**Software Requirements Specification (SRS)**

Task Management Tool

1. Introduction

1.1 Purpose

The purpose of this document is to specify the software requirements for the Task Management Tool, a web-based application to help users manage tasks and projects efficiently. This SRS outlines the system’s functionality, user roles, and essential features.

1.2 Scope

The Task Management Tool is designed for students, workers, and anyone needing a simple tool for managing tasks and projects. The application will allow users to create projects, manage tasks, assign priorities, and track project status. The system will be accessible from any device with an internet connection.

1.3 Definitions, Acronyms, and Abbreviations

- Task: An individual activity or item to be completed within a project.

- Project: A collection of tasks with a common goal.

- User: Any individual using the application to create and manage tasks.

1.4 References

- Example task management tools such as Trello and Asana.

- PHP, HTML, JavaScript documentation for front-end and back-end development.

2. Overall Description

2.1 Product Perspective

This tool is a standalone, web-based application designed to simplify task management. It will be developed using PHP for the back-end and HTML/JavaScript for the front-end, with data stored in a relational database.

2.2 Product Functions

The core functions of the Task Management Tool include:

- Project creation and management.

- Task creation, editing, deletion, and assignment of priorities.

- Tracking task statuses (e.g., To Do, In Progress, Completed).

- Notifications for task deadlines.

2.3 User Characteristics

The tool is intended for university students, school students, and workers who need to manage simple projects and tasks effectively.

2.4 Operating Environment

- Front-end: HTML and JavaScript for browser-based access.

- Back-end: PHP on a web server.

- Database: MySQL (or other relational databases).

- Supported on modern web browsers (Chrome, Firefox, Safari).

3. Functional Requirements (mutable)

3.1 User Login

- Description: Users must login to the account to access the tool.

- Input: Username and password.

- Process: Validate user information and login the account.

- Output: Confirmation message and access to user dashboard.

3.2 Project Management

- Description: Project manager can create, view, edit, and delete projects.

- Input: Project name and description.

- Process: Store project details in the database.

- Output: A new project entry on the user’s dashboard.

3.3 Project Document

- Description: Analyzer can create and update project document and all the users can view document

- Input: Project document name, created date, creator and description.

- Process: Store project document details in the database.

- Output: A new project document on the user’s dashboard.

3.4 Project Files

- Description: Developer can create and update project files

- Input: Project file name and location

- Process: Store project fie details in the database.

- Output: A new project file on the user’s dashboard.

3.5 Task Management

- Description: Project manager can create, edit, delete, and assign tasks to projects and QA

- Input: Task name, description, creator, assignee, due date and status

- Process: Store task details in the database and link it to the appropriate project.

- Output:Task entry within the project view.

3.6 Validating the task files

- Description: QA will validate the project files

- Input: Task id

- Process: Read the task and validate

- Output: Task status will be update by QA

3.7 Chat

- Description: QA will validate the project files

- Input: Task id

- Process:

- Output:

3.5 Notification System

- Description: Notifications will alert users about upcoming task deadlines.

- Input: Task due date.

- Process: Notify the user as the deadline approaches.

- Output: Notification message on the dashboard.

4. Non-Functional Requirements

4.1 Performance Requirements

- The system should load and update task and project information within 2 seconds under normal operating conditions.

4.2 Security Requirements

- Passwords should be stored in an encrypted format.

- The system should use secure protocols (e.g., HTTPS) to protect data transmission.

4.3 Usability Requirements

- The user interface should be simple, with clearly labeled options and minimal clicks required for task or project creation.

- The tool should be accessible on both desktop and mobile devices.

4.4 Reliability Requirements

- The system should maintain 99% uptime, excluding planned maintenance periods.

- Backup and recovery procedures should be implemented to prevent data loss.

4.5 Scalability Requirements

- The application should support up to 100 concurrent users in its initial deployment phase.

5. System Architecture and Design

5.1 System Architecture

- Frontend: HTML, CSS, JavaScript for a responsive user interface.

- Backend: PHP as the server-side language.

- Database: MySQL for storing user, project, and task data.

5.2 Database Design (mutable)

- User Table: Stores user information.

- Project Table: Stores project details, linked to the user.

- Task Table: Stores task details, linked to projects.

6. Appendices

6.1 Assumptions and Dependencies

- The application requires a stable internet connection for functionality.

- Users are expected to have basic knowledge of task management.