Configuration Management Plan
CNG 492

Summer Internship Software

Mert Damburaci – 2453108

Umutcan Celik – 2526200

Ali Fırat Özdemir – 2453447

Efekan Uysal – 2585460

Table of Contents

Source Code

The three main components of our application are the front-end, back-end, and database. The front-end is built with Angular, a contemporary and potent framework that allows us to create dynamic and responsive user interfaces, allowing us to create a smooth and interactive experience for our users. On the back-end, we use Java, a strong and scalable programming language, to handle server-side logic, business rules, and API integrations. PostgreSQL is our relational database system, which offers effective data storage and retrieval capabilities. Additionally, we use pgAdmin4 for database administration, which makes it easy to manage schemas, run queries, and keep an eye on performance. To ensure smooth collaboration and version control, we maintain separate repositories on GitHub for the front-end and back-end. Team members working on the front-end commit their changes to the Angular repository, while those working on the back-end commit to the Java repository. This separation allows for clear ownership and streamlined development processes. Each team member works on their local machine, creating commits and pushing changes to GitHub to keep the codebase up to date. Other team members regularly pull these updates to ensure their local environments are synchronized with the latest changes. In cases where we need to revert to an earlier version of the code, Git's powerful version control features, such as reverting commits or checking out previous states, allow us to manage the code history effectively. This well-organized structure ensures that our project is maintainable, scalable, and collaborative. By leveraging Angular for the front-end, Java for the back-end, and PostgreSQL for the database, we have built a robust and efficient system that meets our project requirements. The use of GitHub for version control further enhances our ability to work as a team, ensuring that everyone is aligned and up to date with the latest developments. You can find related figures in Figures 1, 2, 3, 4, 5 and 6.

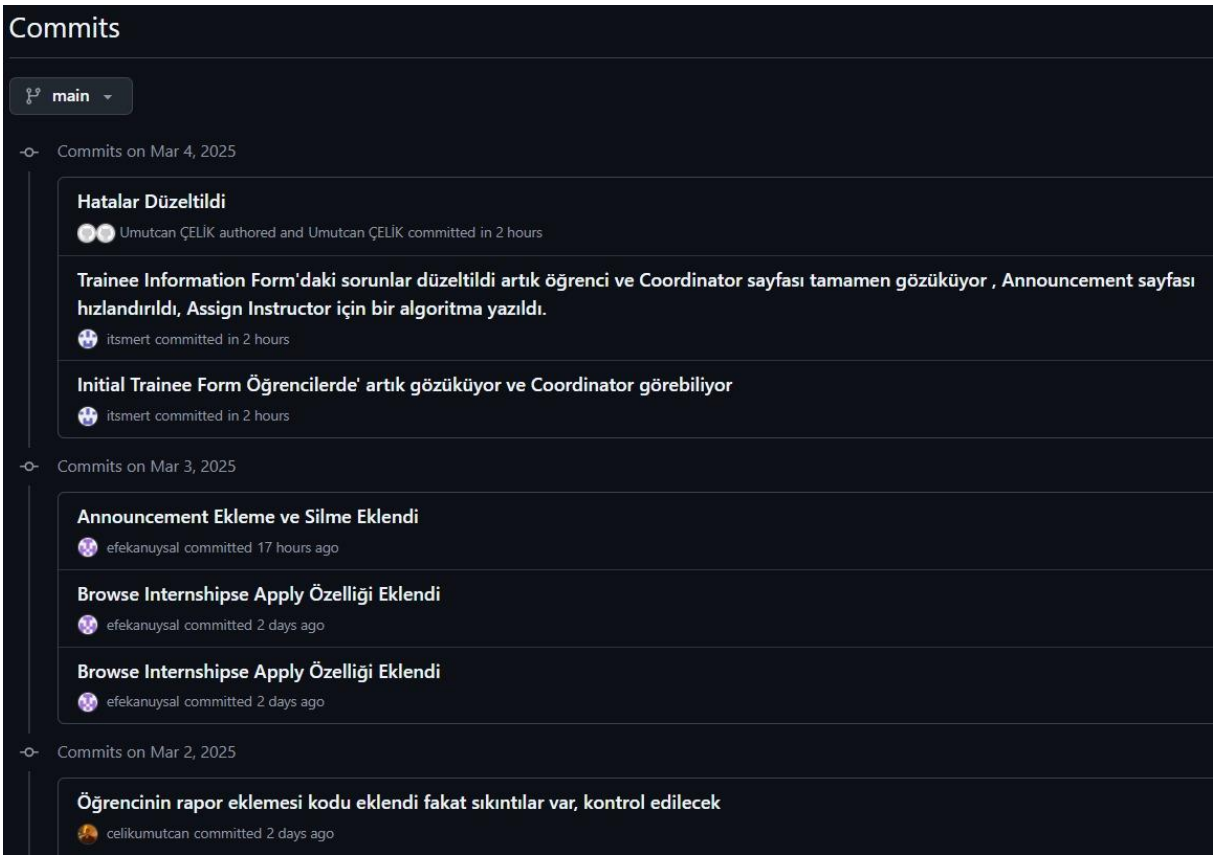

Figure 1: Internship Commits Logs

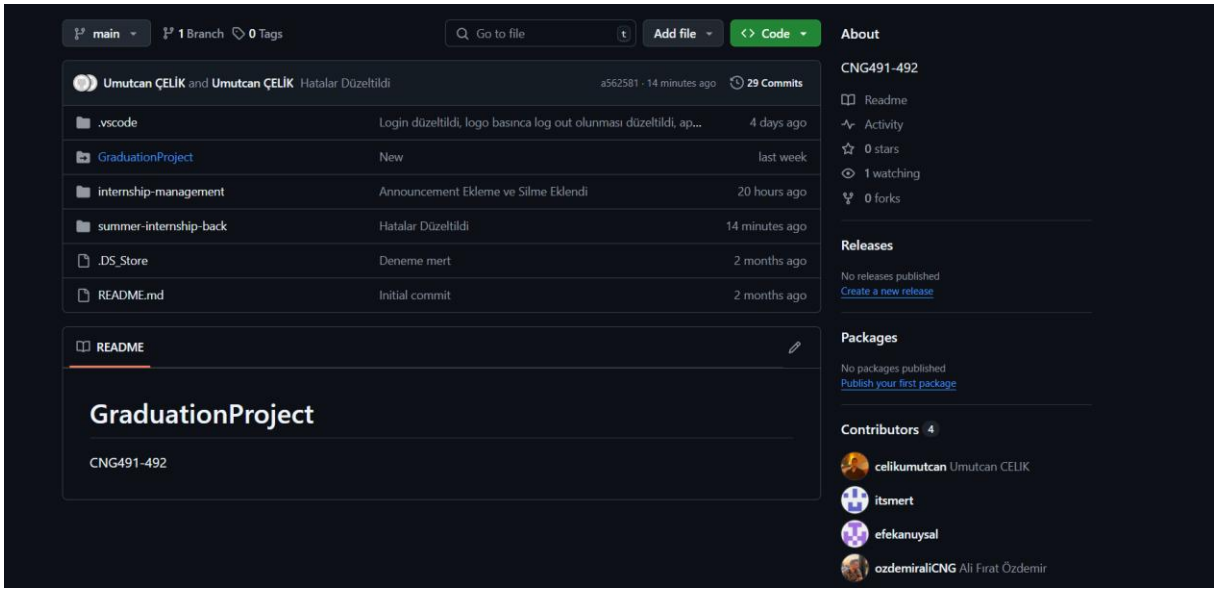

Figure 2: Source Codes

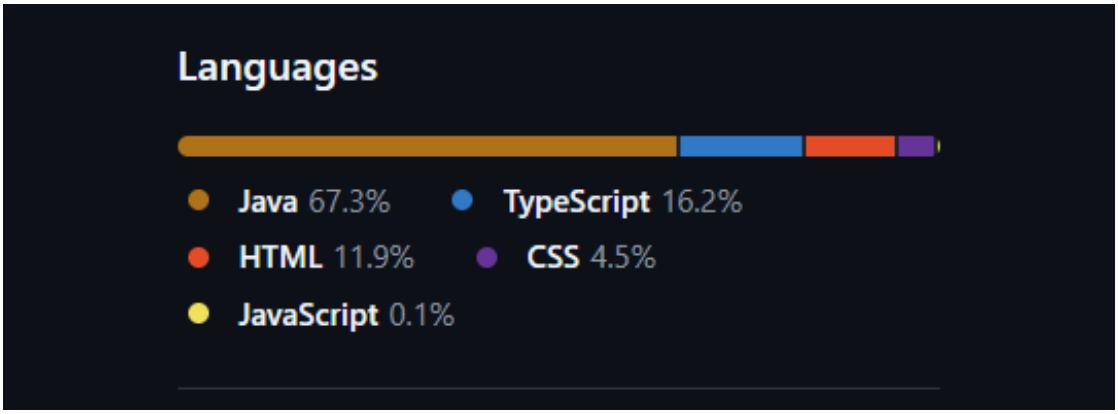

Figure 3: Languages Used

Frontend - Github:

https://github.com/celikumutcan/GraduationProject/tree/main/internship-management

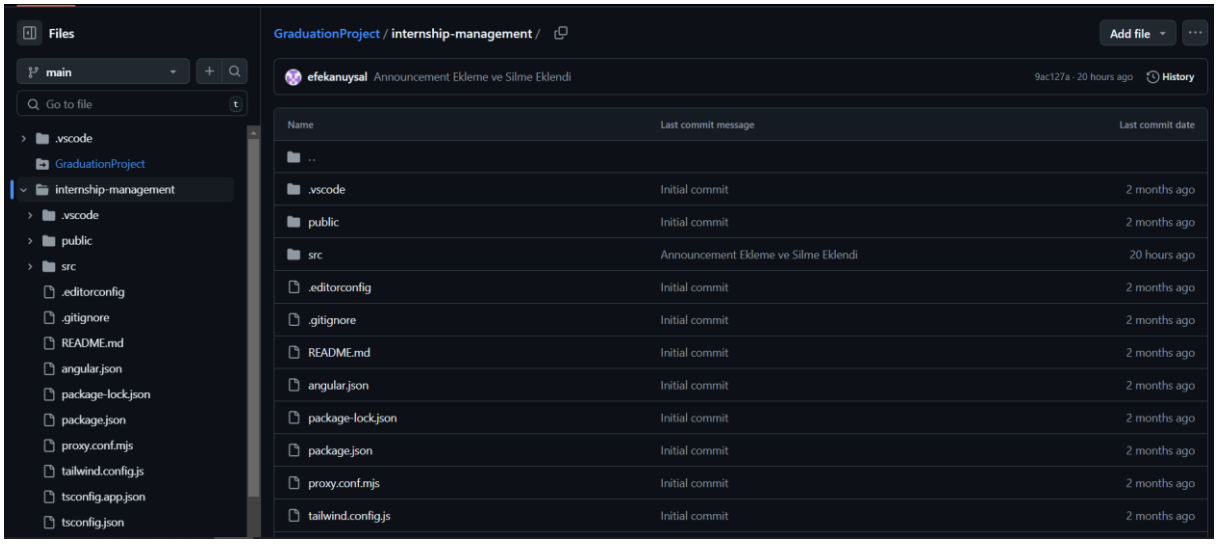

Figure 4: Frontend Repository

Backend - Github:

https://github.com/celikumutcan/GraduationProject/tree/main/summer-internship-back

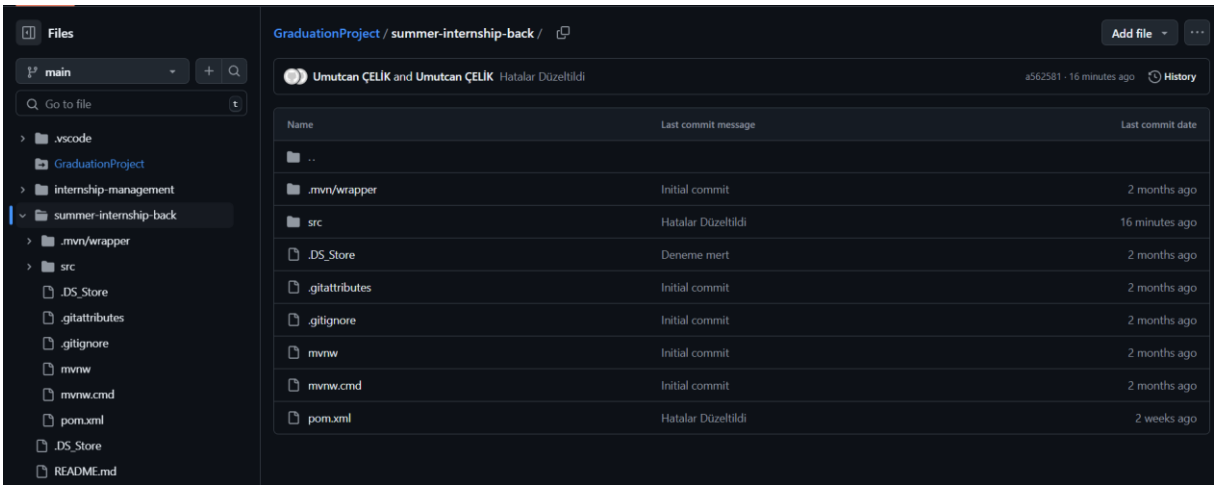

Figure 5: Backend Repository

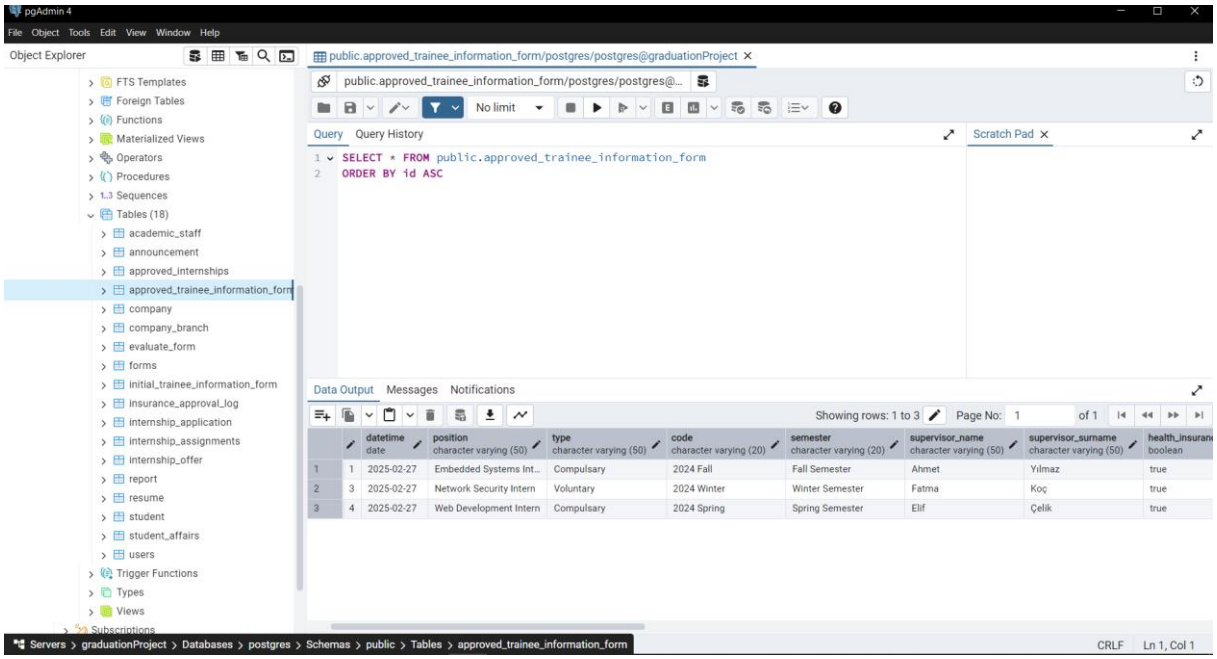

Figure 6: Database (PgAdmin4 & PostgreSql)

Tests

Testing is a critical process in our development lifecycle, ensuring that our application consistently meets high-quality standards and delivers a seamless user experience. In our project, which features an Angular-based frontend integrated with a Java Spring Boot backend and a PostgreSQL database, we are currently prioritizing comprehensive manual testing. Although we intend to adopt automated unit and integration tests in future development phases, our present focus is on methodically verifying each component through hands-on evaluation. At this stage, our manual testing efforts are concentrated on several key areas. Firstly, we meticulously test interactive elements such as buttons and links to confirm that they trigger the intended actions when clicked. Each component’s behaviour is closely observed to ensure that user interactions result in the correct responses and navigations. This approach allows us to catch any discrepancies in functionality early in the development process.

Another crucial aspect of our manual testing is the verification of the user interface's appearance and behaviour under different visual modes. We carefully check that both dark mode and light mode are applied consistently across all pages, ensuring that every element from text colours to background contrast adheres to our design standards. This detailed inspection is essential to provide users with a reliable and visually appealing experience regardless of their mode preference.

Additionally, in the "My Resume" section, we focus on validating the file upload functionality. Our tests specifically confirm that only PDF files are accepted, thereby preventing the upload of unsupported file types. By attempting to upload files in various formats, we ensure that our system reliably enforces this restriction, maintaining the integrity and expected operation of the feature.

These thorough manual tests are documented and reviewed regularly, serving as a foundation for future automated testing strategies. As we continue to evolve the project, our plan is to gradually incorporate automated unit and integration tests. When implemented, these tests will not only streamline the validation process but also provide a continuous feedback mechanism through our CI pipeline, with all changes meticulously tracked via GitHub.

In summary, our current manual testing strategy encompassing interactive functionality checks, visual mode consistency, and rigorous file upload validation, plays a crucial role in guaranteeing the overall quality of our application. This detailed approach ensures that we catch potential issues early, enabling us to enhance our system's performance and reliability while paving the way for the eventual integration of automated testing tools.

Reporting/Notification Scripts

In our project development process, we utilize multiple platforms as a team. It's crucial for each member to stay updated on any changes related to communication, project planning, and version control tools. Luckily, the platforms we use come with built-in notification systems, making it much easier to stay informed. The platforms we rely on are as follows:

- Whatsapp – Communication
- Microsoft Teams – Communication & Version Control
- Github – Version Control

Whatsapp

We use WhatsApp for informal communication, such as coordinating meeting dates. Like Discord, WhatsApp also sends notifications directly to the user's device. You can check from figure 7.

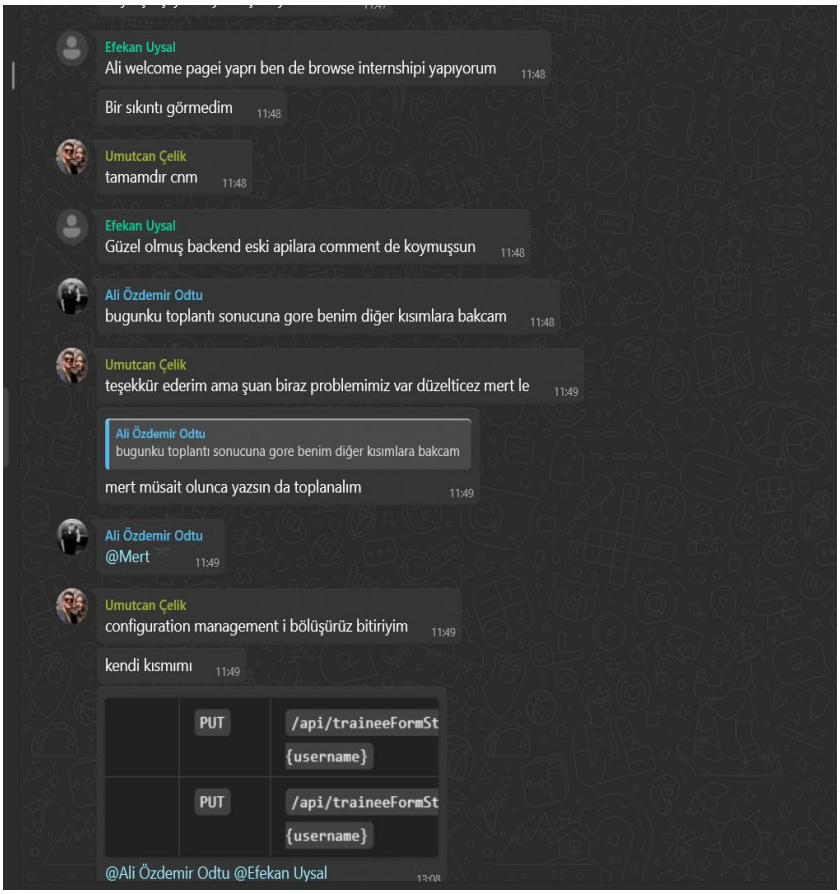

Figure 7: Whatsapp Communication

Microsoft Teams

We use Microsoft Teams for scheduling meeting times and planning each other's tasks. Similar to WhatsApp. Teams also provides notifications to keep users informed. You can control from figure 8.

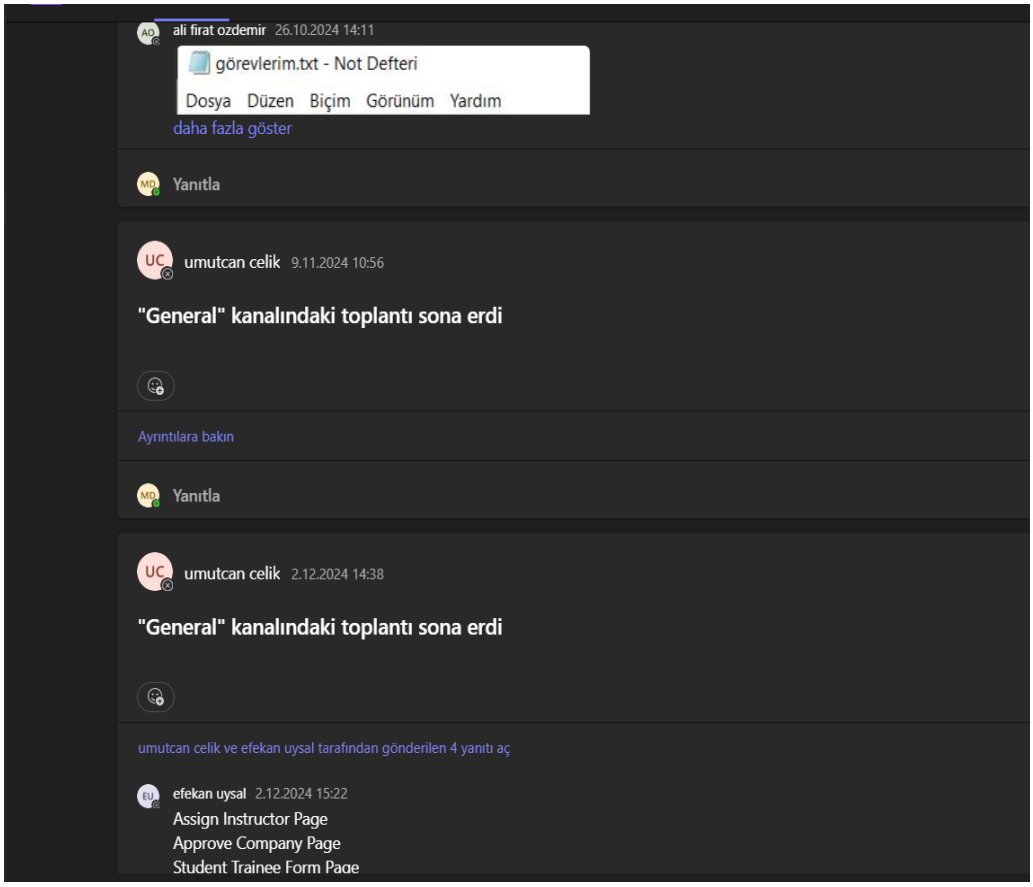

Figure 8: Teams Communication

Github

As we use GitHub for version control, staying informed about any changes to the source code is essential. With GitHub's "watch" feature, team members receive notifications about pull requests, releases, discussions, security alerts, and issues related to the repositories. You can check from figure 9.

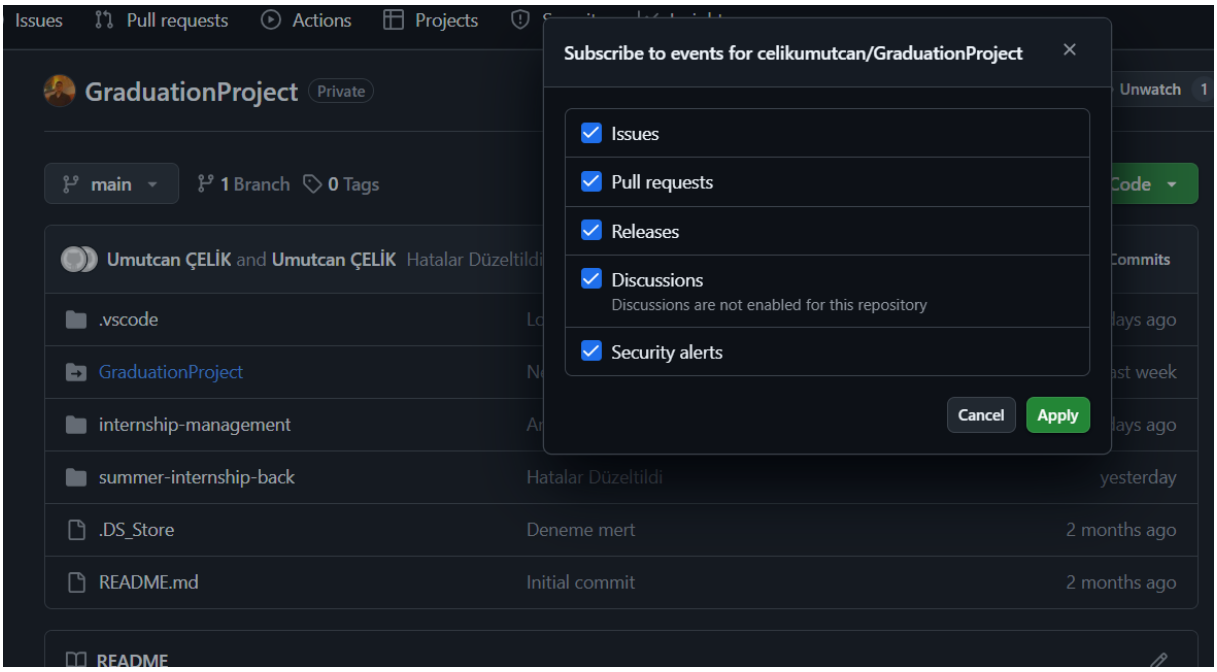

Figure 9: Github Notifications

Document Files

Our project team creates several documents during development. These documents are:

- Proposal
- Software Requirements Specification Report (SRS)
- Software Design Description (SDD)
- Configuration Management Plan
- Test Plan
- Test Results
- Final Report

We store these documents in both Google Drive and Microsoft Teams. This helps us keep all files organized and updated for all team members. Keeping the documents in two places is useful because if there is a problem with one, we can still access them from the other. You can find these in figures 10, 11,12 and 13.

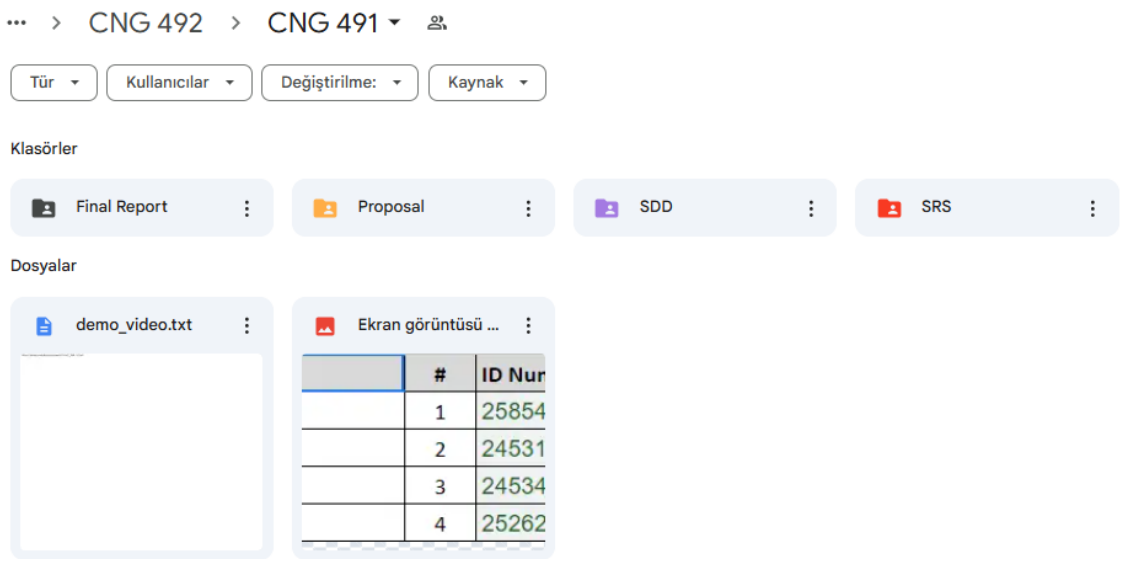

Figure 10: Google Drive Folder CNG 491

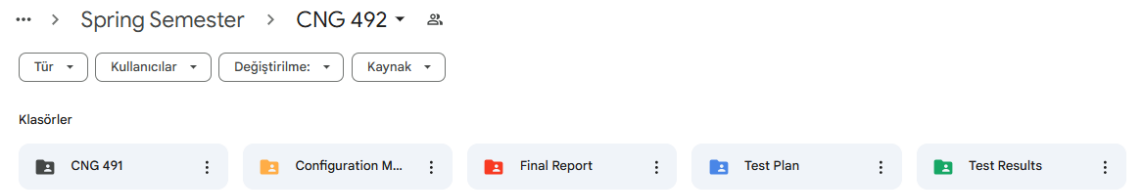

Figure 11: Google Drive Folder CNG 492

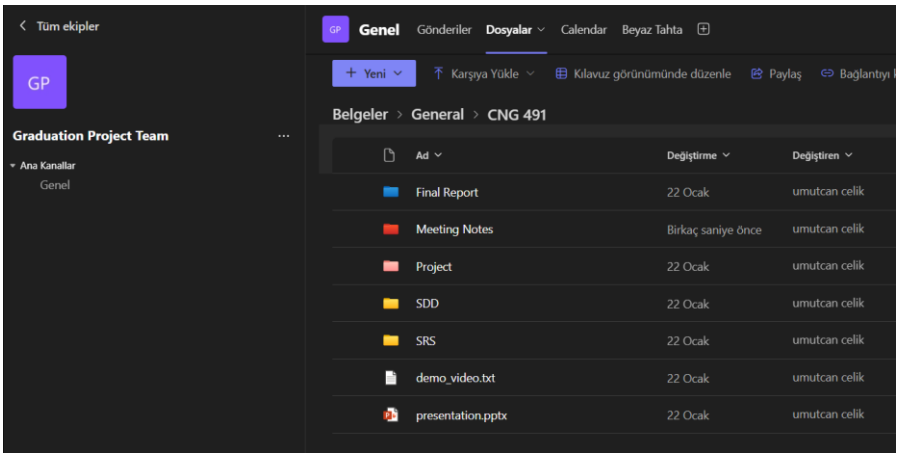

Figure 12: Microsoft Teams Folder CNG 491

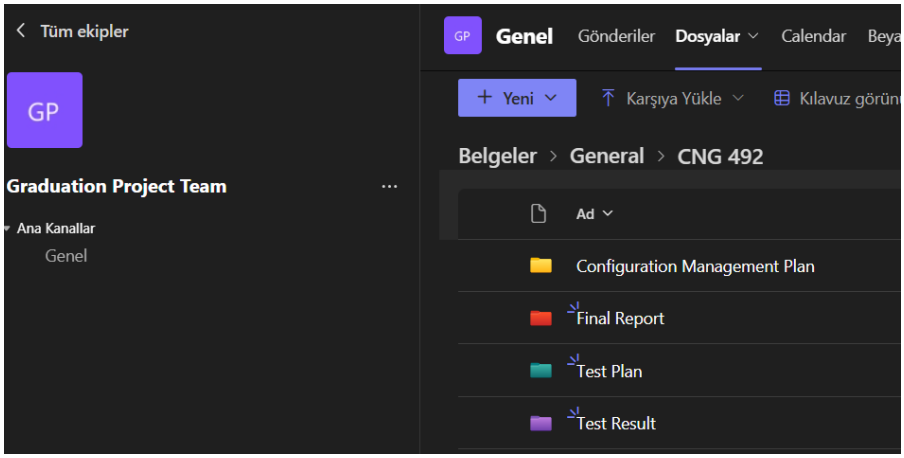

Figure 13: Microsoft Teams Folder CNG 492

In Google Drive and Microsoft Teams, we upload our reports and documents with the updated version so that everyone can work on their parts at the same time. After each update to a document, we save it in the relevant folder. We check the "Modified Date" column to find the latest version. You can see this organization in Figure 14.

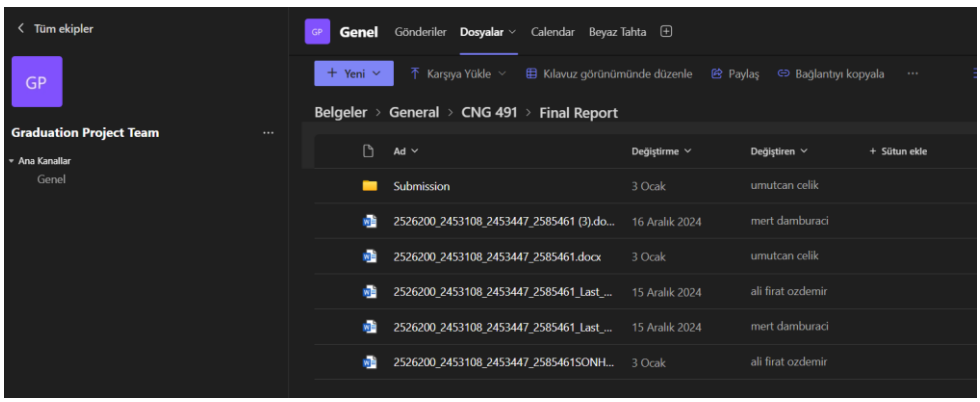

Figure 14: Organization of final report

Build Scripts

During the development of the Summer Internship Project, we have used Spring Boot with Maven for the backend and Angular for the frontend. Our database is hosted on Google Cloud using PostgreSQL. We have consistently used the following commands in the terminal for development:

- **mvn clean install** == This installs all necessary dependencies and builds the backend application.
- **mvn spring-boot:run** == This starts the Spring Boot backend server.
- **npm install** == This checks and installs the required dependencies for the frontend.
- **ng serve** == This runs the Angular frontend on a local development server.
- **mvn package** == This packages the backend application for deployment.

Other than that, we primarily use Maven for managing dependencies in the backend and NPM for handling frontend dependencies.

Version Tree

We utilized GitHub for version control, following industry standards. Our project is managed within a private repository, using a single main branch to maintain simplicity and efficiency for our four-person team. So far, our version tree includes over 25 commits, as shown below:

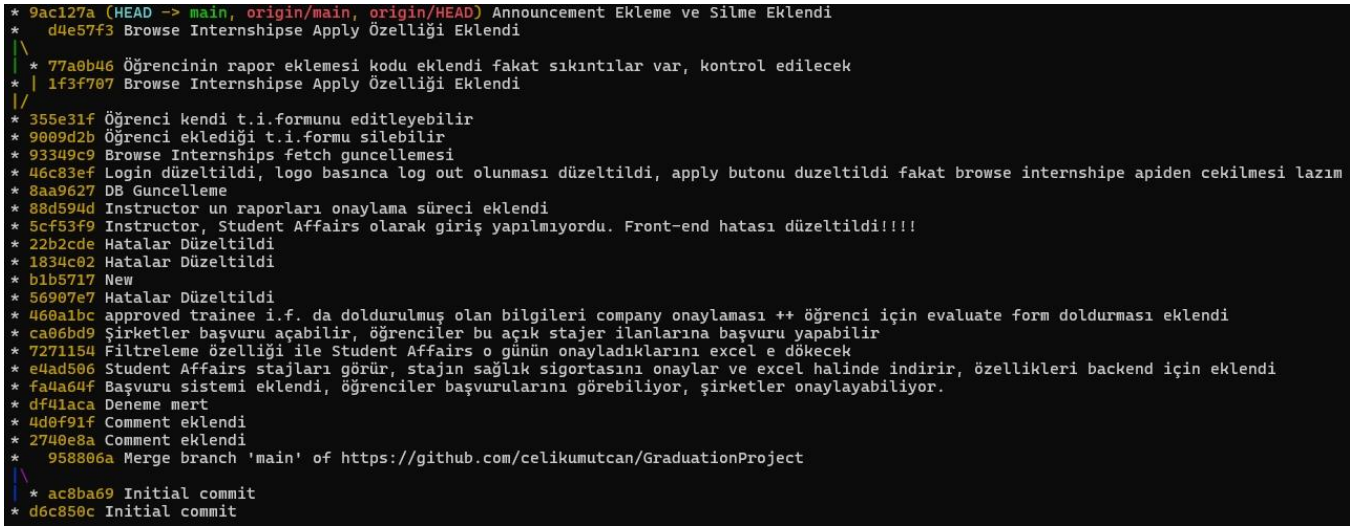

Figure 15: Version Tree

Log/History

You can find our Git log below. Each time a team member makes a change, they commit their updates to the repository. This structured approach allows us to easily track modifications, understand what has changed, and review progress without needing to ask the contributor directly. You can see this organization in Figure 16.

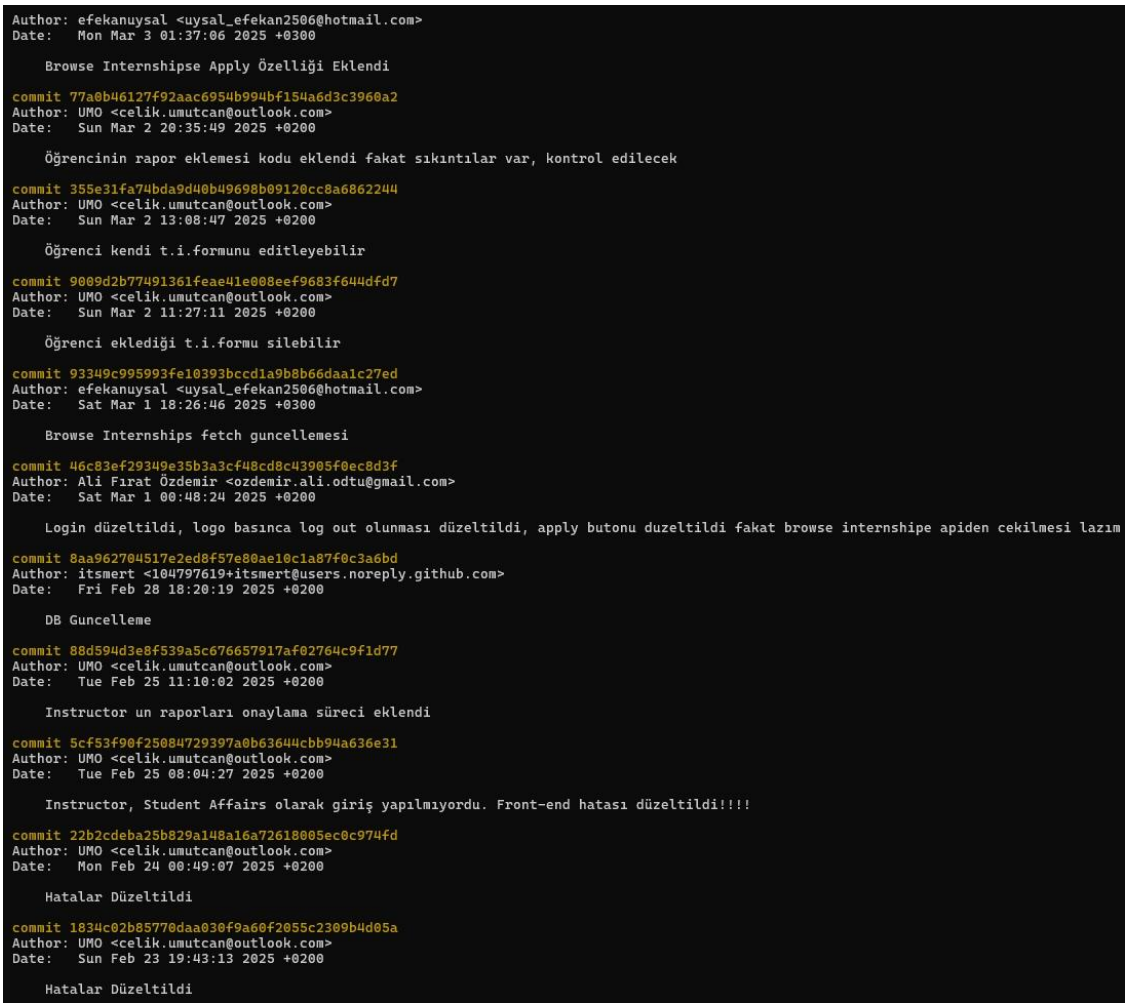

Figure 16: Git Logs

As we define our project goals, we hold regular meetings in Microsoft Teams to discuss important issues and ensure alignment. During these meetings, tasks are assigned to team members. This allows everyone to have a clear idea of their responsibilities and stay informed about the project's progress. Furthermore, to ensure steady progress and accountability, our team is divided into two groups: the backend team, consisting of Mert and Umutcan, and the frontend team, consisting of Efekan and Ali. Each team member is responsible for their assigned tasks and the assigned tasks of their team member. If a team member is unable to complete their task or falls behind, the other member of the team steps in to review the work and ensure continuity in productivity.