**Project: Task Management Platform**

**Objective:**

Develop a platform where users can register, create, manage, and track tasks. Each user will have a dashboard with their list of tasks and can mark tasks as completed, edit, or delete them.

**Key Features:**

1. **User Authentication (Back-end):**
   * Implement user registration, login, and logout functionalities.
   * Use JSON Web Tokens (JWT) or session-based authentication for secure access to user data.
2. **Task Management (Back-end & Database):**
   * Users can create tasks with details such as title, description, due date, and priority.
   * Store tasks in a database (e.g., MySQL, MongoDB, or PostgreSQL).
   * Create API endpoints for task creation, updating, deleting, and retrieving.
3. **User Dashboard (Front-end):**
   * Each user has their own dashboard where they can view all their tasks in different states (e.g., To Do, In Progress, Completed).
   * Implement filters for tasks (e.g., by priority or due date).
4. **Task Editing (Front-end & Back-end):**
   * Allow users to click on a task to edit its details.
   * Send updates to the back-end to update the database.
5. **Responsive UI (Front-end):**
   * Build the interface with HTML, CSS (or a framework like Bootstrap/Tailwind), and JavaScript (or React/Vue).
   * Make sure the platform is responsive for both desktop and mobile devices.
6. **API Implementation (Back-end):**
   * Create a RESTful API using a back-end framework (e.g., Node.js with Express, Django, or Flask).
   * Handle CRUD (Create, Read, Update, Delete) operations for tasks.

**Technologies to Use:**

1. **Front-End:**
   * **HTML/CSS/JavaScript** (or a framework like React, Angular, Vue)
   * **Bootstrap** or **Tailwind CSS** for styling
   * **Axios** or **Fetch API** for making API requests
2. **Back-End:**
   * **Node.js with Express** (or Django, Flask, or Ruby on Rails)
   * **JWT** for authentication
   * **MongoDB/MySQL/PostgreSQL** for the database
3. **Database:**
   * NoSQL (e.g., MongoDB) or SQL (e.g., MySQL, PostgreSQL)
4. **Optional:**
   * **Deployment**: Host your project using platforms like Heroku or Vercel for front-end, and MongoDB Atlas or DigitalOcean for the database.
   * **Version Control**: Use GitHub or GitLab for managing code.

**Bonus Features (For Extra Challenge):**

* Add task reminders with email notifications using an email API (e.g., SendGrid).
* Implement user roles (e.g., admin, user).
* Add tags or categories for tasks.
* Incorporate a drag-and-drop interface for task management.

This project will give you solid hands-on experience in both front-end and back-end development, helping you integrate the two through APIs and database management.