Project Management System - Functional Requirement Document

1. Introduction

This document describes the functional requirements for a Project Management System designed to provide users with tools to manage tasks, collaborate in teams, and visualize project progress. Key features include Kanban boards, task management, and user role segregation to protect data privacy.

2. System Overview

The Project Management System will include:

- **User Management and Access Control**: Ensure data security and limit user visibility to their own workspace.
- **Kanban Boards**: Enable project/task segmentation, organization, and status visualization.
- **Responsive Frontend**: Provide a highly interactive and responsive user interface.
- **Data Structure**: Define an efficient and scalable data organization system to handle multiple projects, tasks, and teams.

3. Functional Requirements

3.1 User Management and Access Control

Objective: To ensure data segregation and control user access.

- **User Roles**:
    - **Admin**: Can manage user roles, projects, and have full visibility into their own organization's data.
    - **Member**: Limited to viewing and editing tasks within projects assigned to them.
    - **Viewer**: Read-only access to assigned projects and tasks.
- **Requirements**:
    - Each user should be assigned a unique identifier and role.
    - A user should not be able to access another user's data or workspace.
    - Authentication and session management must be implemented to maintain secure user sessions.
    - Admins should have additional privileges, including managing projects and teams.

3.2 Kanban Boards

Objective: To provide a visual overview of tasks and their progress.

- **Board Structure**:
  - **Project Board**: Each project has its own board, segmented into multiple lanes (e.g., "To Do," "In Progress," "Done").
  - **Task Cards**: Each task is represented as a card with properties like title, description, assignee, priority, and deadline.
- **Requirements**:
  - Users can create, edit, and delete tasks within their assigned projects.
  - Tasks should be draggable between lanes, updating their status.
  - Boards should automatically save changes and update in real time for other users on the project.

3.3 Task Management

Objective: To allow users to add, update, and track tasks within a project.

- **Task Properties**:
  - **Title**
  - **Description**
  - **Assignee**
  - **Due Date**
  - **Priority** (e.g., Low, Medium, High)
  - **Status** (updates when moved between lanes)
- **Requirements**:
  - Users can create and assign tasks within a project.
  - Tasks should support CRUD (Create, Read, Update, Delete) operations.
  - A task's status should update based on lane position in the Kanban board.

3.4 Responsive Frontend Design

Objective: To provide a smooth and interactive UI for better user experience.

- **Requirements**:
  - The frontend must be responsive and adapt to different screen sizes, including desktops, tablets, and mobile devices.
  - Implement dynamic properties to improve interactivity, such as drag-and-drop for task cards.
  - Use React (or similar framework) with state management for seamless data updates and interactions.
  - Ensure accessibility and intuitive navigation.

---

4. Data Structure and Object-Oriented Design

Objective: To structure the data and code in a scalable and maintainable way.

- **Data Structure**:
  - **User**: Contains information such as username, email, password (hashed), role, and assigned projects.
  - **Project**: Includes title, description, and a list of tasks.
  - **Task**: Contains title, description, assignee, due date, priority, and status.
- **Object-Oriented Design**:
  - **Classes**:
    - `User` - manages user details and role-based access.
    - `Project` - encapsulates project-related data and operations.
    - `Task` - encapsulates task-related data and operations.

---

5. Security and Data Privacy

Objective: To maintain data privacy and security in the system.

- **Requirements**:
  - Implement user authentication and authorization.
  - Encrypt sensitive data such as passwords.
  - Ensure data segregation so users can only view and interact with their own projects.

---

6. Future Considerations

As this document is a preliminary outline, some functionalities might need to be simplified for initial implementation. Areas that may require further refinement include:

- **Task Dependency Management**: Ability to set dependencies between tasks.
- **Notification System**: Notify users of updates or assigned tasks.
- **Real-Time Collaboration**: Live updates across users without manual refresh.
- **Project Archiving**: Enable users to archive completed projects.