Web-Based To-Do List Application – Requirements Document

---

1. Overview

1.1 Purpose

This document outlines the requirements for a **web-based To-Do List Application** that enables users to efficiently manage and organize their daily tasks from any browser. The goal is to provide a simple, responsive, and user-friendly interface for personal productivity.

1.2 Scope

The application will:

- Allow users to create, update, organize, and delete tasks via a browser
- Include essential features like due dates, task categories, reminders, and priorities
- Be responsive and accessible from both desktop and mobile browsers
- Persist data in a secure backend with account-based access

1.3 Intended Audience

- Product Managers
- Software Developers
- UI/UX Designers
- QA Engineers
- Project Stakeholders

---

2. Functional Capabilities of the Services:

2.1 Authorization Service:

2.1.1 POST /register: User registers with email and password.

2.1.2 POST /login: User login with correct email and password

2.1.3 POST /logout: User should be able to logout.

2.1.4 POST /reset: Use should be able to recover the password.

2.2 REST API Endpoints:

2.2.1 GET /todos: Retrieves all tasks for the logged-in user. Supports filtering by completion status, Name, and sorting capabilities to help users organize and find their tasks quickly.

2.2.2 POST /todos: Creates a new task with name, description and completion status. Automatically triggers JIRA ticket creation and sends an email notification to the user with the JIRA ticket link.

2.2.3 GET /todos/{id}: Fetches details of a specific task by its ID, including all task properties, subtasks, and metadata.

2.2.4 PUT /todos/{id}: Updates an existing task's properties such as name and completion status

2.2.5 DELETE /todos/{id}: Permanently deletes a task from the user's account for the given id.

2.3 Event Processing Consumers:

2.3.1 CONSUME on todo-items-audit: Processes and persists audit logs for all TODO-related operations, ensuring compliance tracking and maintaining a complete audit trail.

2.3.2 CONSUME on todo-items-email → PRODUCE on todo-items-audit: Handles email notifications for new tasks. Sends emails to task creators containing direct links to associated JIRA tickets, then logs these notification events for auditing.

2.3.3 CONSUME on todo-items-jira → PRODUCE on todo-items-audit, todo-items-email: Manages JIRA integration for new tasks. Creates corresponding JIRA tickets automatically, triggers email notifications with ticket links, and generates audit records for all JIRA operations.

3. Functional Requirements

2.1 User Accounts

- **FR1.1**: Users can register using an email and password.
- **FR1.2**: Users can log in and log out securely.
- **FR1.3**: Password recovery/reset functionality must be available.

2.2 Task Management

- **FR2.1**: Users can create tasks with a name (required) and optional description.
  → **The task name must be between 5 and 250 characters.**
- **FR2.2**: Users can set due dates and times for tasks.
- **FR2.3**: Users can mark tasks as complete or incomplete.
- **FR2.4**: Users can delete or edit existing tasks.
- **FR2.5**: Users can assign priority levels (e.g., Low, Medium, High).
- **FR2.6**: Users can create subtasks under main tasks.

2.3 Task Organization

- **FR3.1**: Users can create multiple task lists (e.g., "Work", "Personal").
- **FR3.2**: Users can categorize or tag tasks.
- **FR3.3**: Users can filter tasks by category, due date, completion status, or priority.
- **FR3.4**: Users can sort tasks (e.g., by due date, creation date, or priority).

2.4 Notifications

- **FR4.1**: Users can set optional email reminders for tasks.
- **FR4.2**: Reminders are triggered before task due time (customizable by user).

2.5 Search

- **FR5.1**: Users can search for tasks by name or keywords.

2.6 Data Persistence

- **FR6.1**: User data is stored in a backend database.
- **FR6.2**: Data is tied to individual user accounts and securely retrieved upon login.

Here are the enhanced requirements, incorporating JIRA ticket creation, email notifications for new TODO list items with JIRA links, and auditing:

2.7 Integration with JIRA & Notifications

- **FR7.1**: Upon creation of a new TODO list item, a corresponding JIRA ticket must be automatically created.
- **FR7.2**: An email notification must be sent to the task creator upon successful JIRA ticket creation for a new TODO item.
- **FR7.3**: The email notification must include a direct link to the newly created JIRA ticket.
- **FR7.4**: All requests for new TODO list items and their corresponding email notifications must be audited for tracking and compliance purposes.

---

3. Non-Functional Requirements

3.1 Usability

- The application must have a clean, responsive UI compatible with all major browsers (Chrome, Firefox, Safari, Edge).
- It should follow accessibility best practices (WCAG AA compliance preferred).

3.2 Performance

- Web pages should load within 2 seconds on average connections.
- All interactions (add/edit/delete tasks) must respond in under 300ms.

3.3 Security

- Use HTTPS for all traffic.
- Store passwords using salted hashing (e.g., bcrypt).
- Authenticate users using secure tokens (e.g., JWT).

3.4 Availability

- The system should be available 99.5% of the time, excluding planned maintenance.

3.5 Scalability

- The backend must support scaling to at least 10,000 concurrent users.

---

4. User Stories

ID	User Story
US1	As a user, I want to register and log in so that I can manage my tasks securely.
US2	As a user, I want to create and edit tasks so that I can track my work.
US3	As a user, I want to set reminders so that I don’t miss important deadlines.
US4	As a user, I want to filter and sort my tasks to quickly find what I need.
US5	As a user, I want to search tasks by keyword to locate specific items quickly.
US6	As a user, I want my data to sync when I reload the browser or log in from another device.