Bilkent University

Department of Computer Engineering

CS 319 TERM PROJECT

Section 2

ErasmusAga

Group 2B

Project name: Easy Erasmus

Project Final Report

Doruk Altan 21401362 doruk.altan@ug.bilkent.edu.tr

Asım Bilal Ak 21802887 bilal.ak@ug.bilkent.edu.tr

Gökberk Altıparmak 21901798 g.altiparmak@ug.bilkent.edu.tr

Furkan Yıldırım 21902514 f.yildirim@ug.bilkent.edu.tr

Alp Tuğrul Ağçalı 21801799 tugrul.agcali@ug.bilkent.edu.tr

Instructor: Eray Tüzün

Teaching Assistant: İdil Hanhan

1.Implementation Process

As the first step in beginning the implementation process, we decided to identify the different kinds of software we will use and familiarize ourselves with their use. For instance, we made use of IntelliJ IDEA, GitHub and CUBA platform. All members of the group met and installed the required software and connected to the main repository in our project GitHub. The next step was to integrate GitHub with IntelliJ and add the CUBA plugin. At this point, every member of the group was working on the same platforms, was able to push and pull code from other members and was somewhat familiar with the software. Another measure we took as soon as possible was to create communication channels such as whatsApp group chats and Discord channels for the project so we can constantly keep in touch and be updated about the project.

Once the setup process was complete, we divided the work among ourselves and set deadlines for tasks. We kept in mind that we should have a few minor deadlines for each task before the actual project deadline in order to manage our time better and prevent leaving everything to a last minute ditch effort. The three tier architecture of our project led us to decide to assign tasks based on these tiers. In other words, one to two people would be working on each tier which are the application logic, data management and user interface tiers. Furthermore, every member also contributed to the documentation. Members with previous experience on certain subjects were assigned related tasks to make everything smoother and faster. However, the dividing of tasks did not mean that the members only worked on their respective tasks. We understood that this is a team effort and we helped each other whenever a member asked for it or had extra time to help others. This approach helped us a lot in the end as everyone had an area that they excelled in but still had an understanding of every part of the project.

At the beginning of the project, Alp Tuğrul Ağçalı, one of the members of the group, created the project and first designed the left menu and then the background to be used on the screens created. Then he created the necessary entities and gave them the necessary attributes, and imported the necessary data from the excel files provided by the school so that the group could work with the correct data. Later, he created the application

management screen and added the feature of importing the excel, which contains the applications, and the function of placing the winning applications to this screen. He allowed us to use our own exception dialog by writing the AGAExceptionDialog, AGAExceptionHandler and AGARuntimeException classes. To make it easier for us to get State Type objects, he created the AgaConfig interface, but instead we decided to use a query later. Then he made the FileUpload screen. He helped this screen to be used in the Edit Application screen, in the comment feature, and in the Add Course screen. He added the Comment feature when he created the Edit Application with Asım Bilal Ak. Added the feature of adding files to the comment. He created fragments named comment fragment and file fragment in order to display the added comments on the screen, and enabled them to be used on the edit application screen. Together with Gökberk Altıparmak, he created TO-DO screens to be used by the course coordinator and administrator and added the isSeen checkBox to the Edit application. He created the List Application Screen where all Applications are listed.

Doruk made the use case scenarios and sequence diagrams in the analysis report. System architecture part in the design report that included subsystem decomposition, hardware/software mapping, boundary conditions, etc. and the implementation of profile screens for students, coordinators, administrators.

Gökberk made the implementation of listing all the universities, editing university information, page to show university information, To-do page of course coordinator and admin, adding new courses to university, page to show information of course, login page.

Asım, for implementation he created an edit application screen's student information front-end and back-end. Also different persons have different responsibilities and different roles so all of them are done by him.

Furkan made the implementation of UniversityList, EditUniversityList and AddCourse screens with Gökberk.

2.Changes & Improvements

In our implementation there are few changes which are not important. We strictly wrote our code depending on our design patterns and our diagrams.

In addition to code, we decided to change our UI design because it did not look professional so we changed our login page. Also, all comment text fields were made bigger by our team.

3. Status of Implementation

We have fulfilled our functional requirements. We have been faithful to our design goals from the analysis and design stages. As for the non-functional requirements, we believe our application has a very self-explanatory interface which makes it user-friendly. We restricted access of all users to only deal with their related data except the admins to avoid any accidental data changes while still offering the users all features, they need to use the application successfully in order to keep our reliability constraints. All operating systems and web browsers are compatible with the application, so we also fulfilled our supportability requirement. But there is still room to improve of course, one minor extra feature we hoped to deliver but couldn't due to time constraints is being able to import an excel file by dragging and dropping it to the upload file screen. As of right now, users have to enter the path of the file to import it. Aside from minor features and potential optimization, we have delivered a working product that satisfied the requirements.

4. User's manual

3.1. System Requirements

ErasmusAga runs on a web server, therefore there is no need for any installation or hardware requirements. However, a user needs a working internet connection and a web browser. All popular browsers like Google Chrome, Mozilla Firefox, Safari, Opera 15+, Internet Explorer 11, Microsoft Edge are supported. If a browser is extremely outdated, it may be incompatible with some APIs and libraries, therefore older browsers like Internet Explorer 8 might not support all features.

3.2. Overview of the Application

3.2.1. Login

The first screen a user will interact with is the login screen. In order to log in, the user has to enter his credentials in the respective text fields. The application then checks if there is a matching account with those credentials.

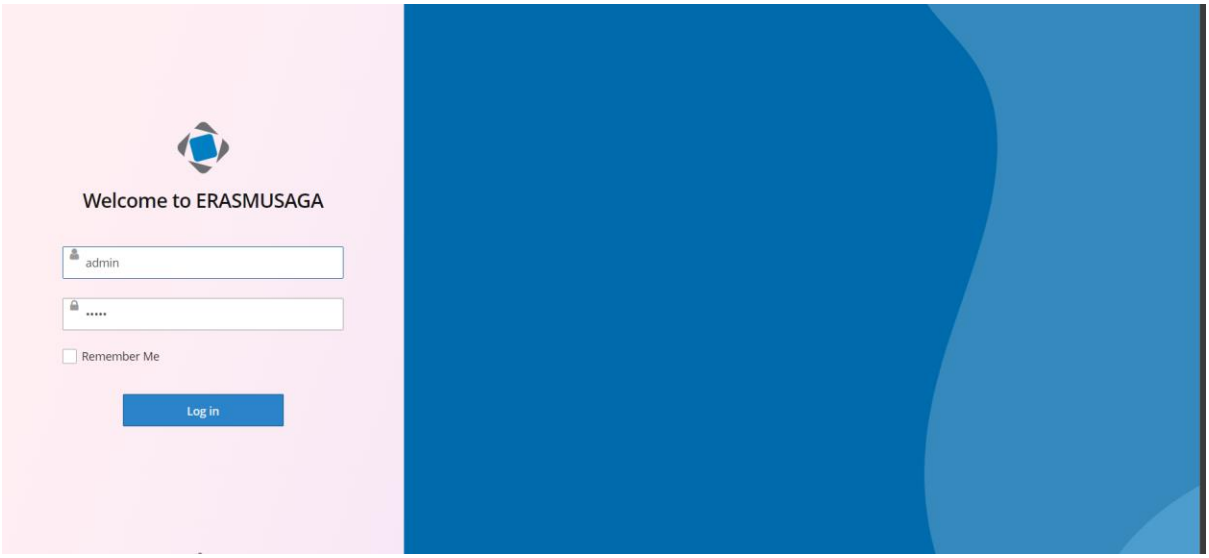

Figure 3.2.1.1 (Login page)

3.2.2. Main Page

If login is successful, the user is directed to the main screen depending on his role (Student, Coordinator, Administrator). Users with different roles have different permissions. A student will only be able to see Student Profile and Application screens. Whereas a coordinator will see Coordinator Profile, University List and To-Do list screens.

Figure 3.2.2.1 (Main page of administrator)

3.2.3. Leave Comments

The main communication channel of the application are comments. Authorized personnel will communicate with students through comments regarding their application status or course approvals. Also, students can ask for information or feedback via comments. A student can leave a comment from the edit application screen. That comment will then go to the To-Do list of a coordinator. The coordinator will be able to set the comment as seen and if needed can leave his comment for the student. Once a comment is set as seen it will be dropped from the To-Do list.

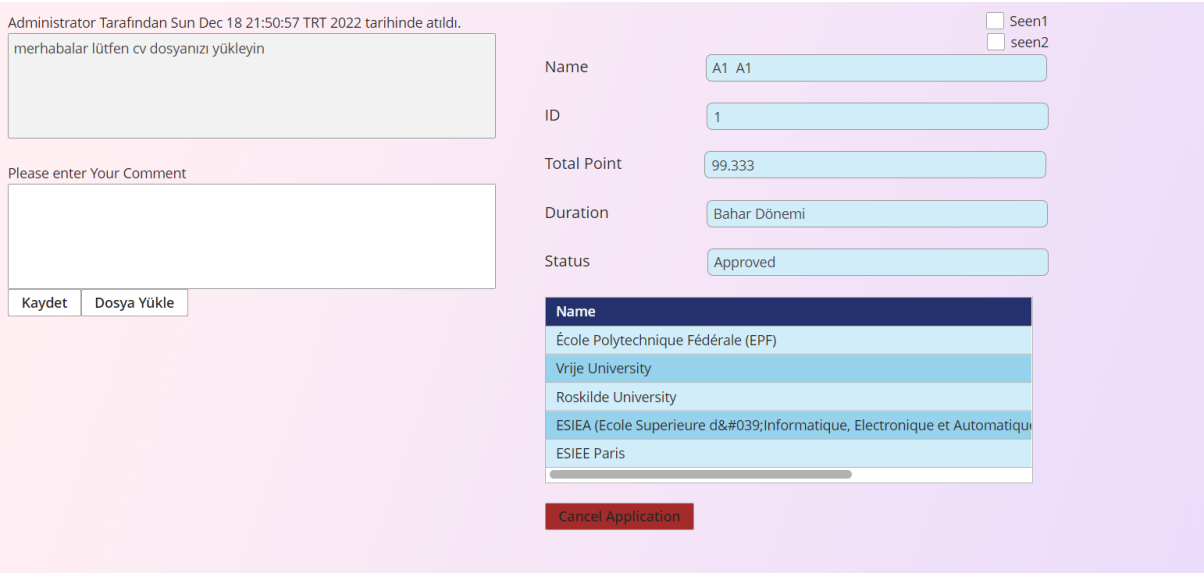

Figure 3.2.3.1 (administrator comment and file upload page)

3.2.4. Upload Files

Students can upload syllabus files for approval. They can access the file upload feature from the Edit Application screen. They will then select the file and click the Upload File button. Administrators and Coordinators can use this feature as well. They select an application from the application list or the To-Do list and upload files with the same steps as the student. All files uploaded for a specific application will be visible in the application page sorted by upload date.

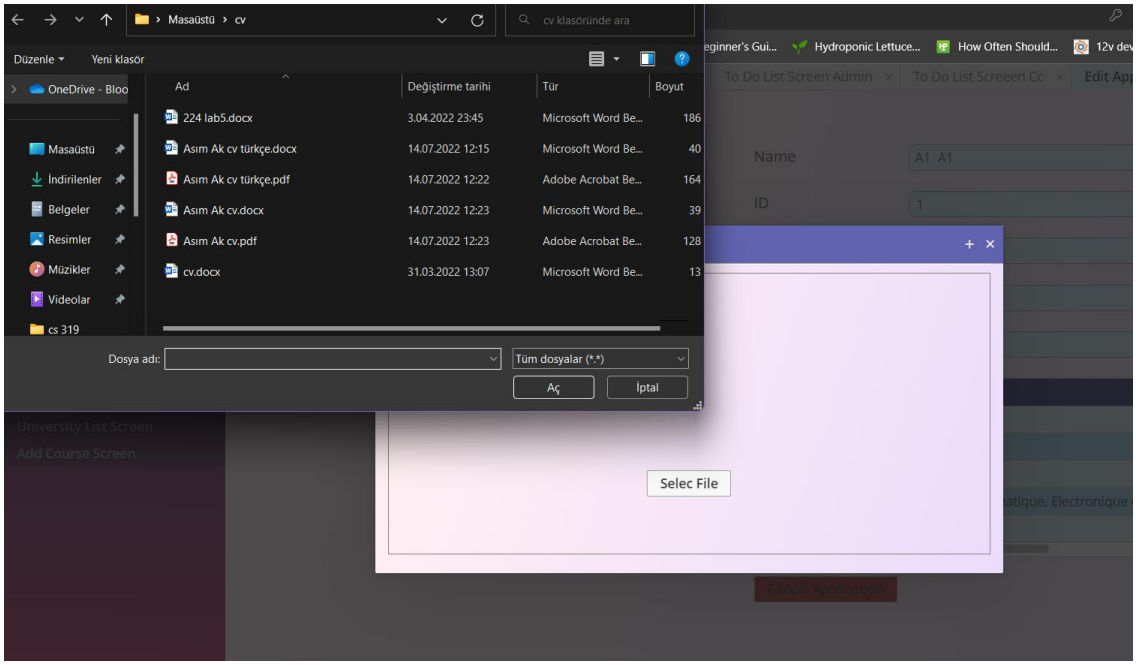

Figure 3.2.4.1 (Selecting wanted file)

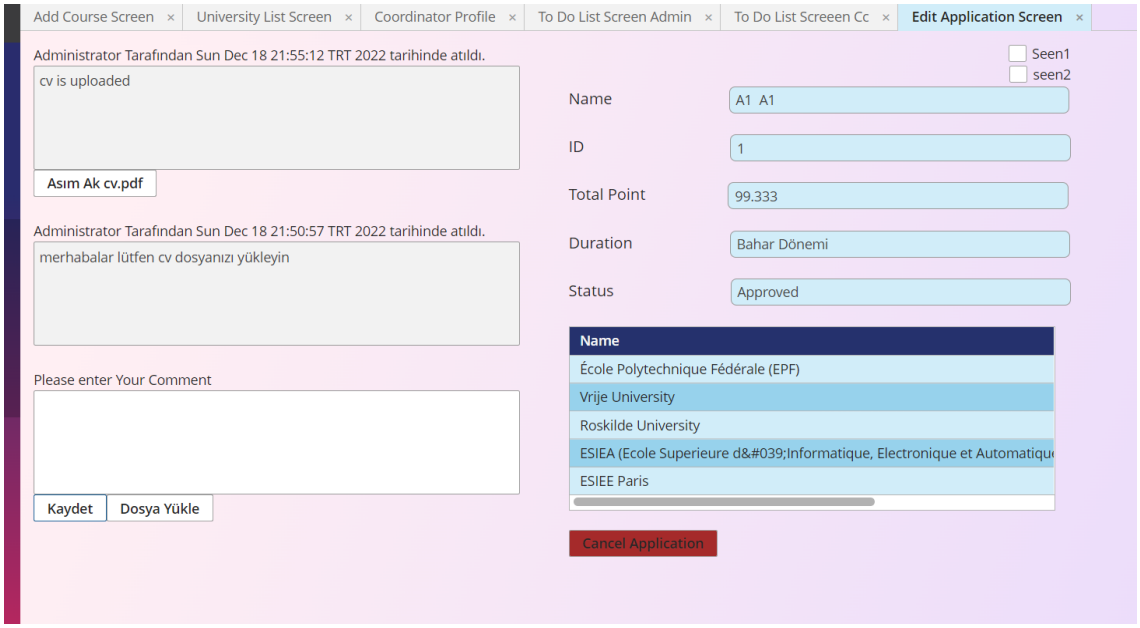

Figure 3.2.4.2 (uploaded file)