Doy!

Project Plan

# Introduction

This document contains information about roles of the team members and details about the development process of the project such as the performance and task size measurements, deployment strategy and iteration phases. A Gantt chart is provided as an appendix for a visual view of the development process.

# Project organization

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Team Member** | **Software Developer** | **Software Project Manager** | **Software Architect** | **Software Analyst** | **Software Configuration Manager** | **Software Tester** |
| Abdussamet Tekin | X |  | X |  |  |  |
| Barış Yıldız | X | X |  |  |  |  |
| Said Çetin | X |  |  |  |  | X |
| Muzaffer Berke Savaş | X |  |  | X |  |  |
| Mehmet Oğuz Kocadere | X |  |  |  | X |  |

# Development process and measurements

We will follow the agile methodology. The development process is divided into 2-week sprints. Progress will be tracked using sprint backlogs, burndown charts and weekly meetings. Performance will be measured using story points, velocity and sprint completion rates. After each sprint, a sprint review and sprint retrospective will be conducted to evaluate progress and improve the development process.

# Project milestones and objectives

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase** | **Iteration** | **Primary objectives** (risks and use case scenarios) | **Scheduled start or milestone** | **Target velocity** |
| Vision and Project Planning | I1 | Objectives   1. Understanding Customer Needs and Interviews 2. Preparing Vision Document 3. Mitigating Risk of Plan Change and Underestimation of Required Time to Develop 4. Preparing Project Plan Document | 1.03.2025 / 8.03.2025 | 4 |
| Software Requirements | I2 | Objectives   1. Mitigate Risk of Requirement Change 2. Preparing Software Requirements Document 3. Defining Use Cases 4. Defining Test Cases 5. Designing Graphical User Interface | 8.03.2025 / 22.03.2025 | 10 |
| Implementation | I3 | Objectives  1. Mitigate Risk of Potential Defect of Reusable Software Components  2. Implementing 3 Use Cases (Finishing %25 of the Project)  3. Preparing Configuration Management Report  4. Preparing Architectural Notebook  5. Preparing Risk Management Document | 22.03.2025 / 5.04.2025 | 21 |
| Design | I4 | Objectives  1. Mitigate Risk of Code Not Being Readable Enough and Slowing Down the Implementation  2. Preparing Software Design Document  3. Preparing Coding Standard Document  4. Finishing %50 of The Project | 5.04.2025 / 19.04.2025 | 26 |
| Testing | I5 | Objectives  1. Mitigate Risk of Code Repair Wasting Too Much Time and Unreliability of Product  2. Implementing test cases  3. Preparing Software Test Result Report  4. Finishing %75 of Project | 19.04.2025 / 26.04.2025 | 23 |
| Deployment | I5 | Objectives  1. Finishing and Deploying the Project | 26.04.2025 / 17.05.2025 | 20 |

# Deployment

The transfer of the software to the production environment will be managed with CI/CD (Continuous Integration/Continuous Deployment) processes. Testing and deployment processes will be automated using GitLab CI/CD pipelines, and manual tests will be performed by testing the system as a live system.

|  |  |
| --- | --- |
| ENVIRONMENTS | |
| **Development** | Local development environment |
| **Staging** | Pre-production environment where CI/CD pipeline automated tests and manual tests are performed. |
| **Production** | Live system |

|  |  |
| --- | --- |
| DEPLOYMENT STRATEGY | |
| **Update Process** | Uninterrupted updates will be provided using the **rolling update** method. |
| **Version Control** | Different versions will be created by opening issues on Gitlab, creating different branches and making commits on these branches. The most optimized branch will be selected as the live system and the process will continue in this way by creating new branches from the selected branch according to new needs. |

|  |  |
| --- | --- |
| FAILED DEPLOYMENT MANAGEMENT AND THE ACTION FOR FIXING | |
| **Rollback Action** | If deployment fails, we will **rollback** to the previous stable version. |

|  |  |
| --- | --- |
| MONITORING AND LOGGING | |
| **Monitoring** | The system will be put live on a weekly basis and anomaly checks will be performed, and the detected anomalies will be reported by opening the Gitlab issue section. If there is no anomaly, the relevant tasks will be marked as successful. |
| **Logging** | System logs will be checked weekly and if a problem is detected, an issue will be submitted to Gitlab with the relevant log and the logs will be documented. |

# Traceability Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Works/ Team Members** | **Barış Yıldız** | **Said Çetin** | **Abdussamet Tekin** | **Muzaffer Berke Savaş** | **Mehmet Oğuz Kocadere** |
| Introduction and Project Organization Sections |  | X |  |  |  |
| Gantt Chart |  |  | X |  |  |
| Traceability Table | X |  |  |  |  |
| Deployment Section |  |  |  |  | X |
| Milestones and Objectives Section | X |  |  |  |  |
| Development Process and Measurements Section |  |  |  | X |  |
| Total Effort | 2 hours | 2 hours | 2 hours | 2 hours | 2 hours |

# 7 Prompts

# *There is no AI prompt used in the production of this document.*

# 8 Appendix

The Gantt chart of the development process is provided below.

A graph of a project schedule

AI-generated content may be incorrect.