Software Requirements Specification (SRS)
To do List

1. Introduction

Introducing the Software Requirement Specification (SRS) for our innovative 'To-Do List' project. This document outlines the key features and functionalities aimed at providing users with a simplified, organized, and collaborative task management experience. Join us on this development journey as we detail the requirements for creating an intuitive and efficient solution that enhances how users approach their daily tasks. Let's redefine productivity together with our To-Do List application.

1.1 Document Purpose

The purpose of this Software Requirement Specification (SRS) document is to provide a comprehensive blueprint for the development of our 'To-Do List' application. This document outlines the functional and non-functional requirements essential for creating a user-centric task management solution. By defining the scope, features, and constraints of the project, the SRS serves as a guiding document for developers, testers, project managers, and stakeholders. The ultimate goal is to articulate a clear vision and roadmap, ensuring the successful design, implementation, and deployment of an intuitive and effective To-Do List application.

1.2 Scope

The 'To-Do List' app aims to streamline task management with features for creation, editing, and categorization. Collaboration tools facilitate easy sharing, while a user-friendly interface ensures accessibility. A robust reminder system enhances timely task management. The app prioritizes data security, and reporting tools offer insights for productivity optimization.

1.4 Definitions, Acronyms, and Abbreviations:

Definitions:

To-Do List: A digital or physical list that individuals use to record and organize tasks or activities they need to complete.

Task Management: The process of planning, organizing, and tracking tasks to ensure they are completed efficiently and effectively.

Collaboration Tools: Software or features that enable users to work together on tasks, projects, or documents, enhancing communication and teamwork.

Reminder System: A mechanism within an application that prompts users with notifications or alerts about upcoming tasks or deadlines.

User Interface (UI): The visual and interactive elements of a software application, encompassing screens, pages, buttons, and other components that users interact with.

Data Security: Measures and protocols implemented to protect digital data from unauthorized access, alteration, or destruction.

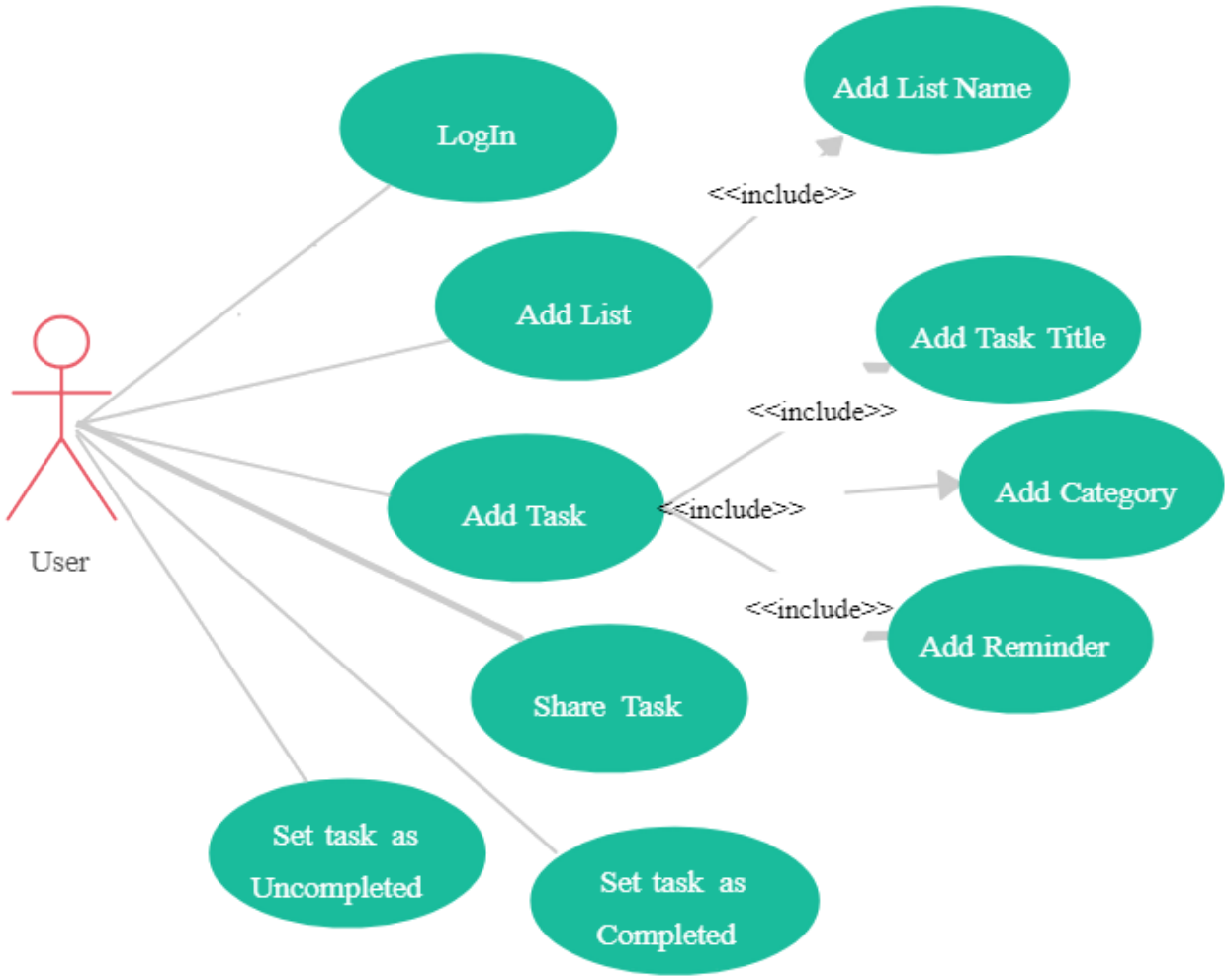
Acronyms and Abbreviations:

UI: User Interface
API: Application Programming Interface
SRS: Software Requirement Specification
SSL: Secure Sockets Layer
CMS: Content Management System
UX: User Experience

2. Overall Descriptions

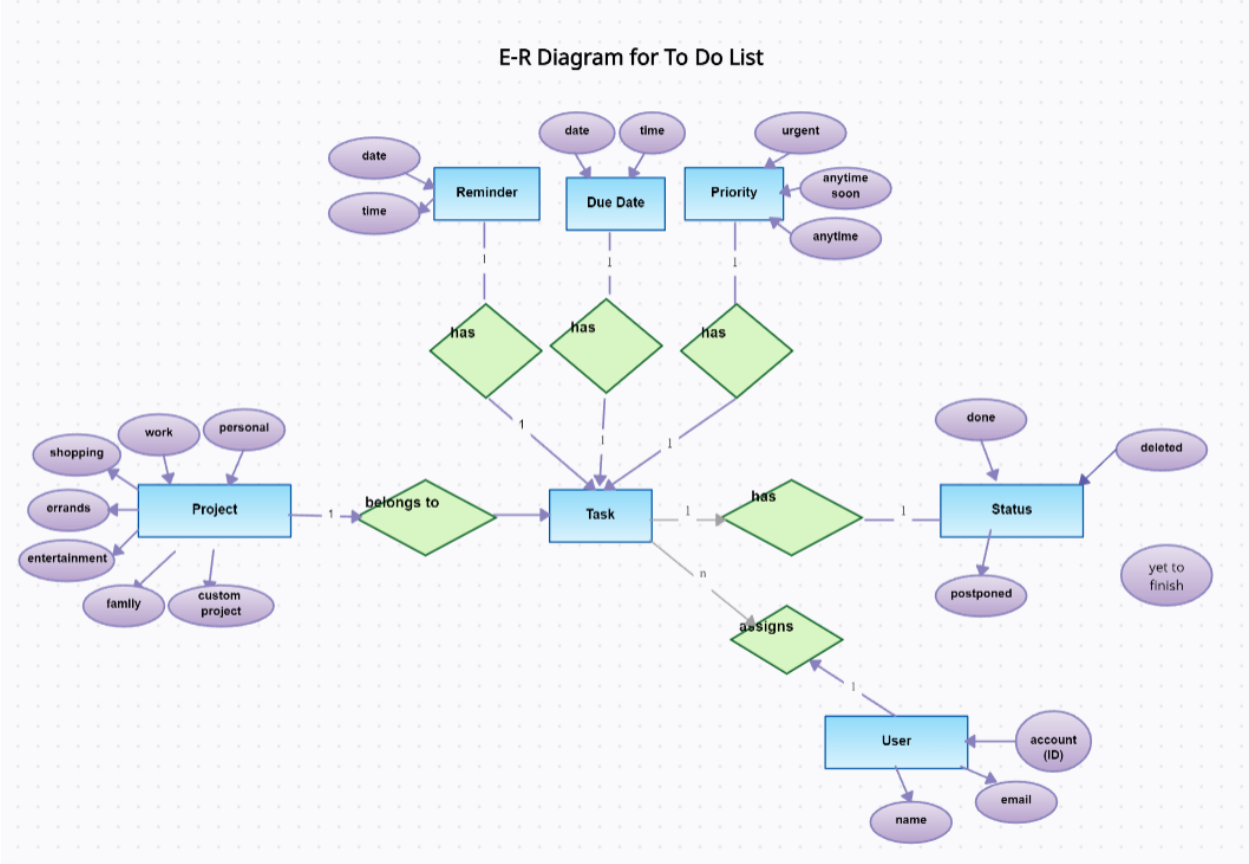
2.1 Product Perspective (Use Case Diagram)

In the product perspective of the 'To-Do List' application, users engage with key functionalities for streamlined task management. The process begins with user login, where individuals provide authentication credentials for secure access. Once logged in, users have the ability to create task lists and populate them with specific tasks, detailing each item with relevant information. The application allows users to categorize tasks for efficient organization and sharing, facilitating collaboration among users. Additionally, users can mark tasks as complete upon their execution or, conversely, mark completed tasks as incomplete for dynamic task management. These fundamental use cases collectively define the user experience within the application, ensuring a user-friendly and versatile platform for effective task tracking and collaboration.



2.2 Product Function (Entity Relationship Diagram)

The 'To-Do List' app simplifies task management, offering users a seamless interface to create, organize, and prioritize tasks. With collaborative features, users can share tasks and enhance teamwork. The intuitive design allows for easy tracking of due dates, marking tasks as complete or incomplete, and efficient categorization. The app strikes a balance between simplicity and versatility, empowering users for efficient daily task navigation.



2.3 User Classes and Characteristics:

Regular User:

Characteristics: Regular users are individuals who utilize the 'To-Do List' app for personal task management.
Behaviors: They create, edit, and organize tasks, customize lists, and leverage reminder features for efficient daily planning.

Guest User:

Characteristics: Guest users have limited access, typically with read-only permissions.
Behaviors: They view shared task lists without the ability to modify or create tasks.

Mobile User:

Characteristics: Mobile users primarily access the application via smartphones or tablets.
Behaviors: They rely on the mobile interface for quick task updates, notifications, and on-the-go productivity.

2.4 Operating Environment:

The 'To-Do List' application thrives in a versatile operating environment, supporting desktops (Windows, macOS, Linux) and mobile devices (iOS, Android). It seamlessly integrates with major web browsers, ensuring accessibility for users across platforms. The application's online nature requires a stable internet connection for real-time collaboration and synchronization. Adhering to industry-standard security measures, it maintains data confidentiality and integrity. With minimal hardware requirements, the app is designed for efficient performance on a wide range of devices.

2.5 Assumptions and Dependencies:

Internet Connectivity: The app assumes users have consistent internet access for real-time collaboration.

Security Practices: It assumes users follow security best practices for data protection.

Third-Party Integration: Assumption is users may integrate third-party services, subject to their availability and compatibility.

Device Compatibility: Assumes users meet minimal hardware and software requirements for optimal performance.

User Data Accuracy: The app relies on users inputting accurate and up-to-date task information.

Software Updates: Assumes users regularly update the application for access to new features and improvements.

2.6 Requirements (Software Configuration):

The 'To-Do List' application is designed for easy accessibility and usability. It operates seamlessly on common platforms like Windows, macOS, Linux, iOS, and Android, ensuring widespread availability. Compatibility with popular web browsers—Google Chrome, Firefox, Edge, and Safari—guarantees a user-friendly experience. The app uses simple database solutions like SQLite for straightforward data storage. Server-side scripting with PHP and client-side scripting with JavaScript ensures smooth functionality. Git is employed for version control, while testing is done using easy-to-implement frameworks like Mocha. Deployment is simplified through cloud services like Heroku, making it accessible to a wide user base. This straightforward software configuration prioritizes ease of use and accessibility across diverse environments.

2.7 Data Requirements:

User Data:
Attributes: User ID, username, password (hashed for security), email.
Purpose: Identification, authentication, and communication.

Task List Data:
Attributes: List ID, user ID (foreign key), list name, color theme.
Purpose: Organization and customization of task lists.

Task Data:
Attributes: Task ID, list ID (foreign key), task name, due date, priority, description, status.
Purpose: Detailed information for individual tasks, tracking completion status.

Category Data:
Attributes: Category ID, category name.
Purpose: Categorization of tasks for efficient organization.

3. External Interface Requirements

3.1 User Interfaces:

User-friendly Web Interface:

The application will feature an intuitive and responsive web interface accessible through major browsers (Google Chrome, Firefox, Safari, Edge).

Mobile Application Interface:

A mobile-friendly interface for iOS and Android devices will be provided, ensuring a seamless experience on smartphones and tablets.

3.2 Hardware Interfaces:

Minimal Hardware Requirements:

The application is designed to operate efficiently on standard desktop and laptop configurations (Windows, macOS, Linux) and mobile devices (iOS, Android).

Compatibility Across Devices:

The application aims for compatibility with a range of hardware, allowing users to access it on devices with varying specifications.

3.3 Software Interfaces:

Database Management System (DBMS):

The application will interface with a lightweight database system, such as SQLite, for efficient data storage and retrieval.

Server-Side Scripting:

Server-side scripting using simple technologies like PHP will facilitate backend processes and user authentication.

Client-Side Frameworks:

Lightweight client-side frameworks like Bootstrap will be employed for responsive and user-friendly interfaces.

3.4 Communications Interfaces:

Secure Sockets Layer (SSL):

SSL encryption will be implemented to secure communication between the user's device and the server, ensuring data confidentiality.

Notification Services:

Integration with basic notification services, such as email notifications, will provide users with timely reminders and updates.

4. System Features

system features collectively contribute to the functionality and versatility of the 'To-Do List' application, offering users a comprehensive tool for efficient task management and collaboration.

4.1 Functional Requirement

User Login:

Description: Users can create accounts and log in securely.

Task Management:

Description: Adding, editing, and deleting tasks easily.

List Organization:

Description: Creating and managing task lists effortlessly.

Priority and Categories:

Description: Marking task priorities and assigning categories.

Collaboration:

Description: Sharing task lists for teamwork.

Task Status:

Description: Marking tasks as done or pending.

Reminders:

Description: Getting timely reminders for upcoming tasks.

Customization:

Description: Personalizing lists and themes.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Responsiveness: The application should respond to user interactions within 2 seconds.

Scalability: The system should handle concurrent users and data growth without performance degradation.

5.2 Safety Requirements

Data Integrity: The application must ensure the integrity of user data, preventing accidental loss or corruption.

User Authentication: Robust user authentication mechanisms should safeguard user accounts.

5.3 Security Requirements

Secure Data Transmission: All data transmitted between the user's device and the server should be encrypted using SSL.
Access Control: Users should only have access to their own data, ensuring data privacy.

5.4 Software Quality Attributes

Usability: The application should have an intuitive interface for easy user adoption.
Reliability: The system should operate without frequent disruptions, ensuring continuous availability.

5.5 Business Rules

User Account Limits: Each user is allowed to have only one active account for data integrity.

Task Delegation: Users can delegate tasks to others, promoting collaboration.