System Requirements Specification (SRS)

Ticktrak App

Version: 1.3
Last Updated: June 17, 2025
Author: Carlo Merin

I. Introduction

A. Purpose

Businesses or individuals struggle to track their tasks and sub tasks efficiently. So Ticktrak App aims to provide a clear platform for task progress and team collaboration and designed to help users manage their tasks and track progress. The main objective of this is to enable the users to create, view, update, delete and complete tasks while retain the view of their progress.

B. Scope

This Application includes:

- Welcome/Landing Page
- User Authentication (Login/Register)
- Dashboard View
- Side and Top Navigation Menu
- User Profile/Setting
- Task List (Table with search and pagination controls)
- Sub Task List (each task have multiple sub tasks)
- Progress Tracking Chart (percentage of completed sub tasks)

C. Tech Stack

This Application uses:

1. PHP
   - Laravel v.10 PHP
2. Database:
   - MySQL
3. JavaScript
   - Node JS, Alpine JS, Rest API / Fetch
4. CSS
   - Tailwind CSS, Flowbite
5. Icons
   - Heroicons, Untitledui
6. Graphs

- Apexcharts

II. Definition, Acronyms and Abbreviation

A.

Terms	Definition
SRS	Software Requirement Specification
API	Application Programming Interface
CRUD	Create, Retrieve, Update, Delete

III. User Role and Permissions

A. User:

1. Manage own profile.
2. Create, View, Update and Delete Tasks and Sub Tasks.

B. Admin:

1. Manage All the users.
2. Perform Administrative tasks.

IV. Functional Requirements

A. User Authentication

1. Register and Login

B. User Profile/Setting

1. Add Profile Picture
2. Update Information
3. Change Password
4. Delete Account

C. Task Management

1. Create new Tasks
2. View List of Tasks
3. Update / Delete Tasks

D. Sub Task Management

1. Create new Sub Tasks
2. View List of Sub Tasks
3. Update / Delete Sub Tasks

E. Progress Tracking

1. Calculate the percentage of completed and total sub tasks
   - Percentage = (number of completed sub tasks / total number of sub tasks) * 100
2. Add Progress bar and chart for representation
3. Add Confetti for interaction

V. Non-Functional Requirements

A. Security: Password hashing

B. Performance: Response should be 500ms minimum

C. Validation: Proper input sanitation and error message

VI. Database Schema (Tables)

A. Table: Users

1. id – Primary Key
2. name – String | max:255
3. profile – String | Null
4. email – String | Email | Unique
5. password – String | Hash

B. Table: Tasks

1. id – Primary Key
2. title – Required | String | max:255
3. details – Null | text
4. priority – Required | String | enum [‘low’, ‘medium’, ‘high’] | default = low
5. status – Required | String | enum [‘pending’, ‘completed’] | default = pending
6. due_date – Null | date
7. user_id – Foreign Key | exists: users, id | delete: cascade

C. Table: Sub Tasks

1. id – Primary Key
2. description – Required | text
3. is_complete – Boolean | String | default = false
4. due_date – Null | date

5. task_id – Foreign Key | exists: tasks, id | delete: cascade

VII. Routes (API)

A. Task API Endpoints (API/TaskApiController)

1. middleware – auth:sunctum

Name	Method	Url	description
index	GET	/api/tasks	View task list
store	POST	/api/tasks	Create new tasks
update	UPDATE	/api/tasks/{id}	Update selected tasks
destroy	DELETE	/api/tasks/{id}	Delete selected tasks

B. Sub Task API Endpoints (API/SubTaskApiController)

1. middleware – auth:sunctum

Name	Method	Url	description
index	GET	/api/tasks/{taskId}	View subtask list
store	POST	/api/tasks/{taskId}	Create new subtasks
Update Mark	PATCH	/api/tasks/{taskId}/subtasks/{subtaskId}	Update selected subtasks to complete/done
Destroy	DELETE	/api/tasks/{taskId}/subtasks/{subtaskId}	Delete selected subtasks

VIII. Models

A. User

1. Fillable Attributes
   - name | profile | email | password
2. Relationship
   - hasMany(Task)

B. Task

1. Fillable Attributes
   - title | details | priority | status | due_date | user_id
2. Relationship
   - belongsTo(User)
   - hasMany(SubTasks)

C. SubTask

1. Fillable Attributes
    - description | due_date| is_complete | task_id
2. Relationship
    - belongsTo(Task)

IX. User Interface

A. Top Navbar

1. Search Function
2. Dark/Light Mode
3. User Profile (Name and Email)

B. Side Navbar

- ➢ Default View: Shows only Icons (Menu and Others)
- ➢ Toggled View: Shows both Icons and label for each items in the side panel

1. Application Logo
2. Menu (placeholder)
    - Dashboard
    - My Task
3. Others (placeholder)
    - Setting
    - Logout

C. Task List UI (CRUD)

1. Header:
    - Label: Task List
    - Button: Add Tasks (Modal)
        - ➢ Title: Add Task
        - ➢ Input Fields: Title, Priority, Due Date, Details
        - ➢ Close Button: Close
        - ➢ Submit Button: Add Task
2. Main:
    - Search Box
    - Table

i. # | Title | Progress | Priority | Due Date | Action

- *Note: Remove the **description column in a table** to be replace by **progress bar** for initial representation of the sub tasks completed percentage.*

ii. Action:

- Anchor Tag Button: Jump Into Sub Task page with Selected Task
- Button: Update Task (Modal)
  - Title: Update Task
  - Input Fields: Title, Priority, Due Date, Details
  - Close Button: Close
  - Submit Button: Update Task
- Button: Delete Task (Modal)
  - Title: Delete Task
  - Description: Are you sure you want to delete this task?
  - Close Button: Close
  - Submit Button: Delete Task

- Pagination Controls

i. Limit per page (10, 25, 50, etc.)

D. Sub Task UI (CRUD)

1. Header:
   - breadcrumb: Go back to the task page
2. Main:
   - Grid

i. Left Side

- Chart: Milestones Progress
  - ➢ Chart
  - ➢ Priority
  - ➢ Deadline
- Form: Add Milestones
  - ➢ Input Field: Description, Due Date
  - ➢ Submit Button: Add Milestones

ii. Right Side

- Title: Title of the selected tasks

- Details: Details of the selected tasks
- Notes: Please Check the box after completing the sub tasks
- Table:
  - ➢ Checkbox | Description | Deadline | Action
    - Checkbox: to mark the sub tasks as complete
    - Action: Delete

Change Logs:

1. v.1.1 (June 14, 2025)

   Add Collapsible sidebar to maximize the space of the content of the main page.

   1. Default: showing sidebar icons only.
   2. Toggled: shows both icons and label.

2. v.1.2 (June 15, 2025)

   Implementing CRUD API for Tasks.

   1. Add Models, Controllers and Relationship
   2. Add Routes API / Rest API

3. v.1.3 (June 16, 2025)

   Implementing CRUD API for Sub Tasks.

   1. Add Models, Controllers and Relationship
   2. Add Routes API / Rest API

4. v.1.3 (June 17, 2025)

   Implementing Progress Percentage Calculation for Sub Tasks.

   1. show the percentage change when mark as complete in chart.
   2. shows confetti when all sub tasks completed for interaction to the users.
   3. Shows alert message when all sub tasks complete.