Bilkent University

Department of Computer Engineering

CS319 Term Project

Project short-name: CUMREP

Final Report

Ömer Fırat Bekiroğlu - 22002239

Sabri Eren Dağdelen - 22001764

Anıl Altuncu - 21901880

Yaşar Tatlıcıoğlu - 22003856

Emirhan Ay - 22203902

Yamaç Yiğit Ozan - 22003595

Instructor: Eray Tüzün
Teaching Assistant(s): Yahya Elnouby, Mohammad Umair Ahmed

Progress/Final Report
May 29, 2023

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Object Oriented Programming Course CS319.

Contents

Summer Training Report Evaluation System

1. Introduction

We have successfully accomplished our main objectives in this project and developed a proper user management framework. We focused on creating a smooth user experience by implementing key functionalities such as user sign up, sign in, and log out. Users can quickly sign up for an account, provide their necessary information, and securely sign in and log out of the system.

Additionally, we have achieved another significant objective by implementing the document upload feature, which allows users to upload documents related to their internships. The system securely stores these documents. Moreover, our project includes document viewing functionality. Users can access and view their uploaded documents, making it easy for them to review and track their internship progress. The system provides a user-friendly interface allowing smooth navigation and easy access to the uploaded documents.

We have also incorporated a feedback mechanism within the system to enhance the current feedback process. This enables the evaluators or teaching assistants, to provide valuable feedback for the uploaded documents.

However, as we progressed through the development process, we identified the need to make some minor adjustments and changes to ensure that our objectives are satisfied.

2. Lesson Learnt

During the implementation process, we faced several challenges as we utilized tools and technologies that were unfamiliar to us. Learning and developing with limited experience in these areas posed a significant hurdle. While most of us were familiar with C-like languages, some of our team members needed more experience with JavaScript, which was the primary language for this project. In addition to JavaScript, we also worked with Node.js, Google Firebase, and React.js.

To tackle these challenges, we divided our group into Frontend and Backend teams. This division allowed us to focus on specific aspects of the project. The Frontend team primarily worked with React.js, a popular JavaScript library for building user interfaces. This involved understanding React's component-based architecture, managing state and props, and handling UI rendering efficiently.

On the other hand, the Backend team delved into Node.js, a JavaScript runtime environment that enabled server-side development. Additionally, they worked with Google Firebase, a cloud-based platform that provided various services like authentication, real-time database, and hosting. This required understanding Firebase's SDK and using it effectively to handle user authentication and interact with the database.

While the learning curve was steep, we overcame these challenges by dedicating time to self-study from online resources. We also encouraged knowledge sharing within the team, allowing members with prior JavaScript experience to mentor and support those new to the language. Despite the initial difficulties, this project served as a valuable learning experience for all of us. We gained a solid understanding of JavaScript, Node.js, React.js, and Google Firebase, expanding our skill sets and broadening our development capabilities.

3. User's Guide

Following figures and descriptions illustrate how to use CUMREP

3.1 Auth

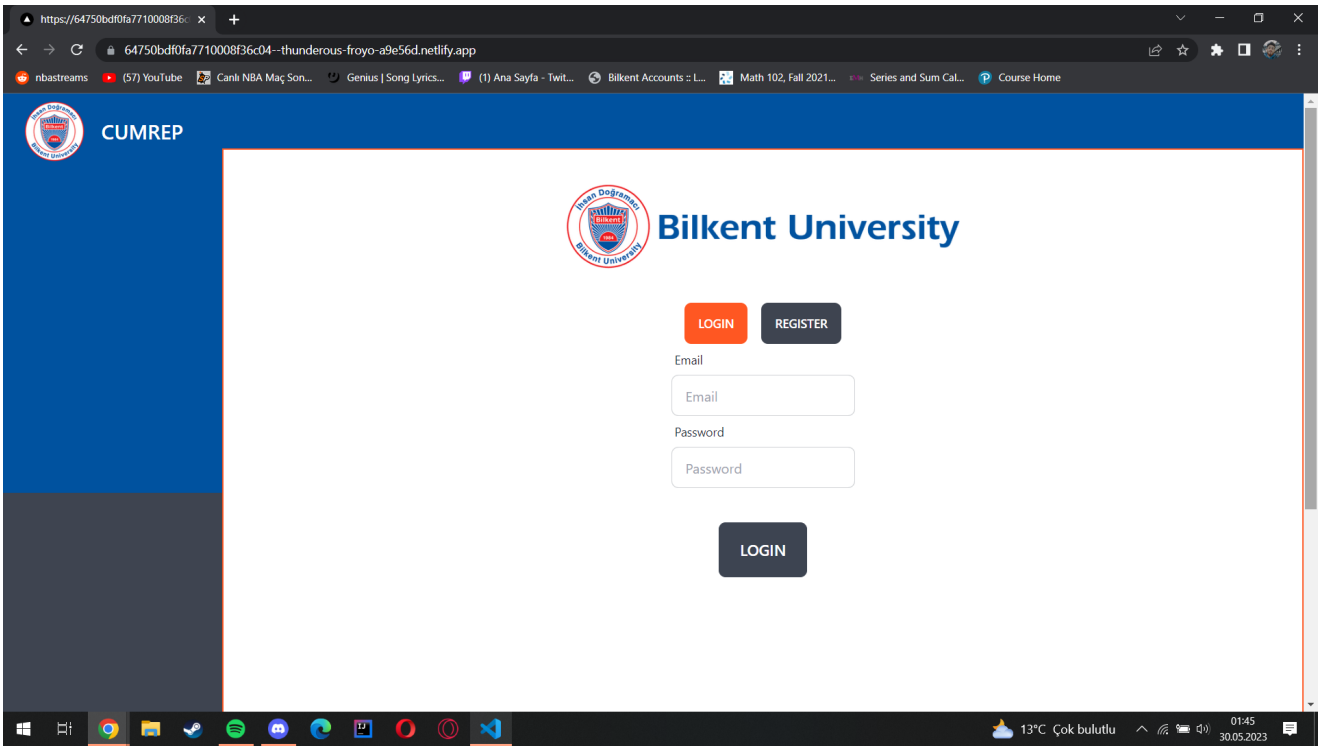

fig. 1 Auth Screen / Log In

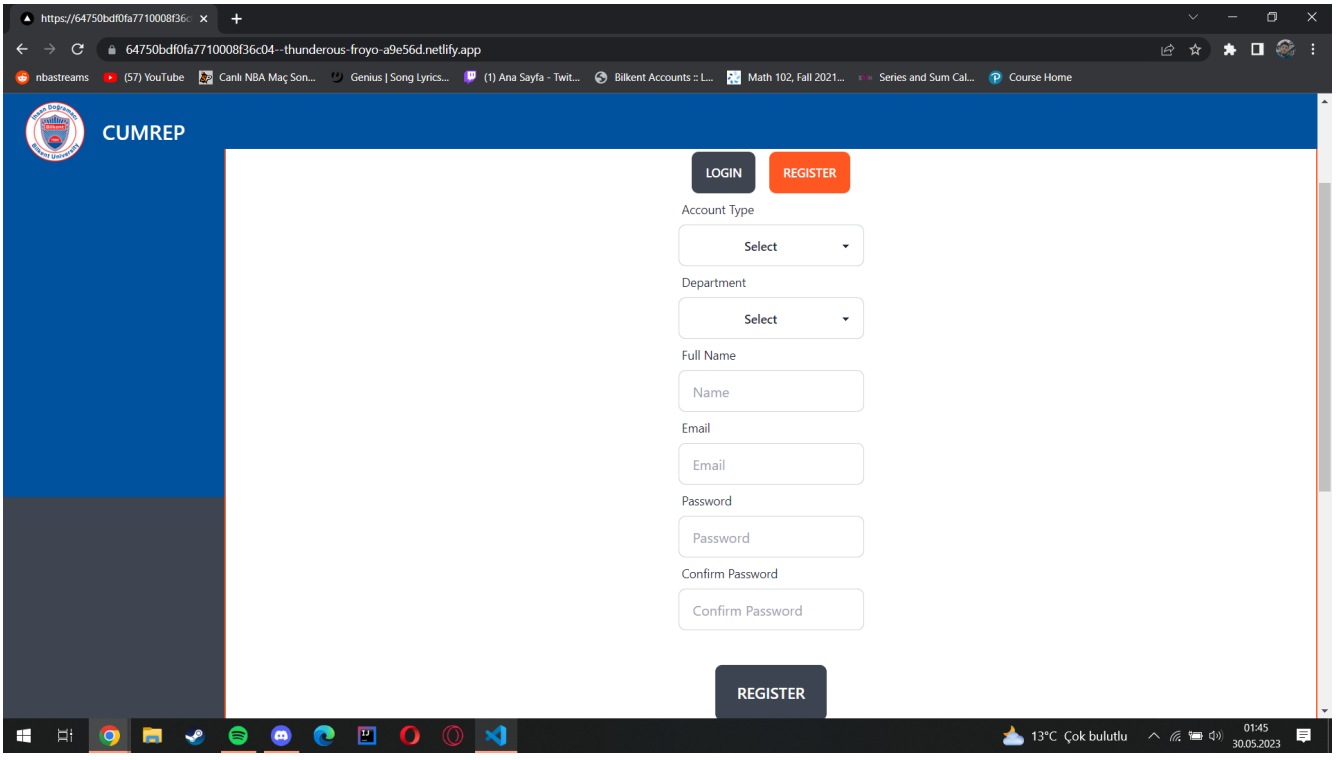

fig. 2 Auth Screen / Register

In the Auth Screen you can register to CUMREP with desired information about you. If you already have a profile, you can log in using the profile's email and password.

3.2 Profile

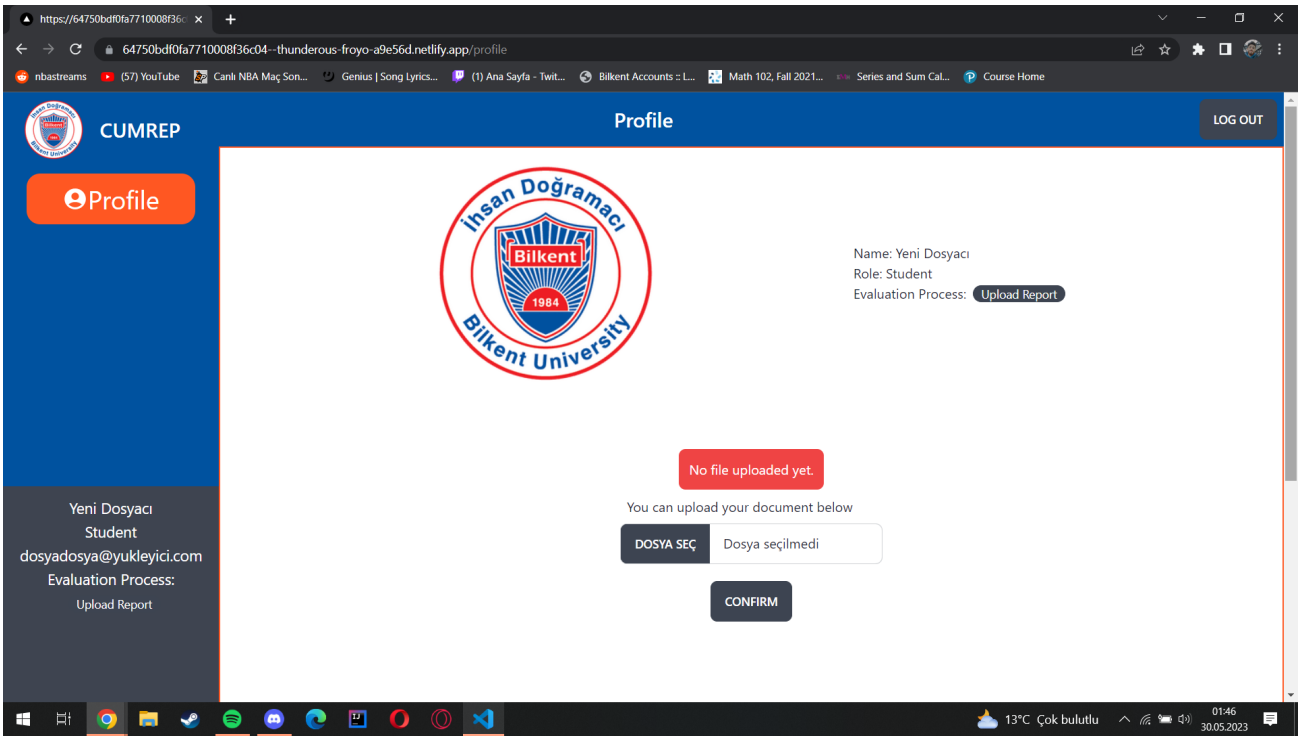

fig. 3 Profile Screen

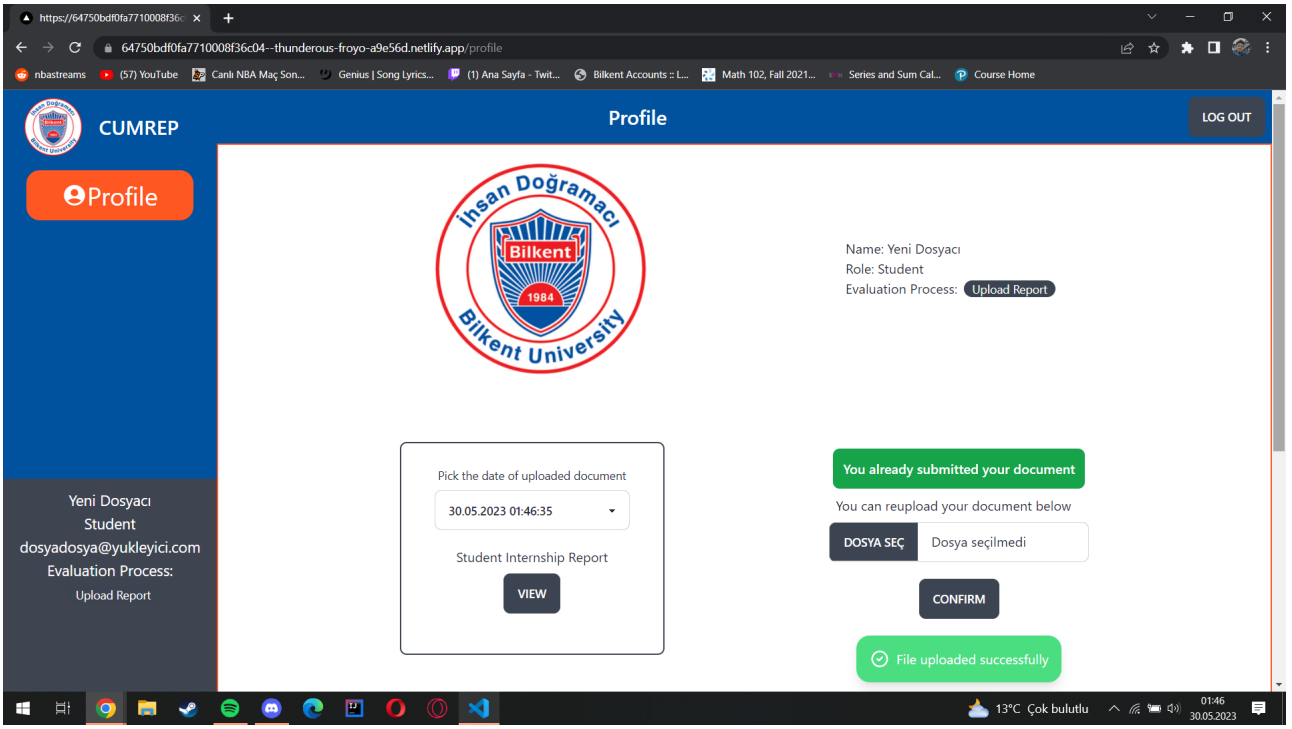

fig.4 Profile Screen After Uploading Document

In the Profile Screen, the Profile information can be seen. Additionally, Students can upload their documents and view the uploaded document.

3.3 Documents

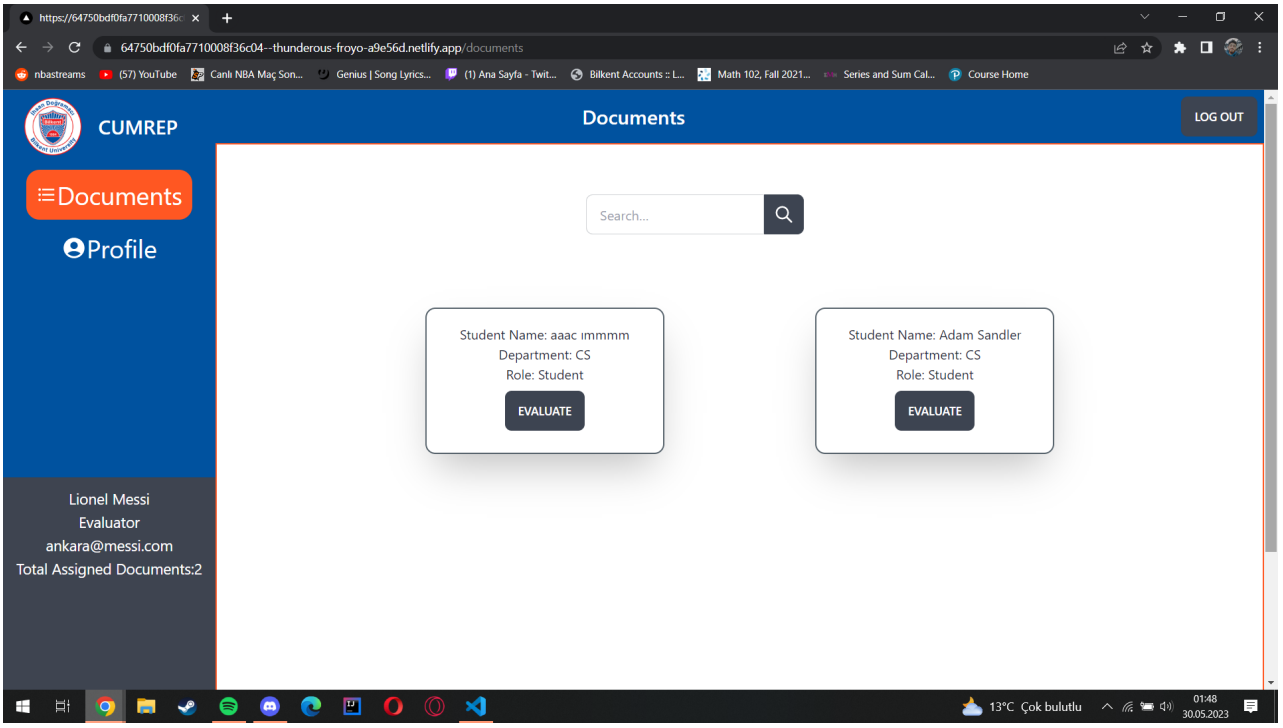

fig. 4 Documents Screen

In the Documents Screen, the Department Secretary approves the Teaching Assistant's and Evaluators' accounts in order to prevent the students from manipulating CUMREP. Also, the Department Secretary can assign documents to Evaluators, and Teaching Assistants and Evaluators can view the documents that are assigned to them.

3.4 Evaluation

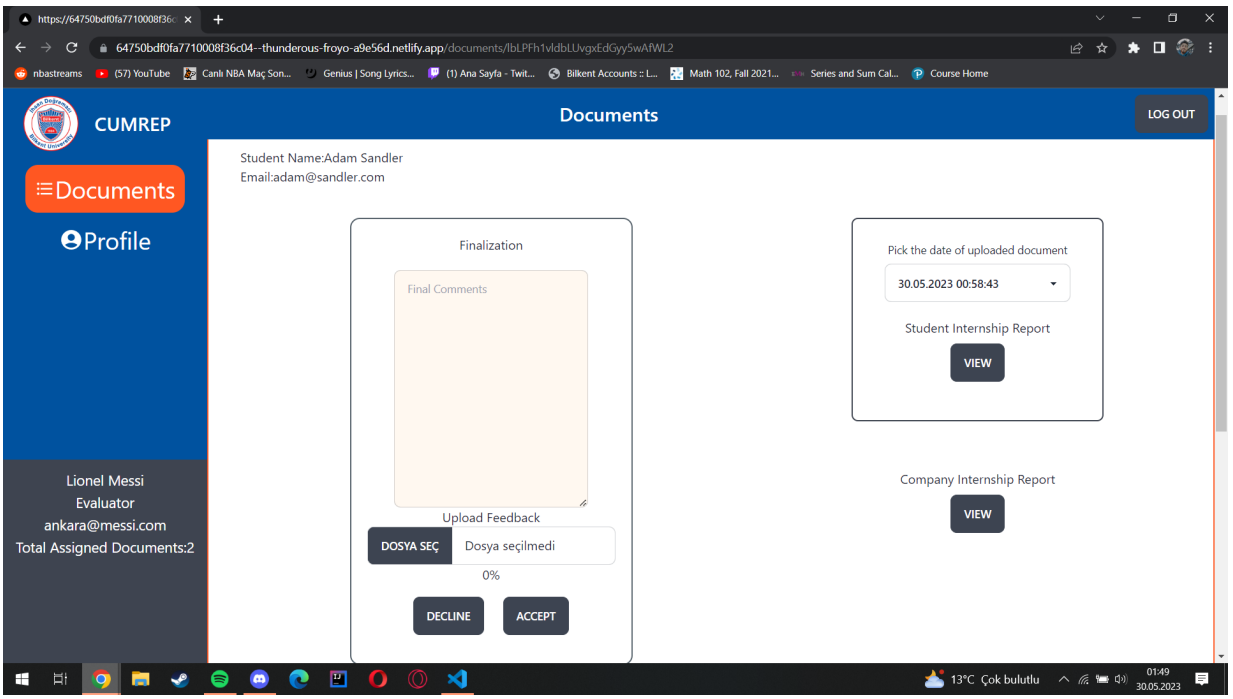

fig. 5 Evaluation Screen

In the Evaluation Screen, Teaching Assistants and Evaluators can grade the document, also they should write a feedback in order to inform the students which uploaded the graded document. Lastly, Teaching Assistants and Evaluators must decide whether the current document is approved or declined in this screen.

4. Build Instructions

4.1 Build and Run

- There are many options to select as an IDE, but we used Visual Studio Code for our development. Therefore, you can prefer our choice. You can download it from here:
  https://code.visualstudio.com/download
- To install node.js you can use this link:
  https://nodejs.org/en/download
- Open the source code from Visual Studio Code.
- Open a new terminal and write “cd Frontend”, to modify the path.

- Run “npm install” on the terminal to install npm. You can also check the versions of both node.js and npm by running codes below:
  -“npm --version”
  -“node --version”
  Application works on:
  https://64750bdf0fa7710008f36c04--thunderous-froyo-a9e56d.netlify.app

5. Work Allocation

Ömer Fırat Bekiroğlu

Analysis Report:

- System Models: Use Case Diagram, Use Case Textual Descriptions, Object & Class Model

Requirements Report:

- High Level Software Architecture: Subsystem Decomposition Diagram, Hardware/Software Mapping, Deployment Diagram, Persistent Data Management, Access Control And Security, Access Control Matrix, Boundaries
- Layers: Data Management Diagram

Implementation: Firebase Authentication Setup, Firebase Collections Setup, Auto-Assign of Documents to TA’s

Final Report: Introduction, Descriptions of the User’s Guide, Own part at the Work Allocation

Anıl Altuncu

Analysis Report:

- System Models: Use Case Diagram, Use Case Textual Descriptions, Object & Class Model

Requirements Report:

- Layers: Data Management Diagram
- User Interface Layer Class Interfaces descriptions

Implementation: Firebase Authentication Setup, Firebase Collections Setup, Firebase Documents Storage

Final Report: Lesson Learnt, Build Instructions, Own part of the Work Allocation

Sabri Eren Dağdelen

Analysis Report: User Interface

Requirements Report: Object Design Tradeoffs, User Interface Class Interfaces, User Interface Management Layer

Implementation: Full Stack Developer. Frontend: All pages and components. Backend: Collection and document management, file uploads

Final Report: Work Allocation

Yaşar Tatlıcıoğlu

Analysis Report: Proposed System, Actors, Non-functional Requirements, Functional Requirements, Pseudo Requirements

Requirements Report: Subsystem Decomposition, Web Server Layer Class Interfaces

Implementation: Dynamic Modifications to Profile page, contribution to Upload page for TA's, contribution to firebase configurations for login/signup page, some storage modifications for feedback file storage, interface for manual assign page.

Final Report: Demo Video, Work Allocations

Emirhan Ay

Analysis Report:

Requirements Report:

Spell Check System:

- Ability to detect and suggest corrections for spelling errors in text documents.
- Support for the English language.
- Highlighting miss spelled words .
- Efficient and accurate detection of spelling errors.

Turnitin System:

- Ability to analyze documents and detect instances of plagiarism.
- Support for file format (PDF).
- Similarity detection and highlighting of plagiarized content.
- Output generation with detailed plagiarism analysis.

Implementation: Python, spellcheck: PyMuPDF, fitz,enchant, turnitin: re, requests, json

Final Report: Case/state machine/sequence diagram, own part of the work allocation

Yamaç Yiğit Ozan

Analysis Report: Introduction, Current System

Requirements Report: Purpose of the system, Design Goals, Final Object Design, Design Patterns, Glossary, References

Implementation:

Final Report: Own part of the Work Allocation