Bilkent University

Department of Computer Engineering

CS 319 Term Project

ErasmusNET

Group 3F

Arda Baktır, Beyza Çağlar, Gülin Çetinus, Mehmet Kağan İlbak, Zeynep Selcen Öztunç

Instructor: Eray Tüzün
Teaching Assistant(s): Metehan Saçakçı, Emre Sülün, Muhammed Umair Ahmed, İdil Hanhan and Mert Kara

Final Report

December 18, 2022

Table of Contents

1. Introduction

ErasmusNET is a web-based manager for the Erasmus program, aiming to aid especially the outgoing students and the department coordinators with their responsibilities. The system's primary purpose is to make every part of the Erasmus procedure more accessible by uniting the essential operations the students and the coordinators have to perform so that every problem can be dealt with in one place. For instance, the students can make an appointment with the coordinators to discuss any issues they may have during the application process. The coordinators can directly resolve the student's problem or arrange a meeting. They can also see and approve all the outgoing students' pre-approval forms in one place without the risk of them being scattered or lost in the hundreds of emails sent to the coordinators.

2. Lessons Learned

The project's front-end is implemented with React and Tailwind CSS and NextUI, and we have used Visual Studio code as an IDE. The team members were unfamiliar with React, although Zeynep had experience working with Vue to implement user interfaces. At first, we decided that Beyza and Zeynep would be responsible for the program's front-end. However, more than two people might be needed for UI implementation shortly after the analysis report. The back-end team decided to help with the front-end if the implementation could not be completed.

One of the main lessons we have learned was to start working earlier. The front-end took little time to finish. Every group member helped with the front-end because we believed we would need more time to complete it. The back-end was more problematic as we prioritized the front.

We experienced the benefits and disadvantages of working in a group as well. As we tried to divide the workload evenly, every member had less work than they would if they were working alone. We communicated with each other regularly to ensure everyone could complete their responsibilities. Along with face-to-face meetings, we have used Discord and WhatsApp to let each other know if we had questions. Keeping in touch or arranging a time when everyone was available became difficult at times.

The project's back-end is implemented with NestJS, a NodeJS-based framework. The reason for this choice is that NestJS is an easily adaptable framework, and since we all are in the project's front-end, using JavaScript, it would be easy for everyone to check the back-end.

The project's database is implemented with Google Firebase since it is a simple-to-use database. Some of us did not know, and some were not confident about handling the database by hand; hence we chose to use Google Firebase.

We faced various challenges during implementation. This is mostly due to design choices we agreed that later turned out to be blockers during implementation. We could easily avoid those challenges if we began implementing them before writing reports.

3. Implementation Status

The implementation of the project started after finishing the class diagram. Each group member used VS Code as an IDE and we all cloned the GitHub repository to our computers. Each group member created a branch when adding new pages or features. Each group member committed their implementation changes to that branch and with the review of one group member, we merged the branches to the main. Work allocation was done by creating issues about the feature or part we are starting and closing the issue after we finish and push the code.

Things left to do:

1. We originally planned to handle everything online, including getting signatures for forms. In addition, we'd generate the forms so that nothing is printed.
2. Some labels in the Next Step section are not implemented since they were user specific.
3. The Contact Us section in the landing page doesn't do anything. It was supposed to send an email to the Admin.
4. While submitting this report, signup has no purpose since every user will be created by admin.

4. Build Instructions

There are two main build instructions that the project can run. First one is development mode and the second one is production mode. Development mode displays some debug information to the top of the code. In addition, development mode sets the API IP to localhost. Production mode is the one that will face users. It sets the API IP to AWS instance so that data will be fetched from the backend that runs on AWS. There is also a stag mode that is essentially production but API IP is set to localhost just like in the development mode.

In addition, the website is live at https://www.erasmusnet.app which you can log in with credentials given below.

The project will run best on MacOS and Linux systems. These are the instructions for such systems:

1- Make sure that NodeJS and npm are installed.
2- git clone "https://github.com/cs319-erasmus/erasmusnet.git"
3- cd erasmusnet/src
4- npm install

You can run the frontend of the system with:
5- npm run
5- npm run start:dev
5- npm run start:stag

You can run the backend of the system with:
6- cd erasmus-net-api
7- npm install
8- npm start
8- npm run start:dev

Authentication is handled by Firebase, and these are the test accounts that can be used:
Student: student@test.com:securepassword
Coordinator: coordinator@test.com:securepassword
Instructor: instructor@test.com:securepassword
Admin: admin@test.com:securepassword

5. Changes and Improvements

At the beginning of the project, we designed systems for the students to apply to the Erasmus program. For example, a Placement Quiz feature was designed to help students choose the school they might attend. However, we realized that the application process was out of the project's scope and removed them from the final implementation. The Student Stories page is not implemented in the final design either.

The coordinators no longer have the authority to accept or reject syllabi sent by the students. After the instructors and the pre approval form approve, all the courses are generated and sent to the coordinators, who can approve or reject the form.
The International Office employees do not need to use the program. The placement list can be uploaded to the system by the admin.

6. User's Guide

1. Sign In

Before a user can use the program, they have to sign in. The process is the same regardless of the user's type, meaning that the coordinators, the instructors, and the students all sign in through the same page.

2. Login

If a user has signed in to the program, they can log in with their e-mail address and the password they have entered during the sign-in process.

3. Personal Information

The users can view and change their personal information on their profile page. The information includes the user's full name, e-mail, department, current semester, student id and university name.

4. Course Counting

The students have to form their preapproval form with approved courses. They can upload the syllabus of a lesson they would like to get approval on. The instructor responsible for that course in Bilkent will examine the provided documents. If the course is suitable, the instructor can approve the syllabus; otherwise, they can reject it.

5. Appointments

If students need help with any part of the process, they can request an appointment from their department coordinator(s). The topic of the problem should be explained in as much detail as possible. If a meeting is necessary or the problem can be solved with a short message, the coordinator can approve the appointment and write the student a message during approval. If a meeting is not needed, they can reject the appointment request.

6. **Preapproval Form**

   After instructors approve all the courses that the student will take during their Erasmus semester, the preapproval form will be generated. The process is complete for the student when the coordinator approves their preapproval form.

7. Work Allocation

Arda Baktır

Analysis Report
- Object and Class Model
- Improvement Summary
- Web Server Layer

Design Report
- Deployment Diagram
- User Interface Management Layer
- Improvement Summary

Implementation
- All of the back-end of the project
- Connecting Front-end and Back-end of the project

Beyza Çağlar

Analysis Report
- Functional Requirements
- Statechart Diagrams
- Sequence Diagrams
- Improvement Summary

Design Report
- Introduction
- High-Level Software Architecture
- Improvement Summary

Implementation
- Front-end of the project

Gülin Çetinus

Analysis Report

- Introduction
- Use Case Diagrams
- Improvement Summary

Design Report

- Object Design Trade-offs
- Web Server Layer
- Improvement Summary

Implementation

- Front-end of the project

Mehmet Kağan İlbak

Analysis Report

- Non-functional Requirements
- Pseudo Requirements
- UI Mock-ups
- Improvement Summary

Design Report

- Subsystem Decomposition
- Final Object Design
- UI Layer
- Data Management Layer
- Packages
- Improvement Summary

Implementation

- Front-end of the project
- Connecting Front-end and Back-end of the project
- Project management
- Code Reviews
- Deployment and CI/CD

Zeynep Selcen Öztunç

Analysis Report

- Current System
- Activity Diagram
- Sequence Diagrams
- Improvement Summary

Design Report

- High-Level Software Architecture
- Improvement Summary

Implementation

- Front-end of the project