Erasmus Registration Management System

Project short-name: ERSMS (Group Number: 1E)

Final Report

Kutay Tire - 22001787
Atak Talay Yücel - 21901636
Yiğit Yalın - 22002178
Borga Haktan Bilen - 22002733
Berk Çakar - 22003021

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

December 18, 2022

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Object-Oriented Software Engineering course CS319
1 Introduction

The final form of the ERSMS software offers different screens and functionalities for students, exchange coordinators, course instructors, chairs, deans, employees of office of international students and exchange programs (OISEP) and finally admins. All of the key features and ideas were implemented in the app to ensure the customers' expectations are satisfied. For a student to register to the system, an employee from OISEP has to upload the score table of the corresponding department as an initial step. Then, the exchange coordinator needs to place the students via the “Auto-Place” button on his/her “Placements” page. After that, students can register through the “Sign Up” page by activating their accounts.

The primary method of interaction is through forms and requests that are present in digitalized forms in the system. Students can create and upload pre-approval forms and course equivalence requests whereas exchange coordinators can upload CTE forms from the official transcript provided them by the host schools. These forms will be evaluated by corresponding administrative staff and their statues can also be tracked through the system. There are also many bonus features like To-Do lists, announcement panels, profile pages for students, loggings of previous forms, charts for displaying overall forms and request data, etc.

1.1 State of the System

- Although the back-end and the front-end of the messaging functionality are implemented, they are not connected.
- There is a page for “Appointments” for students and exchange coordinators, but its backend is not implemented.
- There are some small delays in the system especially when fetching large data from the database.
- The functionality for manually placing students has not been implemented.

2 Lessons Learned

First of all, we learnt how to manage a team by coordinating meetings and doing the proper work allocation. This was crucial because we were able to report back to each other and monitor each other’s processes. Furthermore, giving enough time for designing the structure of the backend pays off really well. Although we spent a lot of time thinking about the relations between classes and how they are going to be implemented, we were able to act on changes and modify our code much more easily later on.
Another crucial lesson we were able to pick up was the significance of requirement analysis. Developing an Erasmus management software requires a lot of brainstorming. There were lots of attributes that could be added, but there was simply not enough time. As a result, we had to filter some qualities and add the ones that are the most important for the client. We also understood the significance of design patterns. Writing the code in a rigid way made our jobs really difficult in the first place. To overcome this, we adapted some design patterns so that we could improve and extend our code.

For languages, many of us did not have prior knowledge about .NET or Angular. We also needed a strategy to learn these languages as quickly as possible and learn the appropriate parts. This was a valuable lesson because trying to be expert on every aspect of a language would cost us precious time. We adapted ourselves to learn and use the relatively important parts for our project. Finally, we experienced many conveniences when we applied commonly used software engineering principles and practices such as AutoMapper library for DTOs, REST API architecture for endpoints and N-layered architecture for the general structure of the project.

3 User’s Guide

3.1 Admin

The information of the admin account is hard-coded into the database. Therefore, there isn’t any way to create an admin account throughout the app. Admin is responsible for creating the accounts of the administrative staff and students. However, creating a student account is optional for the admin as student accounts are automatically created when the exchange coordinators place the students.

There is only an admin panel in the admin page. Through this panel, the admin chooses the user type such as a chair or a dean and enters the necessary information of the user. This information includes the name, last name, email, Bilkent id number, password and confirmation password. Admins can also delete a user or modify the information of a user through the admin panel.
3.2 Dean

Dean can enter the system after his/her account has been created by the admin. The dean account contains three pages which are “Logging”, “Dashboard” and “Forms and Requests”. The “Logging” page includes the previously sent CTE forms as the signature of the dean is required only for the CTE forms. However, they can only see the forms of their corresponding faculty. They can also filter the loggings by entering the filtering value and can sort the data of a corresponding column if they require.

For the “Forms and Requests” part, they can either approve or decline a CTE form. Naturally, they are able to see the previous decisions of the other administrative staff such as the chair or the exchange coordinator. They can provide additional notes regarding their decision which will be displayed to the student. Before making any decision, they can monitor the courses and the corresponding grades in the CTE form and the general information of the student such as his/her CGPA, host school or email address. Finally, they can see the announcements through their dashboard.

3.3 Chair

The account of the chair is almost identical to the account of the dean. The only difference for the chair is a chair can only see the CTE forms of his/her corresponding department rather than the faculty which was the case for the dean.

3.4 Course Instructor

Course Instructors can enter the system after their accounts have been created by the admin. The course coordinators have two main pages in their view, namely “Forms and Requests” and “Logging.” In the “Forms and Requests” page, they can view and respond to the course equivalence requests related to their courses. In the “Logging” page, they can view all the course equivalence requests related to their courses, including the canceled and archived ones.

3.5 Exchange Coordinator

Exchange coordinators can enter the system after their accounts have been created by the admin. The exchange coordinators can manage the ERASMUS process of their departments. They have 6 main pages in the exchange coordinators’ view, namely “Dashboard,” “Forms and Requests,” “Appointments,” “Logging,” “Placements,” “Messages.” In the dashboard, they can see the overview of the forms and requests via charts. They also have a todo list which is updated automatically if there are any updates on form submissions. They also can see the announcements made by deans, department chairs, course coordinators, instructors and exchange coordinators in the dashboard. In the “Forms and Requests” page, the exchange
coordinators can create CTE forms. They also can view and respond to all the active forms and requests in the system. In the “Logging” page, the coordinators can see all the forms and requests regardless of them being in process, i.e., archived and canceled forms are also included in this page. In the “Placements” page, the coordinates can see the students that are placed and that are in the waiting list. They also can see a preview of the students. If a score table for their department is uploaded by an OISEP official, they can perform auto-placement with one click on this page. They also can download the current score table for their department. From the navigation bar, they can see the notifications and make announcements.

3.6 Student

The students are allowed to enter the system if they are placed in a school. For the activation of their accounts, they need to sign up as an initial step. The students can view and manage their ERASMUS process. They have two main pages in the students’ view, namely “Dashboard” and “Forms and Requests”. In the dashboard, they can see the todo list where they can follow the tasks they should do. This todo list is updated automatically if there are any updates on their application process, such as approval of a form. They can also see the announcements made by deans, department chairs, course coordinators, instructors and exchange coordinators in the dashboard. In the forms and requests page. They can upload the pre-approval and CTE forms and view the status of their forms. They can also cancel the forms they sent.

3.7 OISEP

OISEP officials can enter the system after their accounts have been created by the admin. The OISEP accounts have one functionality, which is managing the score tables for every department. The only page that the OISEP officials encounter is the dashboard in which they can upload the score tables. This page also allows the OISEP officials to delete and download the score tables.

4 Build & Execution Instructions

In order to ease the deployment process, ERSMS uses Docker, which is a containerization platform that is used to deliver software and its all dependencies together in the form of containers. Thanks to Docker, ERSMS is platform-independent, which means one can run the ERSMS on GNU/Linux, MacOS, and Windows by using the same build and execution instructions.

For building and executing ERSMS:

1. You need to install the Docker software, if you do not have it installed on your system. For that, official documentation [here](#) can be useful.
2. Similarly, you need to clone the project’s repository from GitHub to your system using “git clone”.
3. Then, using your operating system’s shell (i.e., Bash on GNU/Linux, zsh on MacOS, or Powershell on Windows) navigate to the directory where you cloned the repository, for example, “cd ~/Downloads/ERSMS” on GNU/Linux or MacOS.
4. Then, once again, if you have done the previous two steps correctly, you need to see the “docker-compose.yml” and “Dockerfile” files in the root folder of the ERSMS repository.
5. Now, for running the project:
   a. If you are running the project for the first time, execute the following commands respectively in your operating system’s shell again.
      i. `docker compose -f docker-compose.yml build`
      ii. After the first command is completed, `docker compose -f docker-compose.yml up`
   b. If you are running the project again after the first time, you need to execute only
      i. `docker compose -f docker-compose.yml up`
6. Once your terminal indicates that containers are up and functional, you can use the ERSMS application from [http://localhost:8000](http://localhost:8000) or [https://localhost:8001](https://localhost:8001). (You can always use the HTTPS one, but since the HTTPS certificates are self-signed, your browser might complain about it.)
7. You can close the application by either terminating it from the terminal via CTRL+C Windows and GNU/Linux, or CMD+C on MacOS; or you can open up a separate terminal in the project repository’s root directory and run the “docker compose down” command.

As can be seen, it is possible to deploy the ERSMS application mostly automated in at most six main steps. Without Docker, it would be required to install .NET SDK, Postgresql Database Server, Angular Framework, and Nginx Reverse Proxy by hand and then deploy the project, which is tedious.

5 Work Allocations

Kutay Tire - 22001787

Analysis Report:
- Wrote the introduction part of the analysis report
- Drew and explained the use-cases in “Placement” package
- Drew and explained state machine diagrams of the forms
- Wrote the improvement summary for the second iteration
- Modified the class and state diagrams for the second iteration
- Wrote some parts of the non-functional requirements

Design Report:
- Modified the introduction part and the design goals of the project
- Drew the using entity objects and explained the relation between them
- Drew the “Controller Layer” by creating the class diagrams for controllers and their methods
- Explained the present methods and functionalities of the “Controller Layer”
- Wrote the improvement summary for the second iteration

Implementation:
- Worked on the front-end of the project
- Created the initial version of the charts at the dashboard page of the exchange coordinator
- Designed the initial version of the routing of the app
- Designed the “Placements” and “Logging” pages of the exchange coordinator specifically by placing the tables and implementing the logic behind.
- Used form and to-do list services to connect the backend of the chart and to-do list components
- Added the announcement panel for the dean, chair, student and instructor
- Made bug fixes for the front-end.

Final Report:
- Wrote the introduction part and the current status of the system
- Wrote the “Lessons Learned” part of the report
- Wrote the User’s Guide of the report
- Wrote work allocation for himself

Atak Talay Yücel - 21901636

Analysis Report:
- Wrote some parts in functional requirements
- Drew and explained the use-cases in “Management” package
- Drew the class diagram
- Wrote some parts in non-functional Requirements

Design Report:
- Drew the “User Interface Layer” using entity objects and explained the relation between them
- Made improvements on “Data Access Layer”
- Explained the relations between the entity objects

Implementation:
- Worked on the front-end of the project.
- Designed the Dashboard, Appointments, Placements, Messages, Login, SignUp, Profile Pages.
- Created the panels for viewing and creating Forms & Requests in the application and connected them to the backend using the services. Therefore, connected “Forms and Requests” and “Logging” pages to the backend using the services.
- Created the ToDo List and connected it to the backend using the services.
- Designed navigation bars, notifications, announcement.
- Designed the Admin Panel.
- Tested the application by trying various cases to find edge cases and bugs of the application.

Final Report:
- Wrote work allocation for himself
- Took screenshots of Swagger

Borga Haktan Bilen - 22002733

Analysis Report:
- Drew the auxiliary package for use case diagram
- Drew a part of authentication package for use case diagram
- Wrote explanations for authentication package and auxiliary package for use case diagram
- Drew and wrote explanations for activity diagrams
- Drew a part of class diagram and proof-read the diagram
- Modified the activity diagrams for the second iteration according to the feedback received
- Renamed the auxiliary package components of the use case diagram for the second iteration

Design Report:
- Drew subsystem decomposition and wrote explanation for the diagram
- Co-drew access control matrix for the section 2.5
- Drew final object diagram and wrote explanation for it
- Wrote external packages used in the application and packages created for the (within the) application.
- Drew repository layer diagram
- Wrote explanations for service and controller layer

Implementation:
- Created entity model structure for the backend
- Wrote repository classes for the entities
- Created service layer structure and wrote services
- Wrote REST API endpoints by creating controller layer
- Tested endpoints using Postman and Swagger, fixed bugs according to the tests
- Wrote service layer for frontend in order to connect backend to frontend
- Created models layer (representing DTOs) for frontend
- Bug fixing for the frontend

Final Report:
- Wrote work allocation for himself
- Took screenshots of different views of the application
Berk Çakar

Analysis Report:
- Wrote the current system section of the analysis report
- Wrote the proposed system section of the analysis report
- Wrote the functional requirement’s views subsection in the analysis report
- Wrote the non-functional requirements section of the analysis report
- Drew the Logging package for the use case diagram
- Wrote explanations for the Logging package (i.e., flow of events)
- Drew user interface mock-ups for the analysis report
- Improved the non-functional requirements section and the use case diagram based on the given feedback in the second iteration of the analysis report.

Design Report:
- Wrote the top two design goals subsection of the design report
- Drew the deployment diagram of the application, and explained the deployment procedure for the design report.
- Wrote the hardware/software software mapping subsection for the design report with performing a market research for picking a right cloud solution.
- Wrote the persistent data management subsection of the design report.
- Co-drew the access control matrix
- Wrote the boundary conditions subsection of the design report
- Wrote the design patterns subsection of the design report
- Wrote the glossary part of the design report

Implementation:
- Initialized the project structure (.NET backend + Angular frontend), and determined the required dependencies (i.e., external packages) for the implementation.
- Implemented the authentication system in the backend (including AuthenticationController, AuthenticationService, UserService, UserRepository, and TokenService classes).
- Initialized the actor related classes (including DomainUser, Admin, OISEP, CourseCoordinatorInstructor, DeanDepartmentChair, Student, and ExchangeCoordinator classes).
- Implemented the automated student placement based on the placement table uploaded by the OISEP staff (including AutoPlacer, PlacementController, PlacementService, PlacementRepository classes)
- Documented the backend source code using XMLDoc comments and generated a static documentation website via DocFX
- Overall refactoring and bug fixing of the backend code
- Implemented the routing and access guard mechanisms in the frontend
- Implemented the login and sign up screens with corresponding REST API calls
- Implemented Admin and OISEP dashboards by revising the previous design from the team members with corresponding REST API calls
- Fixed a critical bug in Student’s forms and requests panel
- Implemented a “toaster” service, so that success and error messages can be delivered to the end user graphically
- Containerized the project using Docker for automating the deployment process

Final Report:
- Wrote the build and execution instructions for the final report
- Wrote work allocation for himself
- Took screenshots of different views of the application

Yiğit Yalın
Analysis Report:
- Drew the Profile & Social part of the diagram and wrote their explanations.
- Drew the sequence diagrams and wrote their explanations.
- Wrote the non-functional requirements part.

Design Report:
- Wrote the object design trade-offs part.
- Modified the User Interface layer.

Implementation:
- Worked on the front-end of the project
- Implemented the appointments view.
- Implemented several dialogs for forms, file uploads and confirmations.
- Implemented the placements page for exchange coordinators and its connection to the backend.
- Implemented the models which are the frontend counterparts of the backend DTOs.
- Implemented some components of the dashboard, forms and requests and logging views.
- Implemented the student profiles.

Final Report:
- Wrote the User’s Guide part of the report.
- Wrote work allocation for himself
- Took screenshots of different views of the application
6 Appendix

A. Admin View

Admin Panel

User List

<table>
<thead>
<tr>
<th>First Name</th>
<th>Last Name</th>
<th>Username</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can</td>
<td>Alkan</td>
<td>22062902</td>
<td>Exchange Coordinator</td>
</tr>
<tr>
<td>Edward</td>
<td>Snowden</td>
<td>1</td>
<td>Admin</td>
</tr>
<tr>
<td>Slay</td>
<td>Tazin</td>
<td>22062961</td>
<td>Course Coordinator</td>
</tr>
<tr>
<td>Erik</td>
<td>Tozan</td>
<td>22062950</td>
<td>Office of International Students and Exchange Programs</td>
</tr>
<tr>
<td>Neli</td>
<td>Alar</td>
<td>22062950</td>
<td>Dean Department Chair</td>
</tr>
<tr>
<td>Salim</td>
<td>Alasy</td>
<td>22062904</td>
<td>Dean Department Chair</td>
</tr>
</tbody>
</table>

User Details

User Type:

First Name

Last Name

Email
berkcakar@ug.bilkent.edu.tr

Bilkent ID Number

******

Confirm Password

Submit
B. Login/Signup View

Email
berkcakar@ug.bilkent.edu.tr

Password
*******

Login

Don't have an account? Sign Up
Create Account

First Name

Last Name

Bilkent ID Number

Email

A valid Bilkent University email address is required

Password

Password is required

Confirm Password

Create Account

Already have an account? Login

C. OISEP View

Exchange Score Tables

Preview

<table>
<thead>
<tr>
<th>#</th>
<th>First Name</th>
<th>Last Name</th>
<th>Student ID Number</th>
<th>Faculty</th>
<th>Department</th>
<th>Degree</th>
<th>Transcript Grade(4/4)</th>
<th>Transcript Grade(10/100)</th>
<th>Transcript Grade Contribution</th>
<th>Total Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Kardelen</td>
<td>Ceren</td>
<td>22092700</td>
<td>Faculty of Engineering</td>
<td>Department of Computer Engineering</td>
<td>Lisans</td>
<td>3.98</td>
<td>99.530</td>
<td>49.333</td>
<td>99.333</td>
</tr>
<tr>
<td>2</td>
<td>Furkan</td>
<td>Guzelant</td>
<td>22092701</td>
<td>Faculty of Engineering</td>
<td>Department of Computer Engineering</td>
<td>Lisans</td>
<td>3.97</td>
<td>99.360</td>
<td>49.060</td>
<td>99.000</td>
</tr>
<tr>
<td>3</td>
<td>Safa Eren</td>
<td>Kuday</td>
<td>22092702</td>
<td>Faculty of Engineering</td>
<td>Department of Computer Engineering</td>
<td>Lisans</td>
<td>3.92</td>
<td>98.130</td>
<td>47.333</td>
<td>97.333</td>
</tr>
</tbody>
</table>
D. Exchange Coordinator View
E. Form Views

Pre-Approval Form
Submission Date 18/12/2022 11:11
Form Status: Waiting

Host Institution Courses

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT-X09-630</td>
<td>Social Constructs</td>
<td>5</td>
</tr>
</tbody>
</table>

Bilkent Requirement

Social Science Core Elective

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Bilkent Credits</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HUM312</td>
<td>Humanities</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Pre-Approval Form
Submission Date 18/12/2022 11:11
Form Status: Waiting

Student Details

Name: Borga Haktan Bilen
Student ID: 22002705
Department: Department of Computer Engineering
Email: haktanbilen@ug.bilkent.edu.tr
CGPA: 3.71
Exchange School: Saarland University
Preferred Term: 2022-2023 Spring Semester
Enterance Year: 2020

Approvals

Role: Exchange Coordinator  Role: Administration Board
Waiting  Waiting
### Submit Pre-Approval Form

**Course Group 1**

**Host Institution Courses**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT-X09-630</td>
<td>Social Constructs</td>
<td>5</td>
</tr>
</tbody>
</table>

**Bilkent Requirement**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Bilkent Credits</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

### Submit CTE Form

**Course Group 1**

**Host Institution Courses**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>ECTS</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT-AB2-321</td>
<td>Object Oriented Software Development</td>
<td>4</td>
<td>A-</td>
</tr>
</tbody>
</table>

**Bilkent Requirement**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Bilkent Credits</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS319</td>
<td>Object Oriented Software Development</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>
Equivalence Request

Submission Date: 18/12/2022 11:13
Form Status: Waiting

Student Details

Name: Borga HakanBilen
Student ID: 22002705
Department: Department of Computer Engineering
Email: hakanbilen@ug.bilkent.edu.tr
CGPA: 3.71
Exchange School: Saarland University
Preferred Term: 2022-2023 Spring Semester
Entry Year: 2020

Approvals
Role: Exchange Coordinator or Course Coordinator
Waiting

Submit Equivalence Request

Host Institution Course

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT-B98-786</td>
<td>Psychology of Innovation and Entrepreneurship</td>
<td>5</td>
</tr>
</tbody>
</table>

Upload Syllabus
Psychology of Innovation and Entrepreneurship - AGH - SYLLABUS.pdf

Bilkent University Course

Course Type: General Elective

Course Code and Course Name are Optional

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Bilkent Credits</th>
<th>ECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSYC115</td>
<td>Psychology of Innovation</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Additional Notes
Write your Additional Notes here...

Cancel  Submit
Your Decision

Thank you for your determination.

F. Student View
Apply to the host university before the application deadline.
G. Dean View

H. Chair View

I. Swagger Web View