**Software Requirements Specification (SRS)**

**1. Introduction**

**1.1 Purpose**

The purpose of this SRS document is to outline the requirements for a web-based To-Do List application. The application aims to help users manage their tasks effectively by allowing them to add, edit, complete, and delete tasks. The tasks will be stored persistently in the browser's local storage.

**1.2 Scope**

The To-Do List application will be a single-page application (SPA) that provides the following functionalities:

* Users can add tasks with descriptions.
* Users can edit existing tasks inline.
* Users can mark tasks as completed.
* Users can delete tasks.
* Tasks will persist across sessions using local storage.

**1.3 Definitions, Acronyms, and Abbreviations**

* **SPA**: Single Page Application
* **UI**: User Interface
* **UX**: User Experience

**1.4 References**

* Web Development Standards
* HTML, CSS, JavaScript documentation

**2. Overall Description**

**2.1 Product Perspective**

The To-Do List application will function as a standalone web application accessible via modern web browsers. It will be developed using HTML, CSS, and JavaScript, leveraging the browser's local storage for data persistence.

**2.2 Product Functions**

* **Task Management**: Users can create, read, update, and delete tasks.
* **Task Completion**: Users can mark tasks as complete, which visually distinguishes them.
* **Data Persistence**: All tasks are stored in the local storage, ensuring they are retained across sessions.

**2.3 User Classes and Characteristics**

* **General Users**: Users who want to manage personal or professional tasks. They may have varying levels of technical expertise but should find the application intuitive and easy to use.

**2.4 Operating Environment**

The application will run on any modern web browser (Chrome, Firefox, Safari, Edge) and will be accessible on both desktop and mobile devices.

**2.5 Design and Implementation Constraints**

* The application must comply with HTML5 and CSS3 standards.
* The application will not require server-side code and will rely solely on client-side technologies.

**3. Functional Requirements**

**3.1 Task Management**

* **3.1.1 Add Task**
  + **Description**: Users can input a new task in a text field and click a button to add it to the task list.
  + **Input**: Task description (string).
  + **Output**: Task appears in the task list.
* **3.1.2 Edit Task**
  + **Description**: Users can edit an existing task by clicking an "Edit" button that replaces the task text with an input field.
  + **Input**: New task description (string).
  + **Output**: Updated task description appears in the task list.
* **3.1.3 Complete Task**
  + **Description**: Users can mark a task as complete by clicking a "Complete" button, which visually distinguishes completed tasks.
  + **Input**: Task ID or index.
  + **Output**: Task is visually marked as completed.
* **3.1.4 Delete Task**
  + **Description**: Users can remove a task from the list by clicking a "Delete" button.
  + **Input**: Task ID or index.
  + **Output**: Task is removed from the task list.

**3.2 Data Persistence**

* **3.2.1 Save Tasks**
  + **Description**: The application saves tasks to the browser's local storage after any operation (add, edit, delete).
  + **Input**: Task list (array of objects).
  + **Output**: Tasks are saved in local storage for future access.
* **3.2.2 Load Tasks**
  + **Description**: The application retrieves tasks from local storage upon loading.
  + **Input**: None.
  + **Output**: Tasks are displayed in the task list.

**4. Non-Functional Requirements**

**4.1 Performance Requirements**

* The application should load tasks from local storage and render them within 2 seconds.

**4.2 Usability Requirements**

* The user interface should be intuitive, allowing users to add, edit, complete, and delete tasks with minimal effort.
* The application should be responsive, ensuring usability on both desktop and mobile devices.

**4.3 Security Requirements**

* The application should not expose sensitive information since it is a client-side application.

**4.4 Compatibility Requirements**

* The application should work seamlessly across all modern web browsers (Chrome, Firefox, Safari, Edge).

**5. System Features**

**5.1 User Interface**

* The application will consist of a single-page layout with:
  + A header displaying the title "To-Do List".
  + An input area for adding new tasks.
  + A list area to display all tasks.
  + Each task will include options to edit, complete, and delete.

**5.2 User Interaction**

* Users will interact with the application using a mouse or touch interface to click buttons and enter text.

**6. Conclusion**

This Software Requirements Specification outlines the objectives and requirements for the To-Do List application. By following this SRS, the development team can create a functional and user-friendly application that meets user needs and expectations.