PROJECT MANAGEMENT SYSTEM

1. INTRODUCTION

1.1 PURPOSE

The purpose of this project is to design and implement a **Project Management System (PMS)** that helps teams efficiently manage projects, tasks, resources, and communication. The system will allow users to create and track projects, assign tasks, collaborate, and monitor progress

1.2 SCOPE

The system will be a **web-based** or **desktop** application that includes the following functionalities:

- User Management (Admins, Managers, Team Members)
- Project Creation, Assignment, and Tracking
- Task Management (Assigning, Updating, and Monitoring)
- Messaging and Notifications
- Reporting and Analytics

2. REQUIREMENTS

2.1 USER MANAGEMENT

User Roles

- Administrator (Manages the system, users, and settings)
- Project Manager (Creates and manages projects, assigns tasks)
- Team Member (Executes tasks assigned to them)

User Features

- Register/Login (Username, Email, Password)
- Role-Based Access Control (RBAC)
- Update Profile & Change Password
- Forgot Password & Reset

`

2.2 PROJECT MANAGEMENT

Project Creation

- Name, Description, Start Date, End Date, Status
- Assign **Project Manager**
- Invite **Team Members**

Project Status Tracking

- Active, On Hold, Completed, Cancelled
- Visual indicators for progress

Project Reports

- Summary of tasks
- View Deadlines
- View Team Members

2.3 TASK MANAGEMENT

Create & Assign Tasks

- Title, Description, Deadline
- Assign to Team Member
- Priority Levels (High, Medium, Low)

Task Tracking & Updates

- Change status: **To Do → In Progress → Completed**
- Add comments & attachments
- Mark dependencies (Task A must be done before Task B)

Notifications for Task Updates

- Team members get alerts when assigned a task
- Alerts for **task deadline approaching**

2.4 COMMUNICATION & COLLABORATION

Messaging System

- **Send messages** between team members
- **Group chats** for specific projects

Project-Based Notifications

- Alerts for **task assignments, project updates, new messages**
- Email and in-app notifications

`

2.5 REPORTING & ANALYTICS

Dashboard with Key Metrics

- Number of **Active/Completed Projects**
- Task Progress (e.g., **80% complete**)

Detailed Reports

- **Performance reports** for team members
- **Task completion reports** for managers

3. NON-FUNCTIONAL REQUIREMENTS

3.1 PERFORMANCE

- **Database queries** should be optimized to prevent delays or timeouts, especially for project/task retrieval.

3.2 SECURITY

- **Role-based access control (RBAC)**
- **Data encryption** for sensitive information
- Secure authentication (e.g., **hashed passwords, 2FA**)

3.3 USABILITY

- Simple & intuitive UI

3.4 RELIABILITY

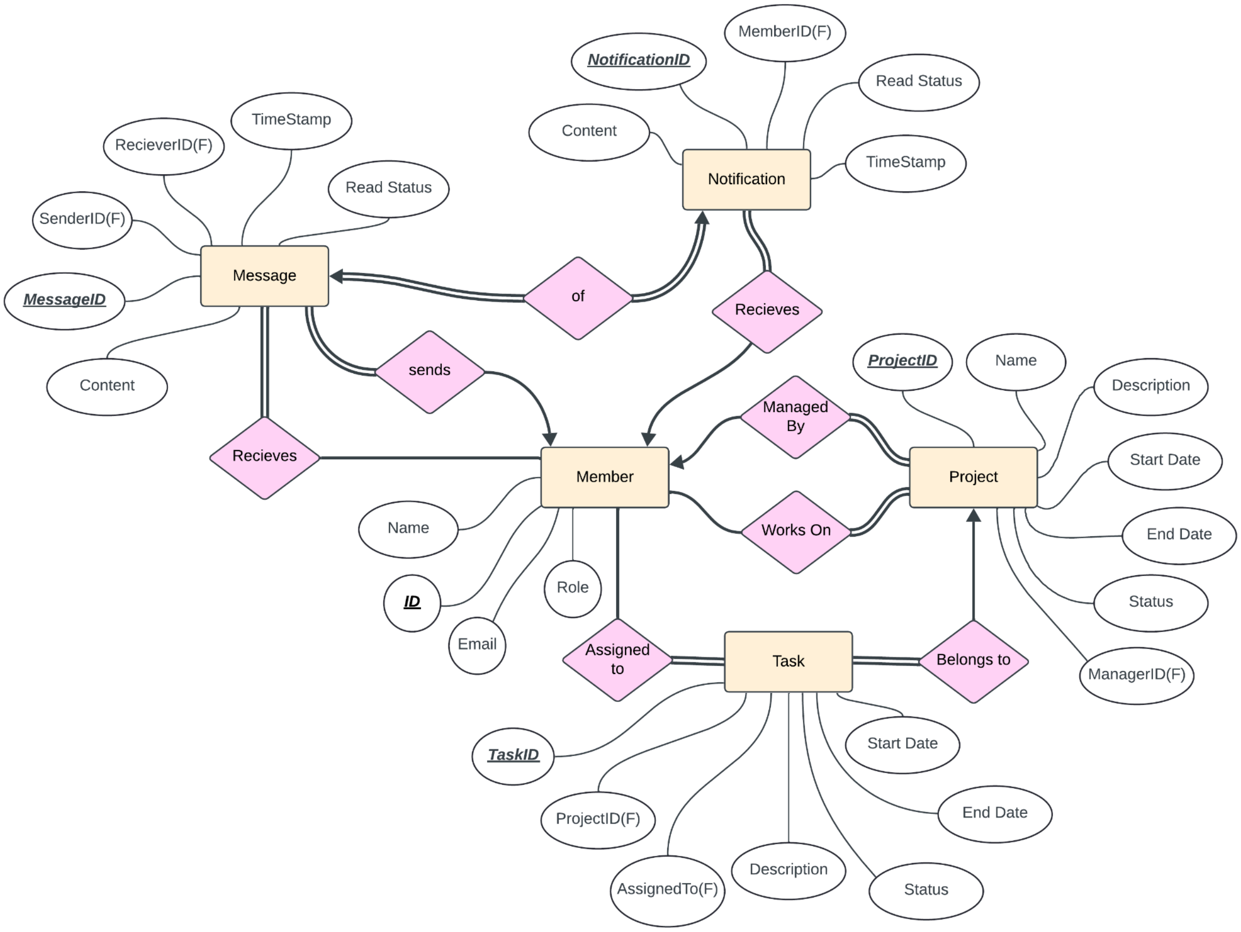
- The system should be **reliable for basic project management use**, without frequent crashes or failures.
- There should be basic **error handling** to inform users when something goes wrong (e.g., task creation failure).

3.5 MAINTAINABILITY

- The system code should be well-organized and documented for easy updates or modifications

`

4. SYSTEM ARCHITECTURE



The ER model represents a **Project Management System** where members manage and collaborate on projects, communicate through messages, and receive notifications. Each **Member** has a unique ID, name, email, and role. A **Member** can send multiple **Messages**, but each message is sent by only one member (**one-to-many** relationship between Member and Message as sender). A member can receive multiple messages and a message can be received by multiple members (**many-to-many** relationship between Member and Message as receiver). Members also receive multiple **Notifications**, while each notification belongs to a single member (**one-to-many** relationship between Member and Notification). A **Project** is uniquely identified by a ProjectID and has attributes like name, description, start and end dates, status, and is managed by a single member, but a member can manage multiple projects (**one-to-many** relationship between Member and Project). Members can work on multiple projects, and a project can have multiple members working on it (**many-to-many** relationship between Member and Project). Each **Task** belongs to a single project, but a project can have multiple tasks (**one-to-many** relationship between Project and Task). A task can be assigned to many members and a member can have multiple tasks (**many-to-many** relationship between Member and Task). The system facilitates structured messaging, automatic notifications, project role management, and task tracking to ensure efficient collaboration and workflow.

`

5. CONCLUSION

This **Project Management System** will streamline **project tracking, team collaboration, and resource allocation**, making it easier for teams to complete projects efficiently.

6. TEAM MEMBERS

NAME	**ROLL NUMBER**	**BRANCH**
Anuraag Tandon	2023110	CSB
Tanmay Aggarwal	2023551	CSB
Jatin Kumar	2023260	CSSS
Dipanshu Parasrampuria	2023204	CSSS

6. Comment Section - Sources and Modifications

The requirements and ER model were developed with the help of **ChatGPT** for structuring and refining explanations. No external websites were referenced. Changes made to the sourced requirements and ER model include modifications to improve scalability, flexibility, and clarity in entity relationships.

`