**Project Idea: TaskFlow - A JIRA Task Management System**

Create a project management tool inspired by Jira, where users can create, manage, and track tasks using a Kanban board layout. This project will help you implement CRUD operations, user authentication, and real-time updates using Node.js and Express.

**Key Features:**

1. **User Authentication**:
   * Secure login and registration using JWT.
   * Password hashing with bcrypt.
2. **Project and Team Management**:
   * Create and manage multiple projects.
   * Assign users to specific projects.
3. **Kanban Board**:
   * Visualize tasks in columns (e.g., "To Do," "In Progress," "Done").
   * Drag-and-drop functionality to move tasks between columns.
4. **Task Management**:
   * Add, edit, delete, and assign tasks to team members.
   * Task details like priority, deadlines, and subtasks.
5. **Real-Time Updates**:
   * Use WebSockets (socket.io) for real-time updates when tasks are updated or moved.
6. **Activity Logs**:
   * Record all changes made to tasks (e.g., status updates, assignments).
7. **Search and Filter**:
   * Search for tasks by title or description.
   * Filter tasks by status, priority, or assigned user.
8. **Optional Features**:
   * Notifications for task updates or deadlines.
   * Integration with calendar tools for deadlines.
   * Export project data as CSV or PDF.

**Tech Stack:**

* **Backend**: Node.js, Express.js.
* **Frontend**: React, Vue.js, or Angular (optional for advanced UI).
* **Database**: MongoDB (NoSQL) or PostgreSQL (Relational).
* **Real-Time Communication**: socket.io.
* **Authentication**: JSON Web Tokens (JWT) and bcrypt for password security.
* **Styling**: Tailwind CSS or Material-UI for responsive design.

**Steps to Build:**

1. **Set Up the Backend**:
   * Use Express.js to create RESTful APIs for task, project, and user management.
   * Implement authentication middleware with JWT.
2. **Database Design**:
   * **User**: id, name, email, password, roles.
   * **Project**: id, name, description, ownerId, teamMembers[].
   * **Task**: id, title, description, status, priority, dueDate, assignedUserId, projectId.
3. **Real-Time Updates**:
   * Integrate WebSockets to push updates for task changes across clients.
4. **Frontend Integration**:
   * Build a Kanban board interface with drag-and-drop (e.g., react-beautiful-dnd).
   * Connect the frontend to the backend APIs for dynamic updates.
5. **Deploy the Application**:
   * Host the backend on platforms like Heroku or AWS.
   * Optionally, host the frontend on Vercel or Netlify