Software Requirements Specification (SRS)

Document Version: 1.0
Project Name: Web-Based To-Do Application
Prepared by: Hosam Mohammad
Date: 19-2-2025

---

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for a web-based To-Do application that allows users to create, manage, and organize todos efficiently. This document will serve as a guide for developers, designers, testers, and stakeholders during the development process.

1.2 Scope

The Web-Based To-Do Application will provide users with a simple and intuitive interface to:

- Create, update, and delete todos.
- Mark todos as completed.
- Access todos from any device with a web browser.

The application will be accessible via a web browser and will not require any installation.

---

2. Overall Description

2.1 Product Perspective

The To-Do application is a standalone web application that will be hosted on a cloud server. It will be accessible via modern web browsers (e.g., Chrome, Firefox, Safari, Edge).

2.2 User Classes

- **End Users:** Individuals who will use the application to manage their todos.

2.3 Operating Environment

- **Frontend:** HTML, CSS, JavaScript Next.js or Vue.js recommended).
- **Backend:** Node.js or Next.js.
- **Database:** PostgreSQL, MySQL, or MongoDB.
- **Hosting:** Containerized on cloud-based hosting (e.g., AWS, Heroku, or DigitalOcean) behind an nginx proxy manager.

2.4 Design and Implementation Constraints

- The application must be responsive and work on both desktop and mobile devices.
- The application must support modern web browsers.
- Data must be securely stored and transmitted (HTTPS).

---

3. System Features

3.1 User Authentication

- Users can create an account using an email address and password.
- Password must be between 3 and 10 characters in length.
- Users can log in and log out of their accounts.
- Password reset functionality will be available.

3.2 Todo Management

- Users can create new todos with the following attributes:
    - Title (required)
- Users can edit or delete existing todos.

3.4 User Interface

- The interface will be clean, intuitive, and responsive.
- Todos will be displayed in a list.

---

4. Functional Requirements

ID	Requirement	Description
FR-1	User Registration	Users can create an account.
FR-2	User Login/Logout	Users can log in and log out of their accounts.
FR-3	Create Todo	Users can create a new todo with a title, description, due date, and priority.
FR-4	Edit Todo	Users can edit an existing todo.
FR-5	Delete Todo	Users can delete a todo.
FR-6	Mark Todo as Completed	Users can mark a todo as completed.

---

5. Non-Functional Requirements

ID	Requirement	Description
NFR-1	Performance	The application should load in under 3 seconds.
NFR-2	Scalability	The application should support up to 10,000 concurrent users.
NFR-3	Security	User data must be encrypted during transmission and storage.
NFR-4	Availability	The application should have 99.9% uptime.
NFR-5	Usability	The application should be intuitive and easy to use.

ID	Requirement	Description
NFR-6	Compatibility	The application should work on all major web browsers.

---

6. External Interface Requirements

6.1 User Interfaces

- **Home Page:** Displays a list of todos and options to create new todos.
- **Todo Details Page:** Displays detailed information about a todo.
- **Login/Register Page:** Allows users to log in or create an account.

6.2 Hardware Interfaces

- The application will run on standard web servers and cloud hosting platforms.

6.3 Software Interfaces

- **Frontend:** HTML, CSS, JavaScript.
- **Backend:** RESTful API for communication between frontend and backend.
- **Database:** SQL or NoSQL database for storing user data and todos.

6.4 Communication Interfaces

- The application will use HTTPS for secure communication.

---

7. Other Non-Functional Requirements

7.1 Performance Requirements

- The application should handle up to 10,000 concurrent users without performance degradation.

7.2 Security Requirements

- User passwords must be hashed and salted before storage.
- The application must use HTTPS to encrypt data in transit.

7.3 Maintainability

- The codebase should be well-documented and follow best practices for readability and scalability.

---

8. Appendices

8.1 Glossary

- **Todo:** A single to-do item with a title, description, due date, and priority.
- **Category/Project:** A group of related todos.
- **Reminder:** A notification to alert the user about an upcoming todo.

8.2 References

- https://www.blazemeter.com/blog/selenium-github

This SRS document provides a comprehensive overview of the requirements for the Web-Based To-Do Application. It can be further refined based on stakeholder feedback and project-specific needs.

Sprint Breakdown for the Web-Based To-Do Application

Sprint 0: Preparation

Duration: 1 week
Objective: Set up the project, define requirements, and prepare the testing environment.

Activities:

1. **Requirement Gathering:**
    - Define user stories and acceptance criteria.
    - Prioritize features for the MVP.
2. **Test Planning:**
    - Create a high-level test plan.
    - Identify testing tools and frameworks.
3. **Environment Setup:**
    - Set up development, and test environments.
    - Configure CI/CD pipelines (e.g., Jenkins, GitHub Actions).
4. **Test Case Design:**
    - Draft test cases for core features (e.g., user registration, todo creation).

Sprint 1: User Authentication

Duration: 2 weeks
Objective: Implement and test user registration and login functionality.

Features Delivered:

- User registration (email and password).
- User login and logout.
- Password recovery.

Testing Activities:

1. **Unit Testing:**

- Test backend logic for user registration and authentication.

2. **Integration Testing:**
    - Test API endpoints for user registration and login.
3. **System Testing:**
    - Test end-to-end workflows for registration, login, and password recovery.
4. **Defect Reporting:**
    - Log and prioritize defects in the tracking tool.

Sprint 2: Todo Management

Duration: 2 weeks
Objective: Implement and test core todo management features.

Features Delivered:

- Create, edit, and delete todos.
- Mark todos as completed.

Testing Activities:

1. **Unit Testing:**
    - Test backend logic for todo creation, editing, and deletion.
2. **Integration Testing:**
    - Test API endpoints for todo management.
3. **System Testing:**
    - Test end-to-end workflows for todo creation, editing, and deletion.
4. **Regression Testing:**
    - Ensure user authentication features are not broken.

Sprint 3: User Interface and Usability

Duration: 2 weeks
Objective: Finalize the user interface and ensure usability.

Features Delivered:

- Clean and intuitive UI design.
- Responsive design for desktop and mobile devices.
- Light and dark theme support.

Testing Activities:

1. **UI Testing:**
    - Test the application on different devices and browsers.
2. **Usability Testing:**
    - Conduct usability tests with real users.
3. **Accessibility Testing:**
    - Ensure the application is accessible to users with disabilities.
4. **Regression Testing:**
    - Ensure all previous features are not broken.

Sprint 4: Final Testing and Release

Duration: 2 weeks
Objective: Conduct final testing and prepare for release.

Activities:

1. **User Acceptance Testing (UAT):**
    - Validate the application against business requirements.
2. **Performance Testing:**
    - Test the application under load to ensure it meets performance requirements.
3. **Security Testing:**

  - Verify that user data is securely stored and transmitted.

4. **Bug Fixing:**
   - Address any critical defects identified during UAT.
5. **Release Preparation:**
   - Prepare release notes and documentation.
   - Deploy the application to the production environment.

Summary of Sprints

Sprint	**Focus Area**	**Key Deliverables**
Sprint 0	Preparation	Test plan, environment setup, test case design.
Sprint 1	User Authentication	User registration, login, password recovery.
Sprint 2	Todo Management	Create, edit, delete, and mark todos as completed.
Sprint 3	Todo Organization and Filtering	Organize todos, filter todos, search todos.
Sprint 4	Notifications and Reminders	Set reminders, display notifications, send email reminders.
Sprint 5	User Interface and Usability	Clean and responsive UI, light/dark themes.
Sprint 6	Final Testing and Release	UAT, performance testing, security testing, bug fixing, and release preparation.

This sprint breakdown ensures that the **Web-Based To-Do Application** is developed and tested incrementally, with a focus on delivering value to the end-user in each sprint. Let me know if you need further adjustments or details!

User stories

Sprint 0: User Authentication

Sprint 1: Todo Management

1. **As a user, I want to create a new todo item so that I can track tasks.**
   - **Acceptance Criteria:**
     - User must provide a title for the todo.
     - Todo should be saved successfully.
   - **Test Cases:**
     - Verify that a new todo can be created with a title.
     - Ensure todos are persisted correctly.
2. **As a user, I want to edit a todo item so that I can update the title.**
   - **Acceptance Criteria:**
     - Users must be able to modify todo title.
   - **Test Cases:**
     - Ensure todos can be edited and updated correctly.
3. **As a user, I want to delete a todo item so that I can remove tasks I no longer need.**
   - **Acceptance Criteria:**
     - Deleted todos should not be retrievable.
   - **Test Cases:**
     - Validate that deleting a todo removes it permanently.
4. **As a user, I want to be able to mark a todo as "completed".**
   - **Test Cases:**
     - Validate that a new todo can be set as "completed".
     - Validate that a todo set to be "completed" can be set as "un-completed".

Sprint 2: User Interface and Usability

1. **As a user, I want a clean and intuitive UI so that I can navigate easily.**
   - **Acceptance Criteria:**
     - UI should be user-friendly and accessible.
   - **Test Cases:**
     - Verify UI elements are correctly aligned and accessible.
2. **As a user, I want the application to be responsive so that it works on both desktop and mobile devices.**
   - **Acceptance Criteria:**
     - Layout should adjust based on device type.
   - **Test Cases:**
     - Test responsiveness across different screen sizes and browsers.

Sprint 3: Final Testing and Release

1. **As a business owner, I want to validate the application against business requirements so that it meets expectations.**
   - **Acceptance Criteria:**
     - Application should function as per initial requirements.
   - **Test Cases:**
     - Conduct user acceptance testing to validate functionality.
2. **As a security analyst, I want to verify user data security so that user information is protected.**
   - **Acceptance Criteria:**
     - Sensitive data should be encrypted.
   - **Test Cases:**

- Verify security measures, including encryption and authentication mechanisms.

This structure ensures each sprint has clear user stories with well-defined acceptance criteria and corresponding test cases.

Team members

محمد عباس مختار

- 01210616777
- Mohamedabbas81@icloud.com

إبراهيم سمير إبراهيم محمد

- 01002571986
- Ibrahimsamir87@outlook.com

أسماء الدالي

- 01128738125
- asmaaeldaly94@gmail.com

حسام محمد علي البسيوني

- 01014124943
- hosamelbasiony@gmail.com

محمود حامد

- 01271373720
- hamd40455@gmail.com

زياد احمد اسماعيل

- 01000465078
- ziadesmail80@gmail.com

Contents