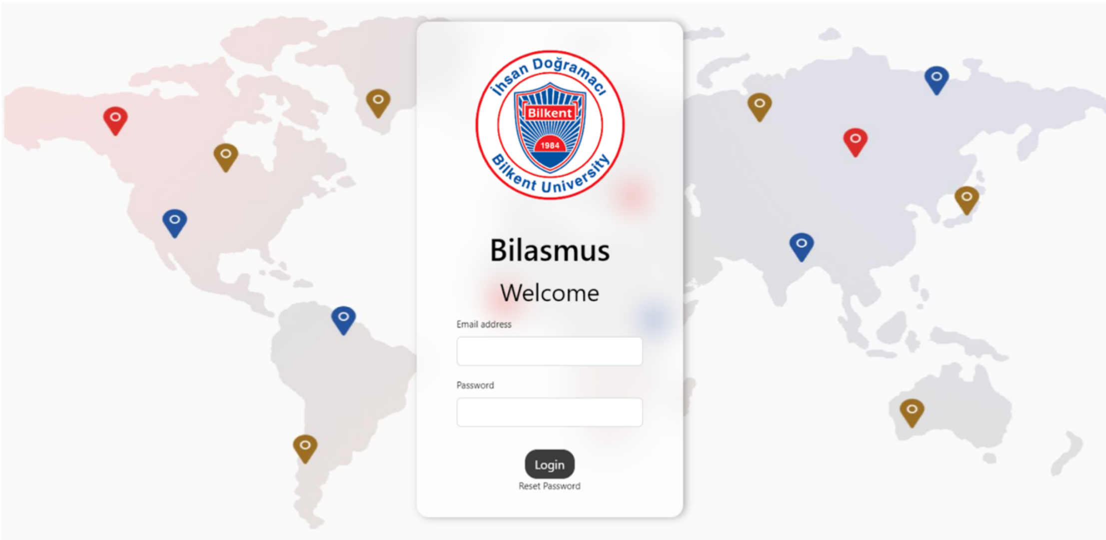

Figure 1: Login Page

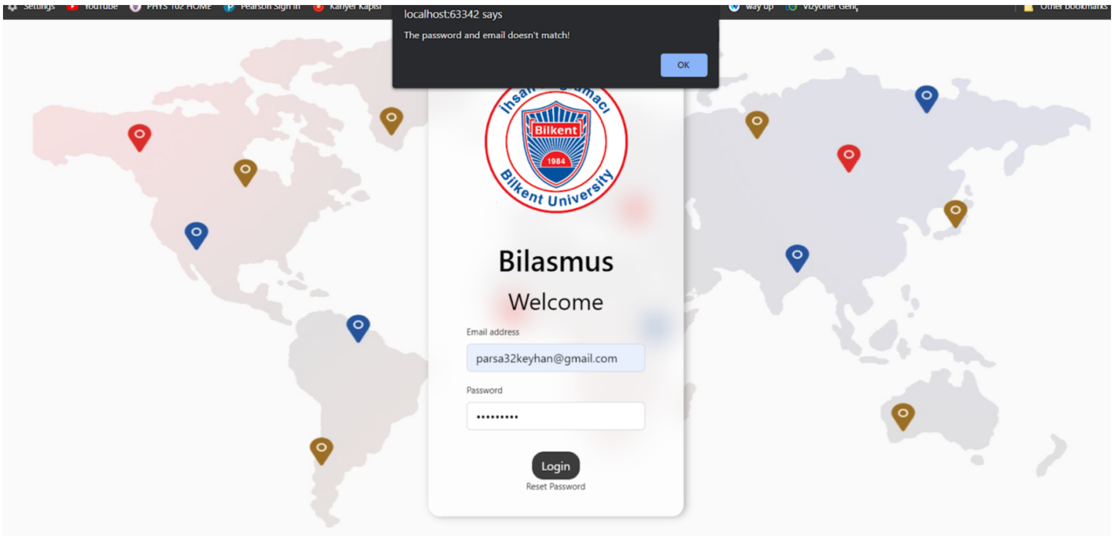

Figure 2: An Error Message in Login Page

Users can register to the system through the link emailed by admin and setting a password. There are different error messages for mistakes in the passwords.

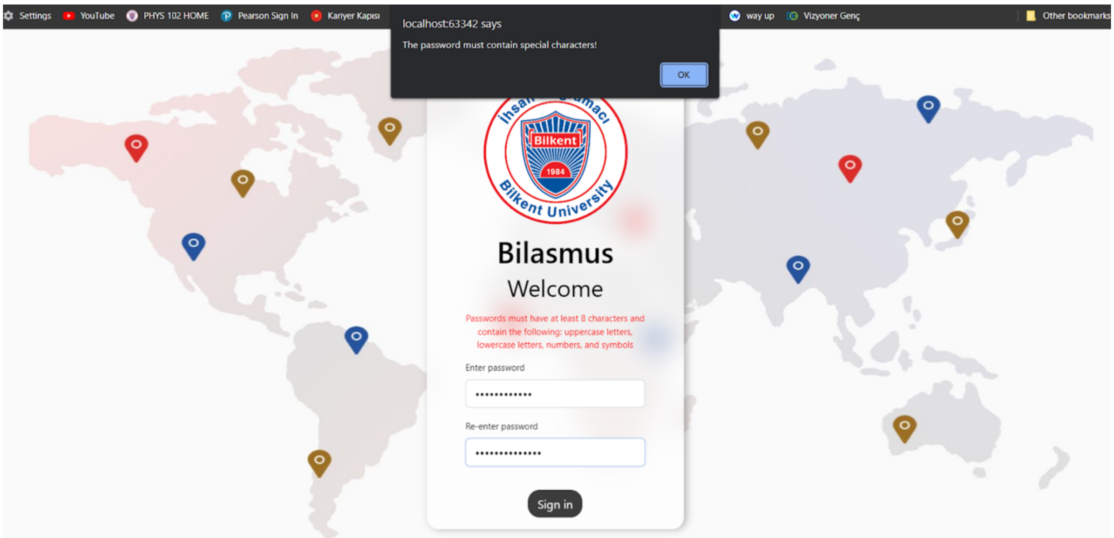

Figure 3: Register Page

Users can reset their passwords. They need to enter their emails and a link will be send to them automatically to direct them to reset the password page.

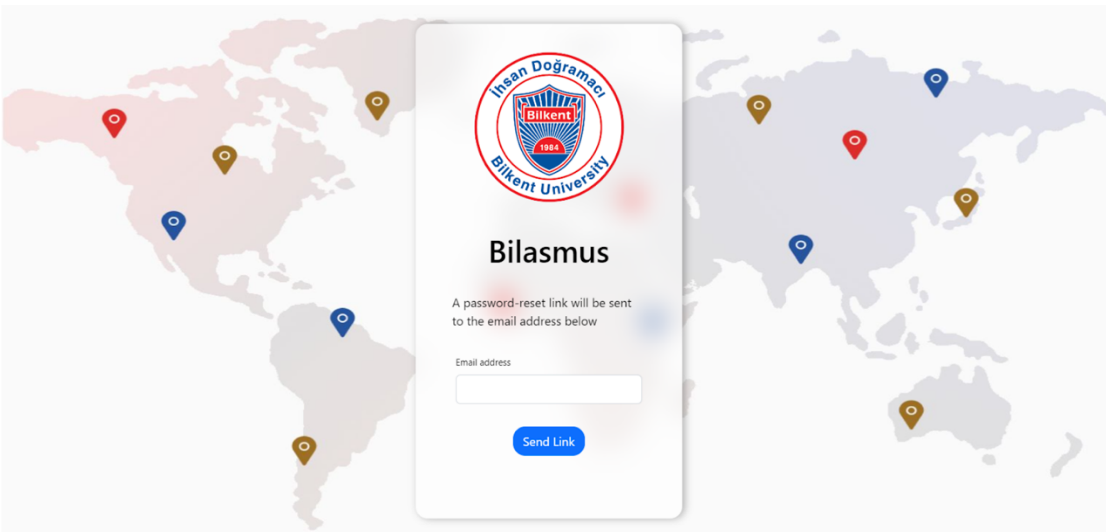

Figure 4: Reset Password Email

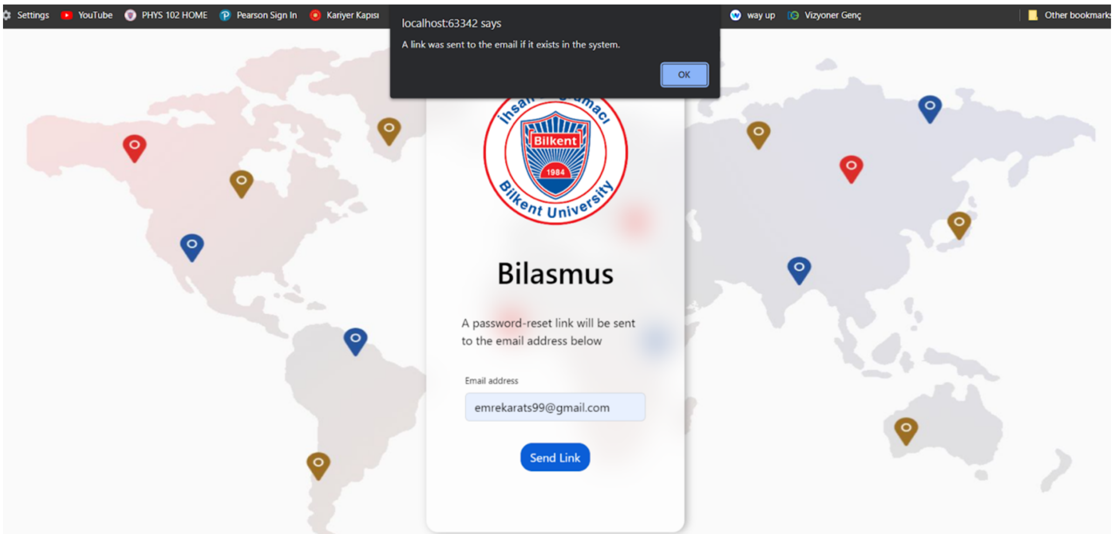

Figure 5: Reset Password Email Notify Message

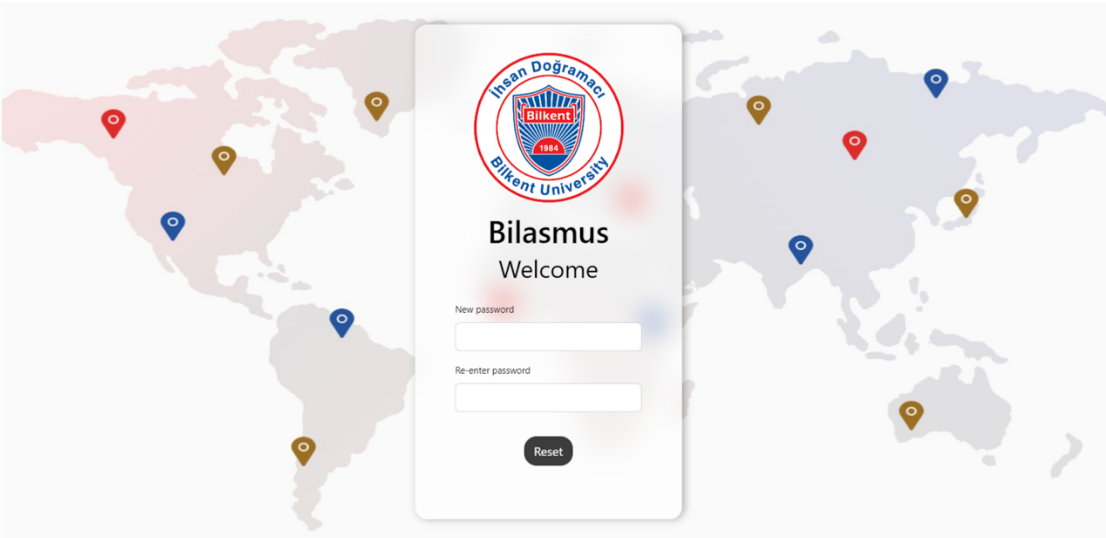

Figure 6: Reset Password Page

3.2. Coordinator

Coordinators can use their main page to go to other pages, either by using navbar or using buttons of Course Requests, Pre-Approval Forms, and Learning Agreement Forms pages.

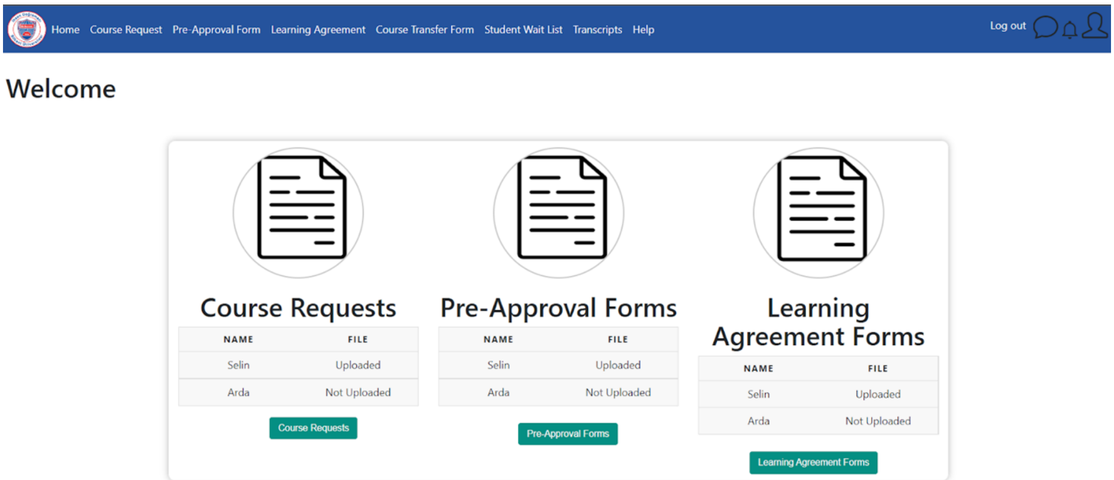

Figure 7: Coordinator Main Page

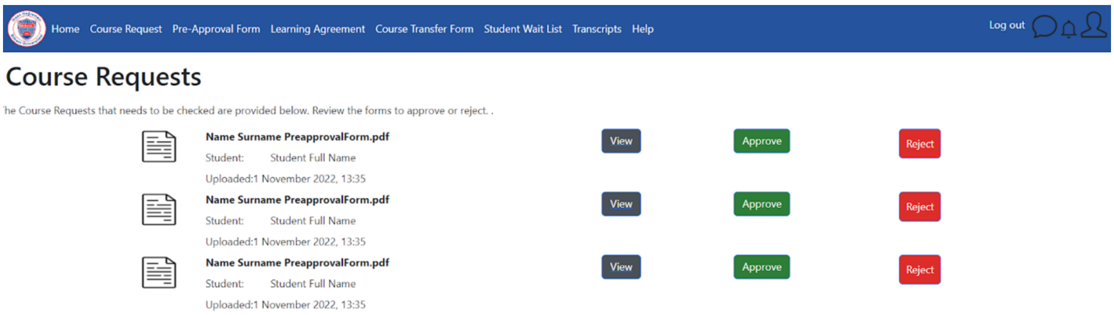

Figure 8: Coordinator Course Requests Page

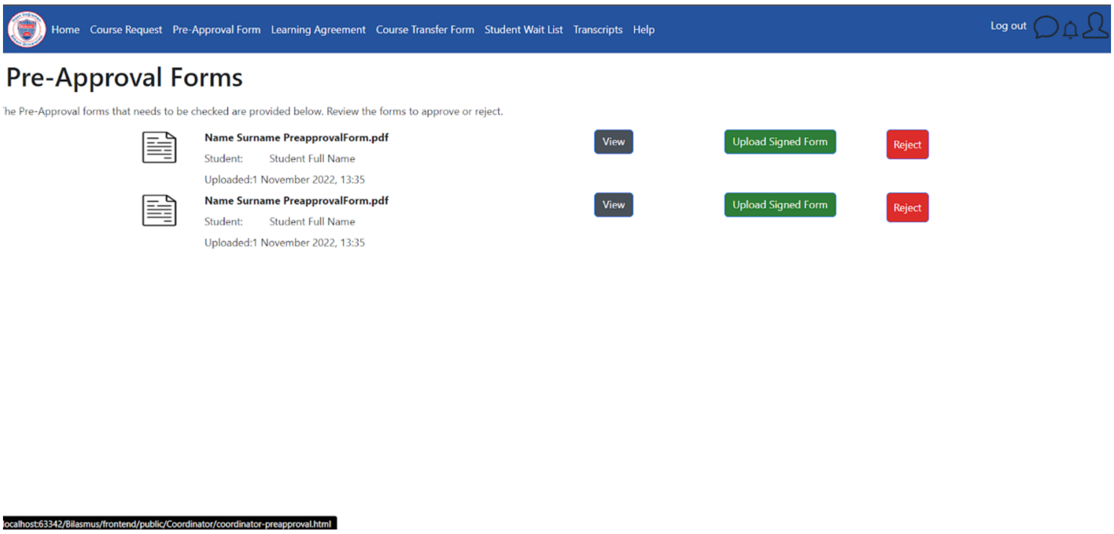

Figure 9: Coordinator Pre-Approval Forms Page

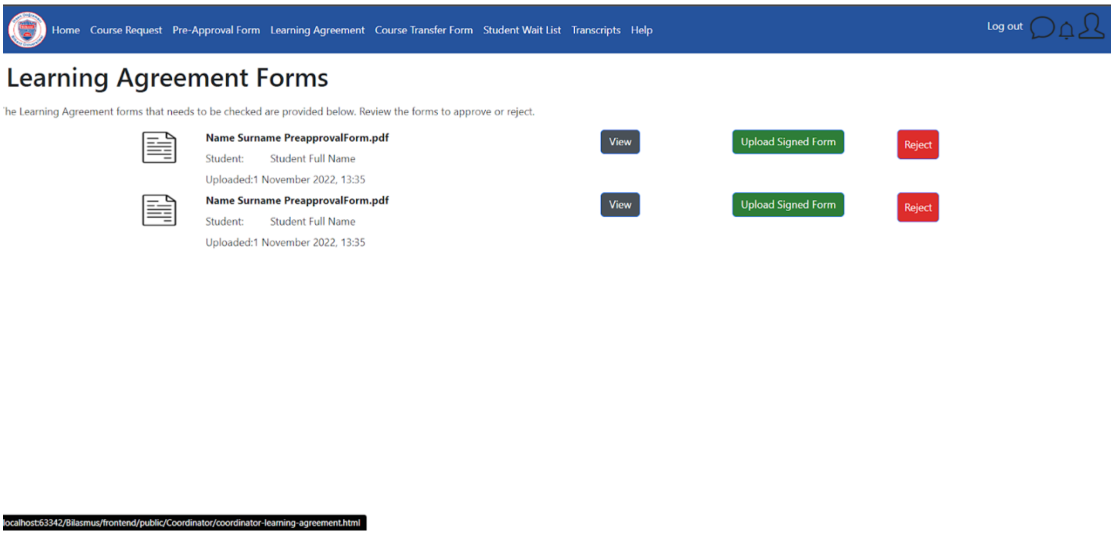

Figure 10: Coordinator Learning Agreement Forms Page

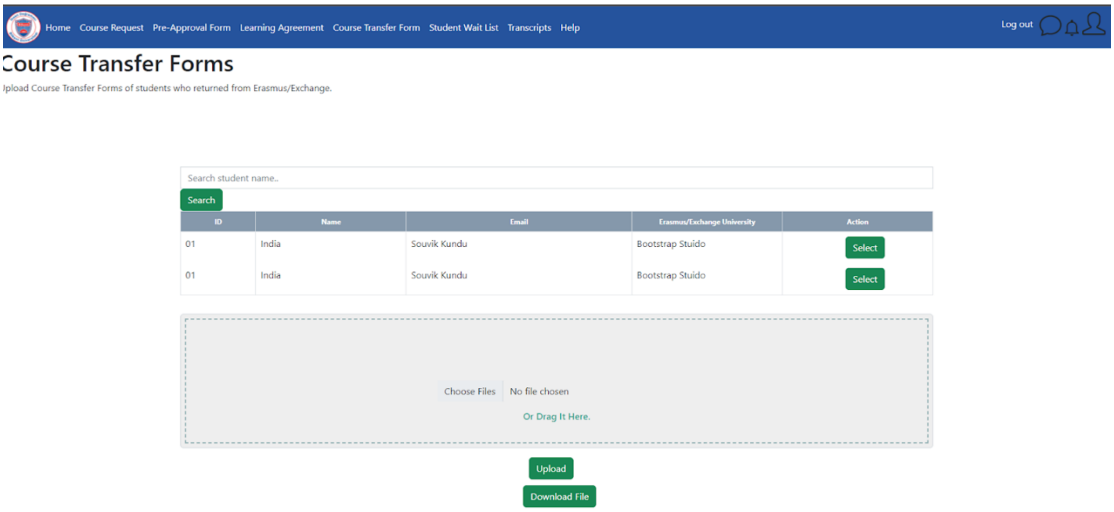

Figure 11: Coordinator Course Transfer Forms Page

Coordinator's Student Wait List Page displays a list of students that are not placed in an ERASMUS or Exchange program in an excel format.

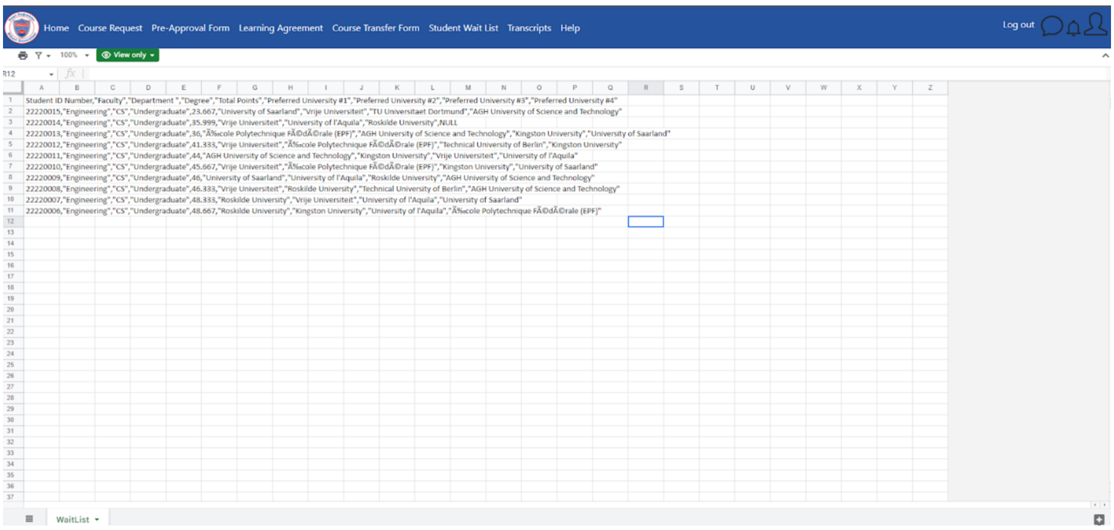

Figure 12: Coordinator Student Wait List Page

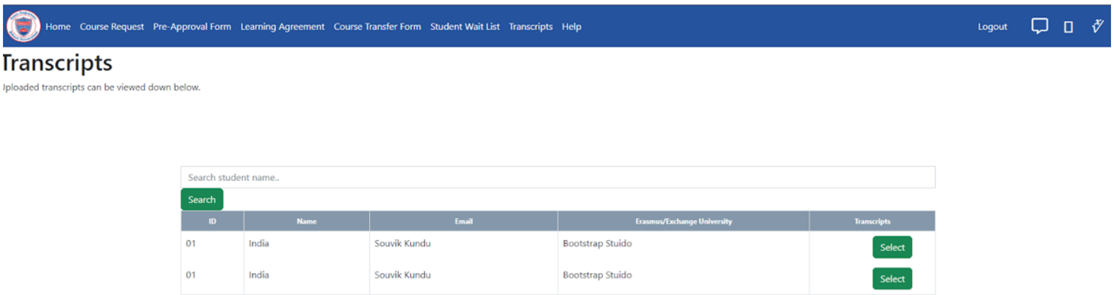

Figure 13: Coordinator Transcripts Page

3.3. Outgoing Student

Outgoing students can use Bilasmus system to create a course request list, upload and delete preapproval, learning agreement, and language proficiency forms.

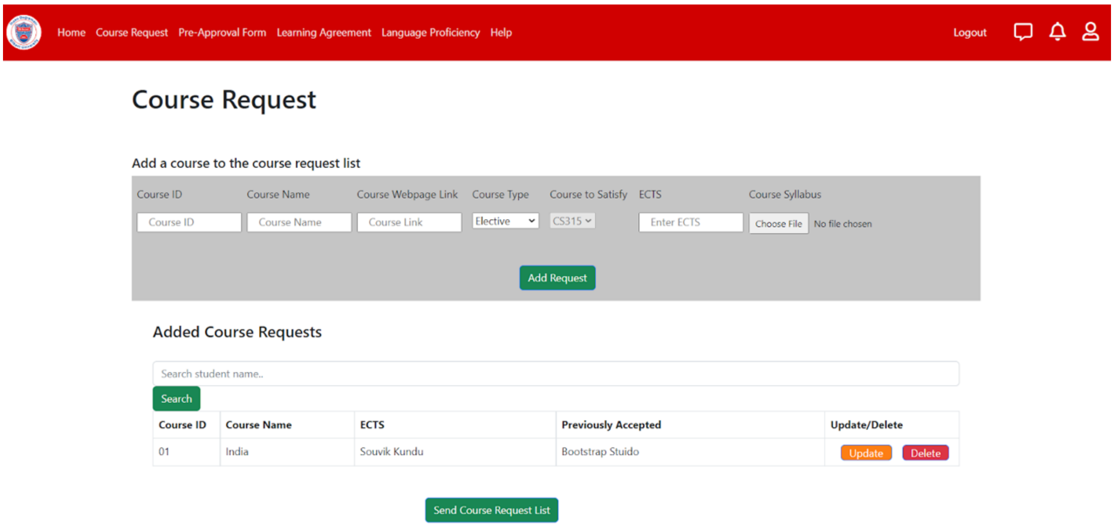

Figure 14: Outgoing Student Course Request Page

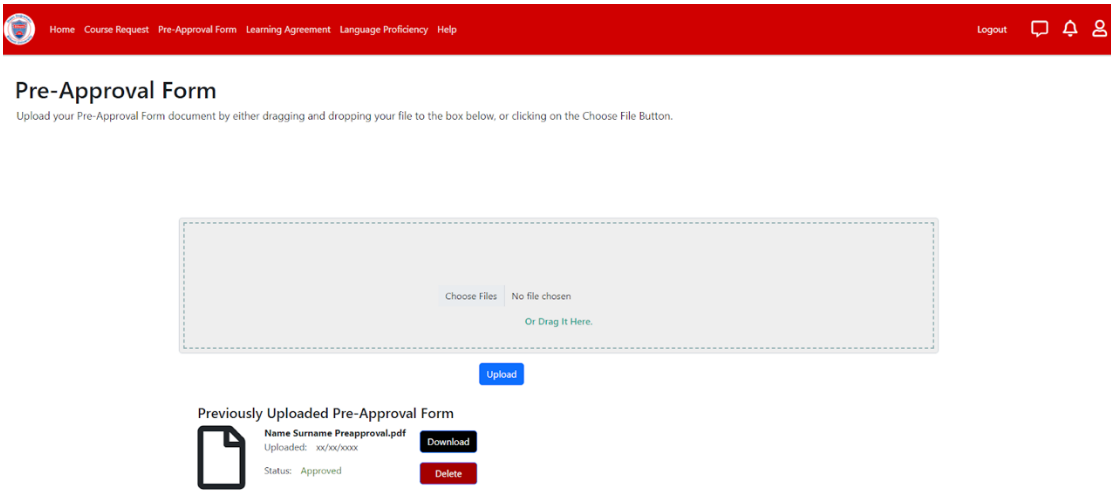

Figure 15: Outgoing Student Pre-Approval Page

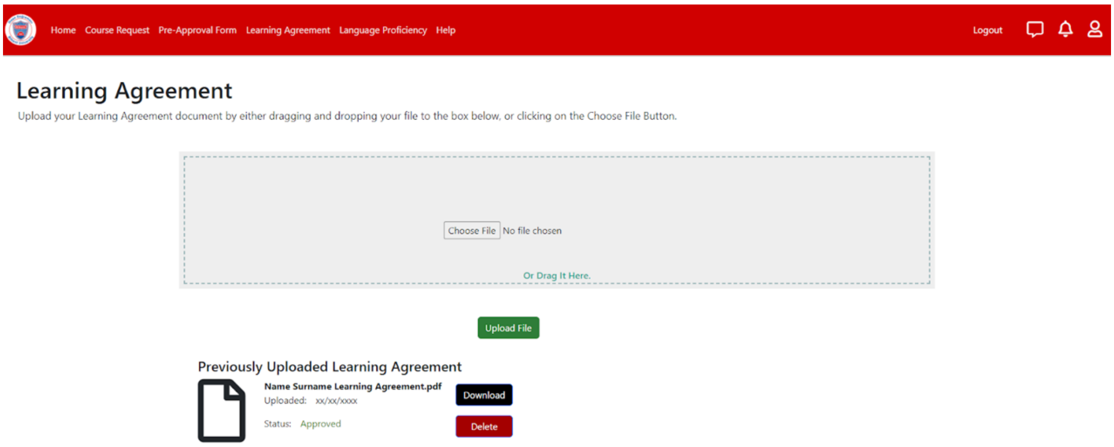

Figure 16: Outgoing Student Learning Agreement Page

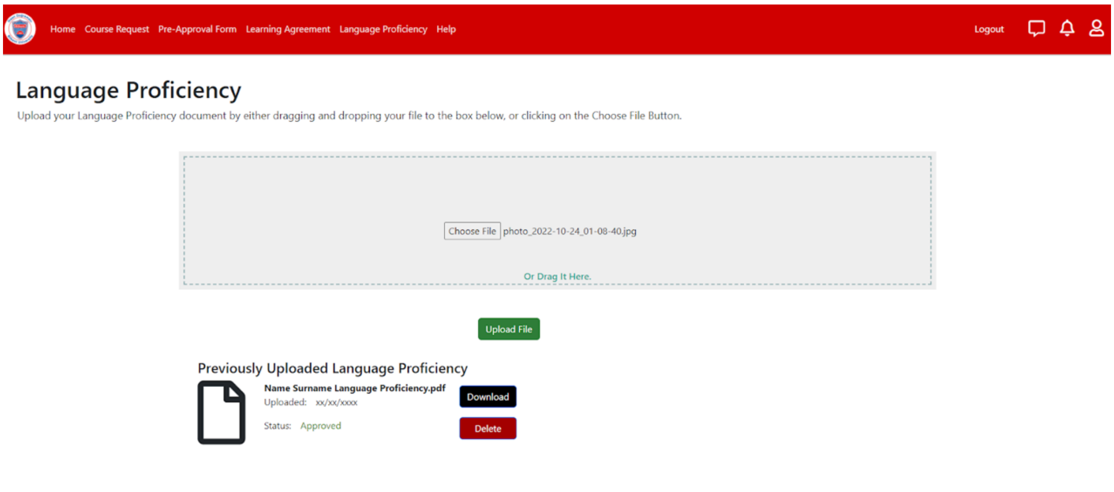

Figure 17: Outgoing Student Language Proficiency Page

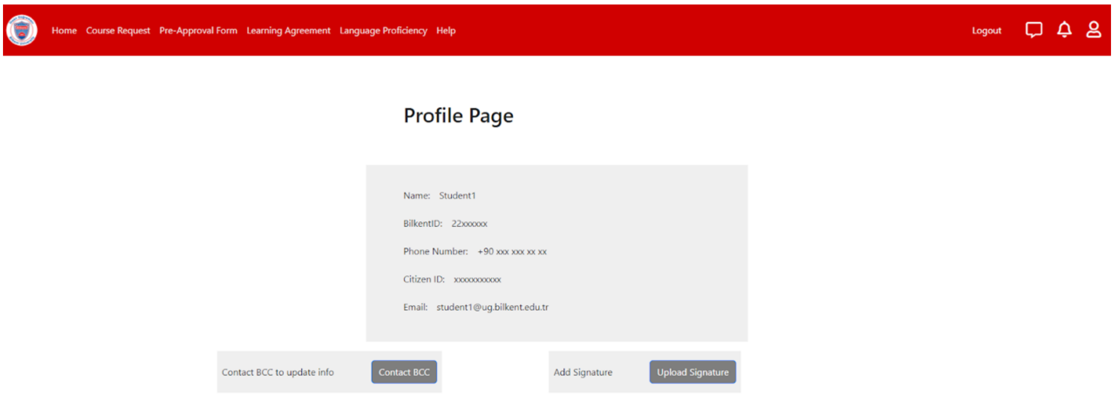

Figure 18: Outgoing Student Profile Page

3.4. Admin

Admin can add users to the system, update user information, start a maintenance by setting its start and end date, and delete users. They also upload transcripts of users that return form Erasmus and Exchange to the system.

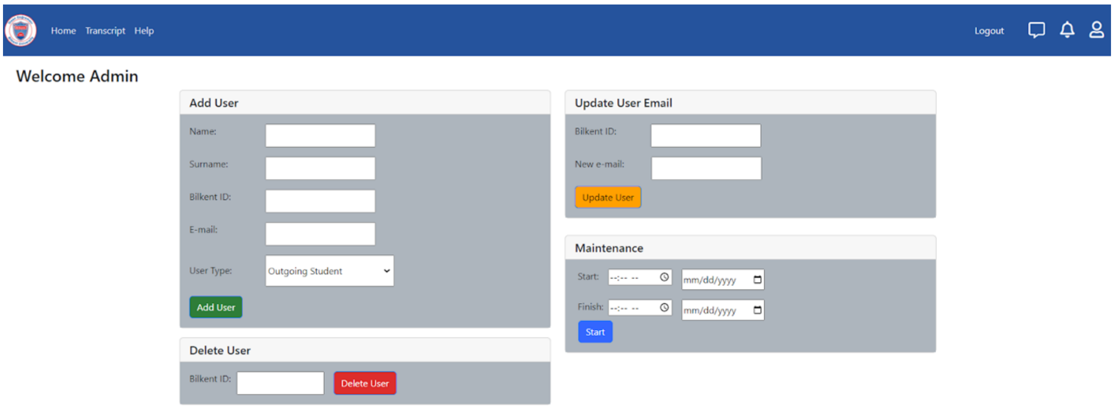

Figure 19: Admin Main Page

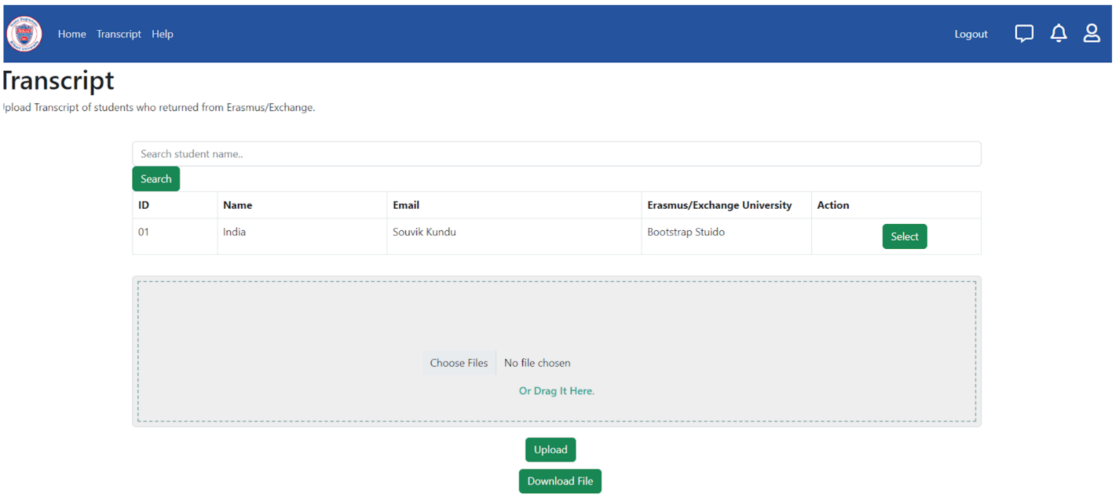

Figure 20: Admin Transcript Upload Page

3.5. Department Secretary

Department secretary can register courses approved by the coordinator.

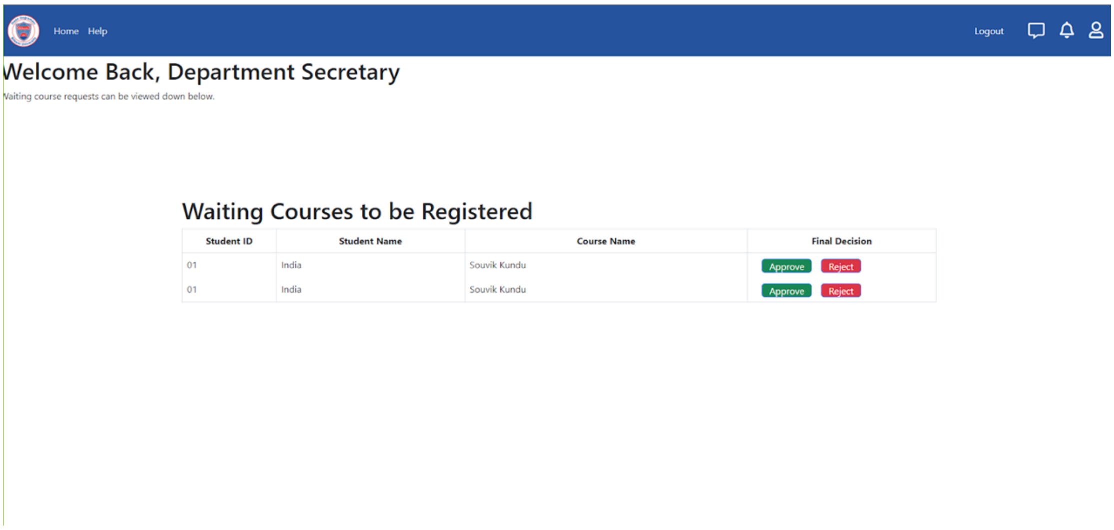

Figure 21: Department Secretary Main Page

3.6. Faculty Committee Executive Board

Faculty Committee Executive Board can approve and reject Pre-Approval and Course Transfer forms.

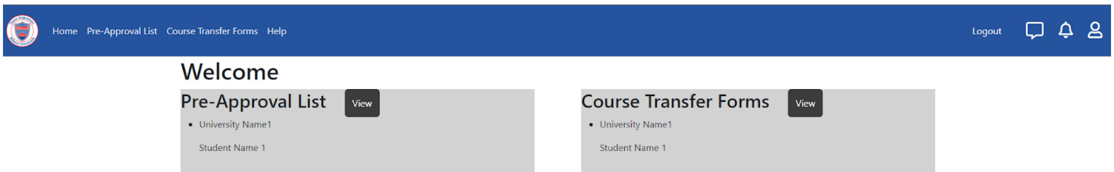

Figure 22: Faculty Committee Executive Board Main Page

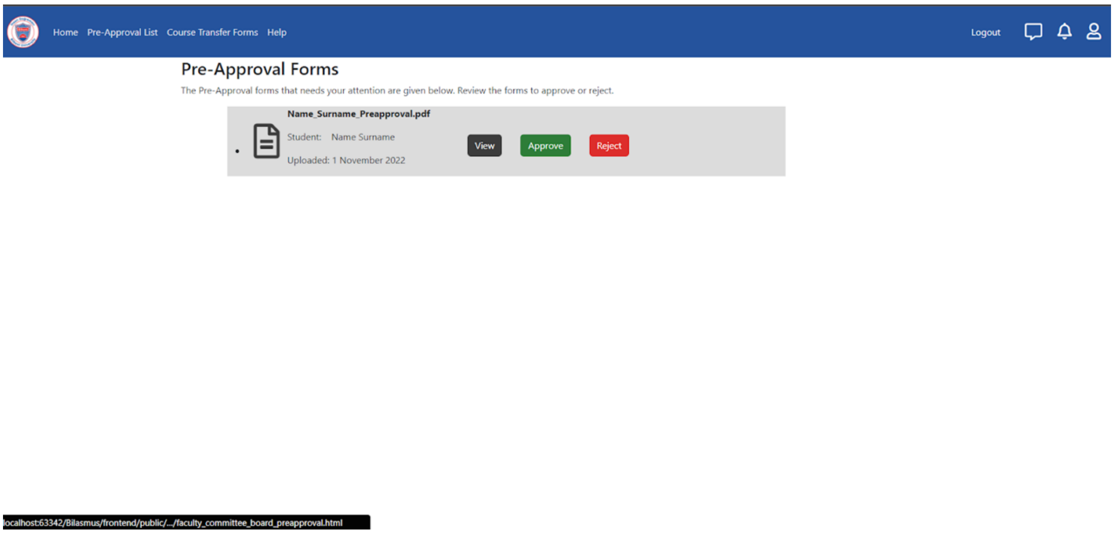

Figure 23: Faculty Committee Executive Board Pre-Approval Form Evaluate Page

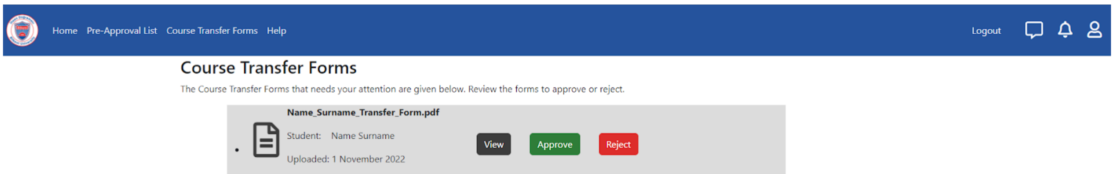

Figure 24: Faculty Committee Executive Board Course Transfer Form Evaluate Page

3.7. International Student Office

International Student Office can set a date for a chosen language exam. They are also responsible of approving or rejecting language proficiency exams done outside of Bilkent.

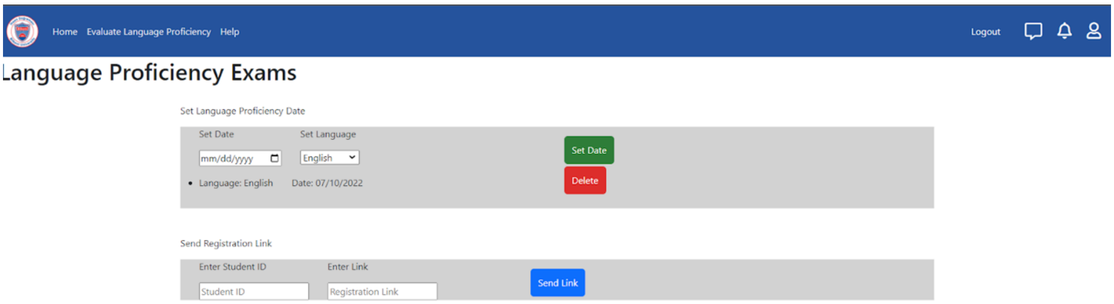

Figure 25: International Student Office Language Proficiency Exam Set Page

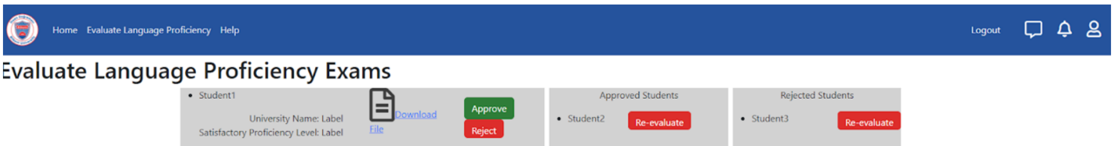

Figure 26: International Student Office Evaluate Language Proficiency Exam Page

3.8. Incoming Student

Incoming Students can use the system to create a course request list.

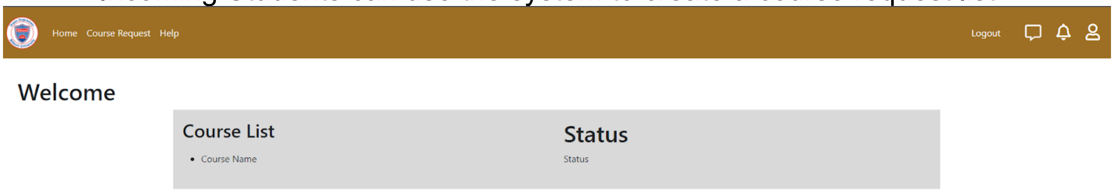

Figure 27: Incoming Student Main Page

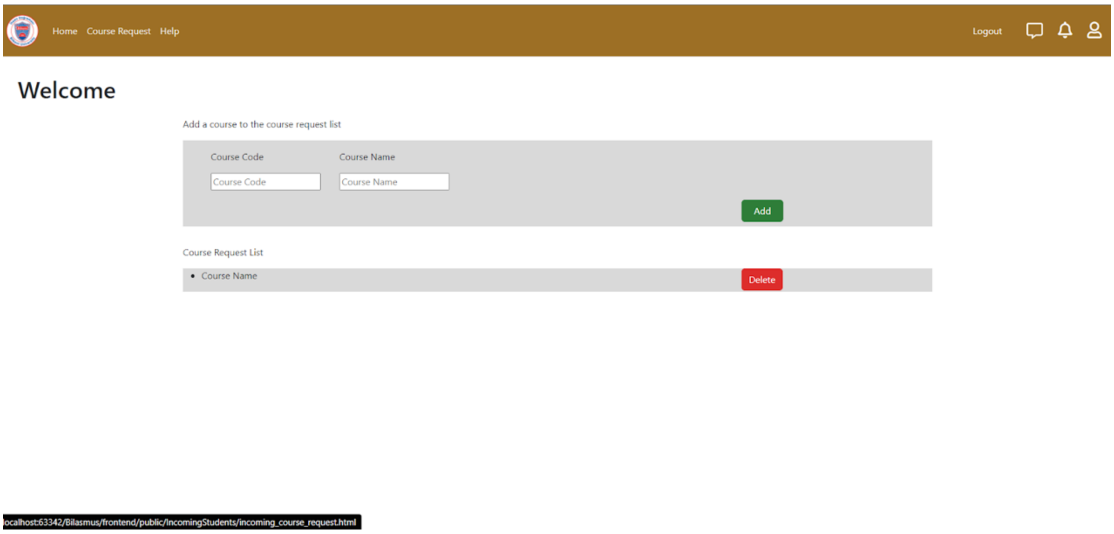

Figure 28: Incoming Student Create Course Request Page

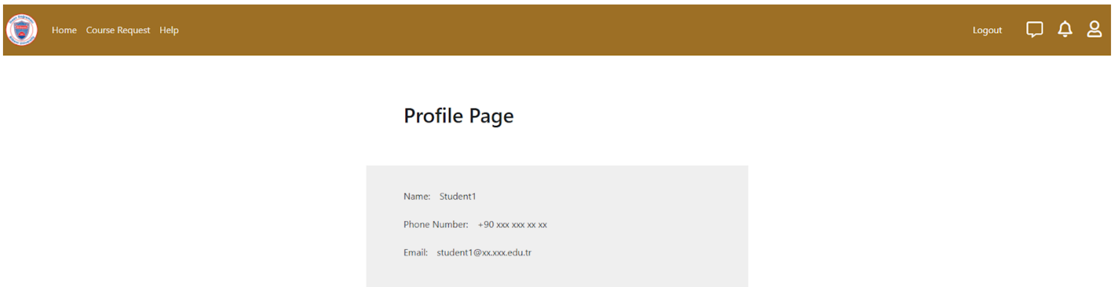

Figure 29: Incoming Student Profile Page

3.9. Instructor

Instructors can approve or reject a course to the system.

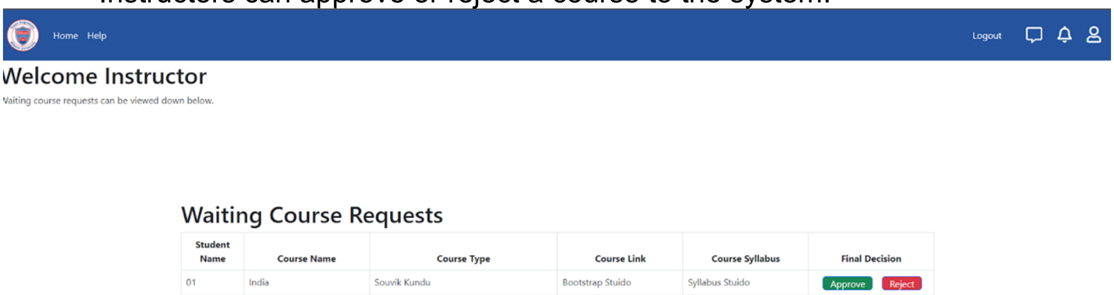

Figure 30: Instructor Main Page

4. Build Instructions

4.1. Build front-end

No JavaScript framework was used for the frontend of Bilasmus. However, Bootstrap 5.2 was used. Hence, Bootstrap 5.2 should be installed on the host machine.

4.2. Build back-end

Bilasmus is run on a local machine. To run the program, .js module “npm install” should be added. Node.js version 16 or higher is required to run on a local machine.

Note that backend, frontend, and database servers are deployed on cloud services. We are using one of the Google Cloud servers located in London, UK for backend and database. For frontend, we are using Firebase. When the system is working on cloud services, thanks to the continuous deployment (CD) set by GitHub Actions, all pushes to the system are deployed automatically to the cloud services. In case of an error, to deploy the program on a local machine, command “gcloud app deploy” should be run on the terminal.

We are using ElephantSQL (which uses PostgreSQL) service for cloud databases. Note that ElephantSQL is a paid service, and they have 5 query limitations at the same time. Because of that, for just now, only 5 users can use the program at the same time. After upgrading ElephantSQL service, hundreds of users can use the program simultaneously.

5. Work Allocation

5.1. Alp Afyonluoğlu

5.1.1. Analysis Report

- Determined and listed Functional Requirements.
- Sketched the Use Case Diagram with Emre.
- Modified the Class Diagram according to the feedback with Emre for the second iteration.
- Sketched the Login Page, Reset Password Page, Notifications Panel, Pre-Approval Form and Learning Agreement Pages for Outgoing Student, and Pre-Approval and Learning Agreement Pages for Coordinator and Faculty Committee Board

5.1.2. Design Report

- Wrote Initialization and Failure sections of Boundary Conditions with Emre.
- Modified the Final Object Design to implement design patterns with Emre and Erkin.
- Sketched the Web Server Management Layer.
- Determined and listed External Packages of backend and Internal Packages.

5.1.3. Implementation

- Implemented session management.
- For objects stored at the database, model classes were created according to the database structures created by Emre.
- Implemented model-to-query and query-to-model converter functions to use modals as DTO classes and access to database tables via model classes.
- Implemented the necessary CORS module and worked with Selin to ensure communication between backend and frontend.
- Set up Google Cloud App Engine for backend and hosting for frontend, made several functionalities, such as session management, work on both cloud and

local. Set up continuous deployment with GitHub Actions for both frontend and backend.

- Designed the main router structure and implemented some routers.
- Designed and implemented the error handling structure.
- Implemented controllers except the DatabaseController, only the model-related parts of which were implemented. Worked with Emre to implement some controllers', such as UserController, database-related parts.

5.1.4. Final Report

- Helped Emre for the Build Back-end section.

5.2. Emre Karataş

5.2.1. Analysis Report

- Drew use-case diagram and wrote use-case explanations with Alp.
- Sketched course-request page for coordinators, main page for admin and international student office, course transfer form for faculty board committee.
- For iteration 2, modified class and sequence diagram according to TA feedback with Alp.

5.2.2. Design Report

- Wrote the "persistent data management" section. Drew matrix diagram and wrote the "access control and security" section.
- Wrote boundary conditions with Alp.
- Wrote object design trade-offs.
- Drew final object design.
- Implemented design patterns to final object design with Alp and Erkin.
- Explained design patterns used in the project.
- Drew data management layer and entity classes.

5.2.3. Implementation

- Database tables created according to excel sheets provided by the client. ElephantSQL server is set for database actions.
- Closely work with Alp to connect DB to the backend code.
- Implemented design patterns.
- Implemented some of the routers.
- Helped to Alp in some of key classes such as login,register,emailService etc.

5.2.4. Final Report

- Wrote the " Lessons Learned" section.
- Wrote the "build-backend" section with Alp.

5.3. Erkin Aydın

5.3.1. Analysis Report

- Wrote the Reliability and Supportibility parts of Non-functional Requirements.
- Draw Activity, State, and Sequence Diagrams, along with Parsa Keihan.
- Made fixes in Use-Case and Class Diagrams along with every other team member.

- Draw Detailed Info, Department Secretary Main, Faculty Committee Main, Incoming Students Main, Outgoing Students Main, and Coordinator Main pages' illustrations for UI Design.

5.3.2. Design Report

- Wrote Hardware/Software mapping and Persistent Data Management parts. Persistent Data Management part was done along with Emre.
- Created Deployment Diagram along with Selin.
- Created Design Patterns in Final Object Design, along with Emre and Alp.

5.3.3. Implementation

- Created all of the pages of Outgoing Student, Incoming Student, Faculty Committee Board, and International Student Office actors.

5.3.4. Final Report

- Wrote Build Instructions of Frontend.
- Did User's Guide Along with Parsa

5.4. Parsa Keihan

5.4.1. Analysis Report

- Wrote the Performance and Usability parts of Non-functional Requirements.
- Draw Activity, State, and Sequence Diagrams, along with Erkin Aydın.
- Made fixes in Use-Case and Class Diagrams along with every other team members.
- Draw the sketch of Help, Messaging, Outgoing student's Language Proficiency page, Coordinator's course transfer forms Page, and international Student Office's language proficiency page illustrations for UI Design.

5.4.2. Design Report

- All parts of Introduction
- User Interface Management Layer and explanations with Selin
- Modifications in second iteration

5.4.3. Implementation

- Created UI of the pages of Log-in, Register, Reset Password, Admin, Department Secretary, and Instructor.
- Wrote JavaScript Codes for Log-in, Register, Reset Password, Admin, and Incoming students.
- Wrote and modified CSS and HTML Codes for Log-in, Register, Reset Password, Admin, and Incoming students.
- Integration of backend and frontend along with Alp.

5.4.4. Final Report

- Did User's Guide Along with Erkin

5.5. Selin Bahar Gündoğar

5.5.1. Analysis Report

- Wrote the introduction.
- Wrote the Overview section.
- Wrote the Class Diagram section and drew the Class Diagram.
- Designed the UI of Profile pages, Outgoing Student Course Request Page, Outgoing Student Main Page, and Incoming Student Course Request page.
- Put the UI mock-ups on the report.
- Proof-read, corrected, organized, and uploaded the Analysis Report.

5.5.2. Design Report

- Wrote the Subsystem Decomposition section and drew the 3-Layer Architecture Diagram.
- Drew the Deployment Diagram and wrote the Deployment Diagram sub-section.
- Wrote the User Interface Management Layer with Parsa and drew the User Interface Management Layer Diagram.
- Wrote the Internal Packages sub-section.
- Proof-read, corrected, organized, and uploaded the Design Report.

5.5.3. Implementation

- Prepared the backbone code of front-end
- Designed the UI of Coordinator, Instructor, Department Secretary, International Student classes.
- Wrote functionalities of Coordinator, Instructor, Department Secretary, International Student classes in the front-end.
- Integrated the backend with the frontend pages I coded along with Alp.

5.5.4. Final Report

- Wrote Work Allocation of my part.
- Proof-read, corrected, organized, and uploaded the Final Report.