SyncSpace

1. Introduction

SyncSpace is a real-time collaborative platform designed for teams to manage projects, share documents, and communicate within a centralized hub. The goal is to enhance team productivity by integrating key features—Kanban boards, document editing, and chat—into a single, unified application.

2. Functional Requirements

- **FR-1: User & Team Management**:
    - The system must allow secure user registration, login, and logout.
    - It must support role-based access control, with admin and user roles.
    - Admins can create workspaces, while regular users must be invited.
- **FR-2: Workspace & Project Management**:
    - Users can create dedicated workspaces for different projects.
    - Workspaces are accessible only to their members.
    - An admin can generate and share unique invite links for a workspace.
- **FR-3: Dynamic Kanban Boards**:
    - The platform must feature an interactive Kanban board within each workspace.
    - Users can create, edit, and delete tasks.
    - Tasks can be moved between columns (To Do, In Progress, Done) using drag-and-drop.
    - All changes must be updated in **real-time** for all members.
- **FR-4: Real-time Collaboration**:
    - An integrated chat system must enable real-time messaging among workspace members.
    - A collaborative document editor must allow multiple users to edit a single document simultaneously.
- **FR-5: File Sharing**:
    - Users must be able to upload and share files within a workspace.
    - The system should maintain a basic version history for uploaded files.

- **FR-6: Notifications System**:
    - The system should provide real-time alerts for events such as task assignments and mentions in chat.

---

3. Non-Functional Requirements

- **Performance**: The application must be highly responsive. Real-time updates via Socket.IO should occur with minimal latency (ideally under 200ms).
- **Security**: All user data and communications must be secure.
    - User authentication must be handled via **JWT (JSON Web Tokens)**.
    - Passwords must be salted and hashed using **bcrypt**.
    - API routes must be protected using appropriate middleware to prevent unauthorized access.
- **Scalability**: The backend must be designed to handle a growing number of concurrent users and workspaces without a significant drop in performance.
- **Usability**: The user interface (UI) should be intuitive, clean, and responsive on various devices. The user experience (UX) should be seamless for common tasks like creating a task or joining a workspace.

---

4. Technical Stack

- **Frontend**: React.js with react-router-dom for navigation.
- **Backend**: Node.js and Express.js.
- **Database**: MongoDB.
- **Real-time Communication**: Socket.IO.
- **Authentication**: JWT and bcrypt.
- **Styling**: Tailwind CSS for rapid UI development.