ICES4HU
Project Plan

1 Introduction

This project plan outlines the design and implementation of a web-based system that will enable students to evaluate their instructors in a reliable and efficient manner. The system is designed to provide a quick and user-friendly environment that facilitates the evaluation process and encourages participation. Additionally, the system will provide instructors with valuable insights into their teaching performance, as well as feedback on their lesson delivery based on the evaluation results. This project plan sets out the timeline for the development and implementation of the application.

2 Project organization

Team Member	Role 1	Role 2
Numan Kafadar	Project Manager	Software Developer
Mustafa Çağrı Korkmaz	Software Architect	Software Developer
Yunus Emre Terzi	Software Configuration Manager	Software Developer
Umut Güngör	Software Tester	Software Developer
Osman Faruk Derdiyok	Software Analyst	Software Developer

2.1 Details About Team Member

Numan Kafadar (Project Manager):

- **Planning and Scheduling:** Sets timelines and allocates resources to ensure the project is completed on time.
- **Communication and Coordination:** Communicates effectively with all stakeholders and coordinates the efforts of different teams working on the project.
- **Risk Management:** Identifies and mitigates potential risks that could impact the project's success, and develops contingency plans to deal with unforeseen issues.

Mustafa Çağrı Korkmaz (Software Architect):

- **Design and Planning:** Responsible for designing and planning the overall structure and architecture of the software system, as well as defining its components and modules**.**
- **Technical Leadership:** Provides technical leadership to the development team, guiding the implementation of the software system and ensuring that it meets the required quality and performance standards.

Yunus Emre Terzi (Software Configuration Manager):

- **Configuration Management:** Managing and maintaining the project's software configuration, including version control, build management, and release management.
- **Process Improvement:** Identifies opportunities for process improvement within the project, and implements best practices to streamline development, testing, and release cycles.
- **Collaboration and Coordination:** Manager facilitates collaboration and coordination between different teams working on the project, ensuring that everyone has access to the latest software version and that changes are communicated effectively to all stakeholders.

Umut Güngör (Software Tester):

- **Test Planning:** Responsible for planning and designing test cases and test suites that ensure the software system meets the required quality and performance standards.
- **Test Execution:** Executes test cases and reports defects to the development team, ensuring that any issues are resolved before the software system is released.
- **Quality Assurance:** Responsible for ensuring the overall quality of the software system, verifying that it meets the required specifications and is fit for its intended purpose.

Osman Faruk Derdiyok (Software Analyst):

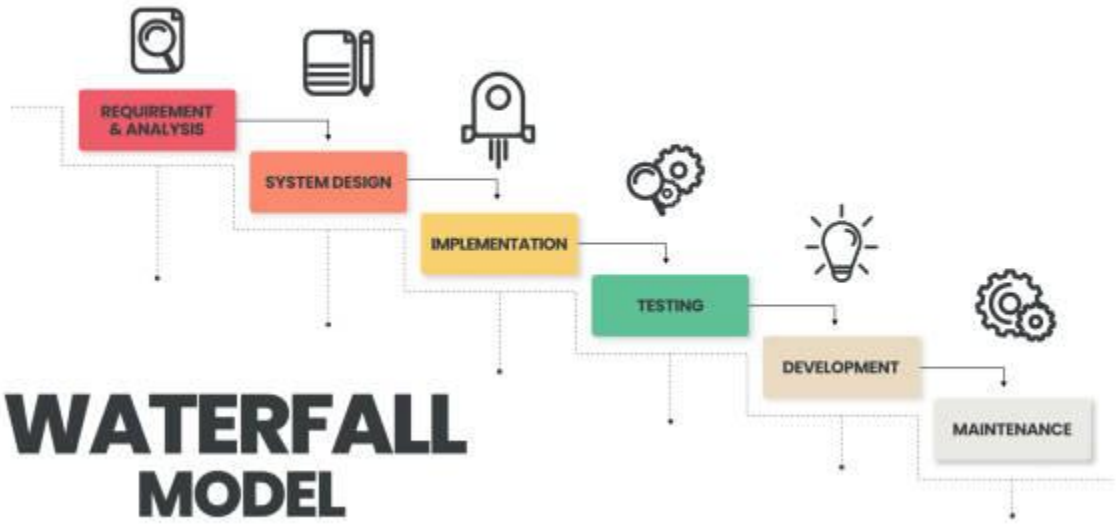
- **Requirements Gathering:** Responsible for gathering and documenting the project requirements from stakeholders, ensuring that all requirements are clearly defined and understood.
- **Analysis and Design:** A software analyst analyzes the requirements and designs the overall system architecture, ensuring that the software system is designed to meet the requirements and can be implemented within the available resources.
- **Stakeholder Communication:** A software analyst communicates the project requirements and design to stakeholders, ensuring that everyone is informed of project progress and that the final software system meets the needs of all stakeholders.

2.2 Neighboring / Referencing Projects

Evaluating lectures are important for instructors to get feedback about the lesson and improve the way they teach. Similar to this project, the following sites can be given as examples: https://bilsis.hacettepe.edu.tr, https://hadi.hacettepe.edu.tr/, https://piazza.com .

3 Development process and measurements

For our development process, we have chosen to utilize the Waterfall model, which comprises six distinct phases. The primary reason for selecting this model was its sequential approach to transactions. After exploring Agile methodologies, we concluded that the Waterfall model is better suited to our requirements. We intend to proceed systematically, following a step-by-step approach.



Tasks	Date Range	Work Load	Hours/Week
Software Project Management	17.03.2023 / 02.06.2023	11 weeks	Through the Project
Software Project and Vision	17.03.2023 / 24.03.2023	1 weeks	5
Software Project Requirements	24.03.2023 / 14.04.2023	3 weeks	8
Software Project Design	07.04.2023 / 01.05.2023	3,5 weeks	8
Software Project Development	07.04.2023 / 26.05.2023	7 weeks	16
Software Project Test	07.04.2023 / 26.05.2023	7 weeks	8

The table provides a work estimate for each group member to help plan and manage their time effectively. The group must meet every Thursday after the BBM382 lecture for two hours to discuss completed tasks, challenges, and progress on remaining tasks.

The weekly meetings will also provide an opportunity for the group to review their progress in achieving their overall project goals, and to ensure that they are on track to meet their time goals. This regular review process will help to keep the group members accountable to one another, and ensure that the project stays on course.

4 Project milestones and objectives

Phase	**Iteration**	**Primary objectives** (risks and use case scenarios)	**Scheduled start or milestone**	**Target velocity**
Developing Technical Vision	**I1**	**Project Vision and Project Plan** **Risk:** Unrealistic project goals and timelines, insufficient project planning and resources, poor communication may cause cancellation of project. **Mitigate Risk:** Defining clear and achievable plan, allocating sufficient resources, efficient management to ensure good team communication and balanced distribution of project tasks.	10.03.2023 - 24.03.2023	8 hours
Performing Requirements Analysis	**I2**	**Software Requirements Documents** **Risk:** Misidentified or unsufficient requirements can lead to delays, budget overruns and poor quality software. **Mitigate Risk:** Taking necessary time to gather all the requirements and ensure that they are complete, sufficient and accurate.	24.03.2023 - 07.04.2023	5 hours
Developing the Software Architecture	**I3**	**Architecture Notebook, Risk Management Report and Configuration Change Management Report** **Risk:** Inadequate software architecture causes failing to meet project requirements, implement features correctly. Inadequate risk identification can lead to some risks that may go unnoticed and leave the team unprepared to respond to risks. **Mitigate Risk:** Using established design patterns and testing the architecture to ensure that it is flexible and maintainable. Conducting a comprehensive review of potential risks and involving all skateholders to ensure that all risks are identified.	07.04.2023 - 21.04.2023	13 hours

Evolving the Software Design	**I4**	**Software Design Document** **Risk:** If the design is poor, the final software may not satisfy the requirements and may be difficult to implement. **Mitigate Risk:** Conducting a comprehensive review of the design to ensure it meets the requirements and is not too complex or simplistic for the project's features. Flexible and maintainable design should be chosen.	21.04.2023 - 12.05.2023	15 hours
Implementing the System	**I5**	**Project Resource Code** **Risk:** Integration issues can arise if the software is not compatible with other components of the system. Software may not perform as expected. **Mitigate Risk:** Conducting whole and integration testing to ensure software performs as expected, is functional and compatible with other components of the system. Also proper documentation must be provided.	21.04.2023 - 12.05.2023	25 hours
Performing Required Software Tests	**I6**	**Software Test Report** **Risk:** Insufficient test coverage can lead to missing important scenarios that may cause issues in the final system. **Mitigate Risk:** Wide range testing to ensure that all functionality and scenarios are covered.	12.05.2023 - 26.05.2023	10 hours
Delivering the System	**I7**	**Deploying the Final System** **Risk:** Compatibility issues can arise if the software is not compatible with the hardware or software environment in which it is being deployed. Software being not able to satisfy the needs of the users cause a decrease in number of users. **Mitigate Risk:** Applying whole testing to final software in the production environment to ensure that it performs as expected. Communicating with users and gathering feedback about their experience can make	12.05.2023 - 26.05.2023	5 hours

		stakeholders understand the needs of users to solve their problems.		

5 Deployment

Source code will be deployed to GitHub and the progress will be updated weekly. The local server will be used to execute software unless an online server is provided.

6 Lessons learned

- Creating a software project plan helps to clarify the goals and objectives of the project. This makes team members understand better what we want to achieve and how we will achieve it.
- Identifying required resources in an efficient and effective way reducing the risk of delays.
- Effective communication ensures that all stakeholders have a clear understanding of the requirements for the project.
- Large projects often require too much effort and time to complete, and the complexity of such tasks make it difficult to accomplish them individually. However working as a team leads to overcoming these challenges easily and successfully.
- Effective management ensures that the project is completed successfully and meets the requirements.