Software Requirements Specification (SRS)

TaskManager (To-Do List Web Application)

1. Introduction

1.1 Purpose

TaskManager Web Application aims to provide users with an efficient and easy-to-use platform to manage their tasks or to-do lists. The system enables users to create, update, delete, and track tasks with various attributes such as title, description, due date, tags, and priority levels. It includes user authentication and persistent login functionality.

1.2 Scope

Task Manager Web Application is a web-based system designed for users who need to organize their tasks efficiently. The system provides the following key functionalities:

- **User Authentication:** Sign-up and Sign-in with email and password.
- **Task Management:** Users can add, update, delete, and mark tasks as completed.
- **Task Attributes:** Each task includes a title, description, due date, tags, and priority level (High, Medium, Low).
- **Task Completion Tracking:** Tasks can be checked or unchecked; completed tasks will have a strikethrough title.
- **Logout with Confirmation:** Users can log-out, with a confirmation prompt before logging out.
- **Session Management:** User credentials are stored in cookies for a seamless experience.

1.3 References

- **Node.js Documentation**
- **Express.js Documentation**
- **MongoDB Documentation**
- **NPM Documentation**
- **Bootstrap 5 Documentation**
- **Stackoverflow**
- **W3schools**
- **Github**

2. Overall Description

2.1 Product Perspective

TaskManager Web Application is a standalone web-based system that allows users to manage their daily tasks efficiently, This is for Users who need a personal task management system. It integrates authentication mechanisms and task management features while ensuring an intuitive user experience.

2.2 Product Functions

The system provides the following key functionalities:

- **User Authentication:** Secure sign-up and sign-in using email and password.
- **Persistent Login:** Users remain signed in using cookies unless they log out.
- **Task Creation:** Users can add tasks with a title, description, due date, tags, and priority level.
- **Task Modification:** Users can update task details.
- **Task Deletion:** Users can delete tasks with a confirmation prompt before deletion.
- **Task Completion:** Tasks can be marked as complete by checking a checkbox.
- **Logout Process:** Users can log-out with a confirmation prompt before logging out.

2.3 Constraints

- The system should be responsive and work on both desktop and mobile browsers.
- The database should efficiently handle multiple users and their tasks.
- Cookies should securely store encrypted user authentication data.

2.4 Dependencies

- Users must have a valid email address to register and log in.
- The system assumes an active internet connection for real-time updates.
- The application depends on JavaScript & Node.js for implementation, Express framework, MongoDB (Atlas), and Bootstrap 5 framework for UI.

3. Functional Requirements

3.1 User Authentication

- ✓ Users must be able to sign-up using an email and password.
- ✓ Users must log in using their registered credentials.
- ✓ Logged-in users should remain authenticated using cookies.
- ✓ Users can log-out, with a confirmation prompt before logout.

3.2 Task Management

- **Task Creation:** Users should be able to create a task with the following details:
  - ✓ **Task Title** (Required)
  - ✓ **Task Description** (Optional)
  - ✓ **Due Date** (Optional)
  - ✓ **Tags** (Optional)
  - ✓ **Priority Level:** High, Medium, Low (Default: low)
- **Task Modification:**
  - ✓ Users should be able to update the task details.
- **Task Deletion:**
  - ✓ Users should be able to delete a task with a confirmation prompt before deletion.
- **Task Completion:**
  - ✓ Each task has a checkbox input.
  - ✓ If checked, the task is considered completed, and its title is displayed with a strikethrough style.
  - ✓ Users can toggle the completion status by checking/unchecking the box.

3.3 Session Management

- ✓ User credentials should be securely stored in cookies for a seamless experience.
- ✓ Logged-in users should not have to sign-in again unless they log out.
- ✓ Cookies should expire after a certain period for security.

4. Non-Functional Requirements

4.1 Security

- User passwords should be hashed before being stored in the database.
- The application should prevent unauthorized access to user data.

4.2 Usability

- The UI should be intuitive and easy to use.
- The task list should be easily navigable.

4.3 Performance Requirements (optional)

- The application should respond to user actions within a few milliseconds.
- Database queries should be optimized for efficient task retrieval.

4.4 Availability (optional)

- The application should be available 24/7.
- Downtime should be minimized during maintenance or updates.

5. Use Case Scenarios

5.1 User Sign-up and Sign-in

- **Actor:** User
- **Precondition:** The User does not have an account (for Sign-up) or has an account (for Sign-in).
- **Flow:**
    1. User enters email and password.
    2. The system verifies credentials.
    3. If successful, the user is redirected to the task dashboard.

5.2 Task Creation

- **Actor:** User
- **Precondition:** The User is logged in.
- **Flow:**
    1. User clicks the “Add Task” button.
    2. User enters task details.
    3. User submits the form, and the task is added to tasklist.

5.3 Task Completion

- **Actor:** User
- **Precondition:** The User is logged in and has tasks.
- **Flow:**
    1. User clicks the checkbox next to a task.
    2. The task title gets a strikethrough.
    3. User can uncheck to mark it as incomplete and strikethrough removed from task title.

5.4 Task Deletion

- **Actor:** User
- **Precondition:** The User is logged in.
- **Flow:**
    1. User clicks delete on a task.
    2. A confirmation prompt will appear.
    3. If confirmed, the task is removed.

5.5 User Logout

- **Actor:** User
- **Precondition:** The User is logged in.
- **Flow:**
    1. User clicks “Logout.”
    2. A confirmation prompt will appear.
    3. If confirmed, the user is logged out and redirected to the login page and encrypted credentials are removed from cookies.