

Project Management System

A PROJECT REPORT

Submitted by

Vaghela Navin Rameshbhai (21CP049)

In partial fulfillment for the award of the degree of

B. TECH. in COMPUTER ENGINEERING

4CP33: Full Semester External Project (FSEP)



**BIRLA VISHVAKARMA MAHAVIDYALAYA
(ENGINEERING COLLEGE)**

(An Autonomous Institution)

VALLABH VIDYANAGAR

Affiliated to



GUJARAT TECHNOLOGICAL UNIVERSITY, AHMEDABAD

Academic Year: 2024 – 2025

BVM ENGINEERING COLLEGE, VALLABH VIDYANAGAR-388120

APPROVAL SHEET

The project work entitled “**Project Management System**” carried out by **Vaghela Navin Rameshbhai (21CP049)** is approved for the submission in the course **4CP33, Full Semester External Project** for the partial fulfillment for the award of the degree of B. Tech. in Computer Engineering.

Date:

Place:

Signatures of Examiners:

(Names and Affiliations)

CERTIFICATE

This is to certify that Project Work embodied in this project report titled **“Project Management System”** was carried out by **Vaghela Navin Rameshbhai (21CP049)** under the course **4CP33, Full Semester External Project** for the partial fulfillment for the award of the degree of B. Tech. in Computer Engineering. Followings are the supervisors at the student:

Date:

Place:

Satyam Raval
Deputy General Manager
Tech elecon,
Anand

Prof. Bhavesh A Tanawala,
Assistant Professor
Computer Engineering Department,
BVM Engineering College

Dr. Narendra M Patel,
Professor (CAS)
Computer Engineering Department,
BVM Engineering College

Dr. Darshak G Thakore
Professor & Head,
Computer Engineering Department,
BVM Engineering College

COMPUTER ENGINEERING DEPARTMENT, BVM ENGINEERING COLLEGE, VALLABH
VIDYANAGAR – 388120

CERTIFICATE

This is to certify that **Vaghela Navin Rameshbhai (21CP049)** of 4th Level, B.Tech. in Computer Engineering of Birla Vishvakarma Mahavidyalaya, Vallabh Vidyanagar has worked on his project under the course (FSEP, 4CP33) titled **“Project Management system”** under the supervision of **Satyam Raval** at **Tech elecon**. He has done his work sincerely and has completed his project. We found his work satisfactory.

Place:

Date:

Name and Signature of the External Supervisor

Seal of the company

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this report under the course 4CP33 (Full Semester External Project) and that neither any part thereof nor the whole of the report has been submitted for a degree to any other University or Institution.

I certify that, to the best of my knowledge, the current report does not infringe upon anyone's copyright nor does it violate any proprietary rights and that any ideas, techniques, quotations or any other material from the work of other people included in my report, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the boundary of fair dealing within the meaning of the Indian Copyright (Amendment) Act 2012, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in the current report and have included copies of such copyright clearances to the appendix of this report.

I declare that this is a true copy of report, including any final revisions, as approved by the report review committee.

I have checked write-up of the present report using anti-plagiarism database and it is in permissible limit. However, at any time in future, in case of any complaint pertaining of plagiarism, I am the sole responsible person for the same. I also understand that, in case of such complaints of plagiarism, as per the University Grants Commission (UGC) norms, the University can even revoke the degree conferred to the student submitting such a plagiarized report.

Date:

Institute Code: 007

Institute Name: Birla Vishvakarma Mahavidyalaya (BVM) Engineering College

Signature

Vaghela Navin Rameshbhai

21CP049

ACKNOWLEDGEMENT

I am writing to express my sincere gratitude for the opportunity to participate in the React.js internship program. This experience has been incredibly valuable to my professional development, and I am grateful to the many individuals who contributed to its success.

First and foremost, I would like to thank Dr. Indrajit patel, Principal and Professor Dr. Darshak G Thakore Head of the computer engineering Department, for their support and guidance throughout my internship. Their willingness to offer their expertise and provide valuable advice significantly shaped the outcome of my work. I am particularly grateful for their encouragement and assistance during the preparation of this internship report.

I would also like to extend my deepest appreciation to my industry mentor, Mr. Satyam Raval. His guidance, knowledge, and unwavering support throughout the internship were invaluable. Mr. Raval's dedication to sharing his expertise and his willingness to help me overcome challenges significantly contributed to the success of the project.

Furthermore, I am thankful to my fellow team members for their unwavering dedication, collaborative spirit, and hard work. Each team member brought unique skills and perspectives to the table, and their contributions played a crucial role in overcoming challenges and achieving project milestones.

Thank you once again to everyone who provided an opportunity to participate in this internship program. Your support and encouragement have been indispensable, and I am grateful for the opportunity to have learned and grown throughout this experience.

Vaghela Navin Rameshbhai

21CP049

Abstract

In today's fast-paced business environment, efficient project management is essential for organizations to meet deadlines, track progress, and collaborate effectively. A well-structured Project Management System ensures seamless coordination between Project Managers, Team Members, and Administrators, improving productivity and task execution.

Our system leverages React.js for the front-end, Java with Spring Boot for the back-end, and MySQL for database management to provide a responsive, scalable, and user-friendly platform. React.js ensures a dynamic user interface, while Java and MySQL offer a robust backend to handle project data efficiently.

This system allows Project Managers to create and assign tasks, track project progress, and manage teams efficiently, while Team Members can view assigned tasks, update their status, and collaborate effectively. The inclusion of role-based access control ensures that each user has the necessary permissions based on their role.

The objective of this system is to simplify project management, automate task tracking, and enhance team collaboration, making it easier for organizations to plan, execute, and complete projects successfully.

Table of Contents

Section	Page
1. Introduction	1
1.1 Overview	1
1.2 Purpose of the Project	1
1.3 Objectives	2
1.4 Scope of the Project	2
2. Tools and Technologies Used	4
2.1 React.js	4
2.2 HTML	5
2.3 CSS	5
2.4 Material UI	6
2.5 React Router DOM	6
2.6 React Hooks	7
2.7 React DatePicker	7
2.8 React Bootstrap	7
2.9 React Recharts	7
2.10 JavaScript	7
2.11 GitHub	8
3. System Analysis	10
3.1 System Features	10
3.2 Feasibility Study	12
3.3 System Software and Hardware Requirements	14
4. System Design	16
4.1 Use Case Diagram	16
4.2 ER Diagram	17
4.3 Class Diagram	18
4.4 Data Dictionary	20
5. Implementation	25
5.1 Home Page	25
5.2 Login Page	25
5.3 Registration Page	26
5.4 Admin Login Page	27
5.5 Project Manager Dashboard	27
5.6 Profile Page	28
5.7 Project Form Page	29
5.8 Project List Page	29
5.9 Task Form Page	30
5.10 Task List Page	31
5.11 Team Members Page	31
5.12 Team Member Profile Page	32
5.13 Team Member Dashboard	33

Section	Page
5.14 Team Member Task List	33
5.15 Team Member Team Page	34
5.16 Event Form Page	35
5.17 Admin Dashboard	35
5.18 Admin Project Management Page	36
5.19 Admin Task Management Page	37
5.20 Admin User Management Page	38
6. Conclusion	40
7. References	41

LIST OF FIGURES

Figure No.	Title	Page No.
4.1	Use Case Diagram	17
4.2	E-R Diagram	18
4.3	Class Diagram	19
5.1	Home Page	25
5.2	Login Page	26
5.3	Registration Page	26
5.4	Admin Login Page	27
5.5	Project Manager Dashboard Page	28
5.6	Project Manager Profile Page	28
5.7	Project Form Page	29
5.8	Project List Page	30
5.9	Task Form Page	30
5.10	Task List Page	31
5.11	Team Members Page	32
5.12	Team Member's Profile Page	32
5.13	Team Member Dashboard Page	33
5.14	Team Member's Task List Page	34
5.15	Team Member's Team Page	34
5.16	Event Form Page	35
5.17	Admin Dashboard Page	36
5.18	Admin Project Management Page	37
5.19	Admin Task Management Page	38
5.20	Admin User Management Page	39

List of Symbols, Abbreviations and Nomenclature

Term	Description
UI	User Interface
UX	User Experience
API	Application Programming Interface
DB	Database
JWT	JSON Web Token
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
SPA	Single Page Application
CRUD	Create, Read, Update, Delete
SQL	Structured Query Language
ER Diagram	Entity Relationship Diagram
PM	Project Manager
TM	Team Member
ID	Identifier

1. Introduction

1.1 Overview

The **Project Management System** is designed to enhance the efficiency of organizations in managing their projects, tasks, and team collaborations. This system operates on a role-based structure, categorizing users into three primary roles:

- **Admin** – Responsible for managing user accounts, system settings, and overall system health.
- **Project Manager** – Tasked with creating and assigning projects, overseeing task completion, and managing team members.
- **Team Member** – Assigned specific tasks, which they can view, update, and collaborate on with other team members.

This system addresses the challenges associated with manual project tracking, such as miscommunication, task mismanagement, and project delays. By offering a centralized platform, users can log in, access project details, track progress, and ensure that all tasks are completed within the assigned timeframes.

A secure authentication system ensures that only authorized individuals can access specific functionalities based on their role, safeguarding data integrity. The system is designed to improve work efficiency through a structured, user-friendly workflow.

1.2 Purpose of the Project

The **Project Management System** aims to streamline the management of projects, tasks, and team collaboration for organizations. By offering a single platform to assign, track, and manage projects, the system simplifies the process of organizing and overseeing tasks.

For **Project Managers**, it provides the ability to assign tasks, set deadlines, and monitor progress, ensuring that projects stay on track.

The system offers a **role-based structure**, where **Admins** manage the system, **Project Managers** oversee the projects, and **Team Members** are responsible for task completion. This clear division of responsibilities minimizes confusion and optimizes work efficiency.

With this system, organizations can enhance productivity, improve communication among teams, and ensure that projects are completed on time and within scope. By eliminating manual tracking and fragmented communication channels, the system makes project management more streamlined and effective.

1.3 Objectives

- **User Role Management**
 - Allow users to register and log in based on their roles (Admin, Project Manager, and Team Member).
 - Provide secure authentication to prevent unauthorized access.
 - Ensure each user has specific permissions based on their role.
- **Project Management**
 - Enable **Project Managers** to create, edit, and delete projects.
 - Provide clear project overviews, including deadlines, progress tracking, and assigned tasks.
 - Offer easy search and filtering options for efficient management of multiple projects.
- **Task Assignment and Tracking**
 - Allow **Project Managers** to assign tasks to **Team Members** and set deadlines.
 - Enable **Team Members** to update the status of tasks (e.g., In Progress, Completed, Pending).
 - Provide task filtering options based on status, priority, or project.
- **User-Friendly Interface**
 - Ensure a clean, intuitive, and simple user interface.
 - Provide clear navigation for each role to access their respective features.
 - Implement notifications to keep users updated on task progress and deadlines.
- **Secure System and Data Management**
 - Store all project and task data securely in a backend database.
 - Restrict access to authorized users only, ensuring data security.
 - Maintain data consistency and integrity across all user roles and modules.
- **Collaboration and Team Communication**
 - Enable **Team Members** to collaborate on tasks and see who is responsible for what.
 - Allow real-time updates to tasks, improving collaboration.
 - Enhance teamwork by ensuring each member knows their responsibilities within the project.

1.4 Scope of the Project

The **scope** of the **Project Management System** encompasses its functionalities, features, and limitations. It is designed for small to medium-sized teams and organizations seeking to improve the planning, execution, and tracking of their projects.

1.4.1 Features and Functionalities

- **Admin Role:**
 - Manage user accounts (Project Managers and Team Members).
 - Monitor system usage and ensure smooth operations.
- **Project Manager Role:**

- Create, edit, and manage projects.
 - Assign tasks to **Team Members** and set deadlines.
 - View and manage team members involved in a project.
- **Team Member Role:**
 - View assigned tasks and update their status.
 - Collaborate with other team members for task completion.
 - Access team member profiles to understand project responsibilities.

1.4.2 Backend and Database Management

- The system will store all **user**, **project**, and **task** data securely.
- Real-time data updates will ensure accuracy and reliability.
- Role-based authentication ensures privacy and security of user data.

2. Tools And Technology

2.1 React.js :

React.js is a JavaScript library for building user interfaces, primarily for web applications. It enables developers to create interactive UI components efficiently by using a component-based architecture. Here's a breakdown:

- **Component-Based:** React breaks down the UI into reusable components, each responsible for rendering a small, self-contained part of the UI. Components can be nested within each other, allowing for complex UI structures.
- **Virtual DOM:** React uses a virtual DOM to improve performance. Instead of directly manipulating the browser's DOM, React creates a lightweight virtual representation of the DOM in memory. When the state of a component changes, React compares the virtual DOM with the real DOM and only updates the parts that have changed, minimizing DOM manipulation and increasing performance.
- **JSX:** JSX is a syntax extension for JavaScript that allows you to write HTML-like code within JavaScript. It simplifies the process of creating React elements, making the code more readable and expressive.
- **Unidirectional Data Flow:** React follows a unidirectional data flow, meaning data flows in one direction from parent to child components. This makes it easier to understand how data changes over time and debug applications.
- **State Management:** React components can have state, which represents the data that change over time. When the state of a component changes, React automatically re-renders the component, updating the UI to reflect the new state.
- **Lifecycle Methods:** React components have lifecycle methods that allow developers to hook into various points in a component's lifecycle, such as when it is first mounted or updated. This enables developers to perform actions like fetching data or cleaning up resources at the appropriate times.
- **Declarative Syntax:** React uses a declarative syntax, where you describe what you want the UI to look like, and React takes care of updating the DOM to match that description. This makes it easier to reason about your code and build complex UIs.

2.2 HTML (HyperText Markup Language):

HTML is the standard markup language used to create the structure and content of web pages. It consists of a series of elements, represented by tags, which define the different parts of a web page such as headings, paragraphs, images, links, and more. HTML elements are organized in a hierarchical structure, with nested elements representing the relationship between different parts of the content. HTML provides semantic meaning to the content, making it accessible to both users and search engines.

2.3 CSS:

CSS is a style sheet language used to control the presentation and layout of HTML elements on a web page. It allows developers to define styles such as colors, fonts, spacing, and positioning, making it possible to create visually appealing and responsive designs. CSS operates by selecting HTML elements and applying styles to them using selectors and declaration blocks. CSS can be applied inline within HTML documents, embedded within `<style>` tags in the document's `<head>` section, or linked externally to the HTML document as a separate stylesheet.

2.4 Material Ui:

Material-UI is a popular React component library that implements Google's Material Design guidelines. It provides pre-designed and customizable UI components, such as buttons, cards, menus, forms, and more, to help developers build modern and visually appealing web applications with ease. Material-UI components are built with React, making them easy to integrate into React projects and leverage React's component-based architecture. Additionally, Material-UI offers extensive theming capabilities, allowing developers to customize the look and feel of their applications to match their brand or design preferences. With its rich set of components, thorough documentation, and active community support, Material-UI simplifies the process of creating responsive and intuitive user interfaces for React-based web applications.

2.5 React-Router-DOM:

React Router DOM is a popular routing library for React applications, enabling developers to handle navigation and routing within single-page applications (SPAs). It allows you to define different routes in your application, each corresponding to a specific URL path, and render different components based on the current URL. React Router DOM provides a `<BrowserRouter>` component to manage browser history using HTML5 history API, allowing for navigation without full page reloads. It also offers various route components like `<Route>`, `<Switch>`, `<Redirect>`, and `<Link>` to define route matching, switch between routes, redirect users, and create navigation links, respectively. With React Router DOM, developers can create dynamic and interactive web applications with multiple views, enabling seamless navigation between different pages while maintaining a single-page experience.

2.6 React Hooks:

React Hooks allow functional components to use state and other React features without needing class components. They include `useState` for state management, `useEffect` for side effects, `useContext` for accessing context, `useReducer` for more complex state logic, `useCallback` and `useMemo` for performance optimization, and `useRef` for creating mutable references. Hooks provide a more concise and readable way to manage state and side effects in React applications, promoting cleaner and more maintainable code.

2.7 React-Date Picker:

React-DatePicker is a flexible and customizable date picker component for React applications, widely used for selecting dates and times within forms or other user interfaces. It is designed to be user-friendly and offers a range of features to enhance date selection, such as customizable date and time formats, support for date range selection, and localization options. This makes it ideal for applications requiring precise date inputs, like booking systems or scheduling applications. React-DatePicker integrates seamlessly with form libraries and state management tools, making it easy to manage form data and validations. Additionally, it offers extensive theming options to ensure the date picker matches the look and feel of the application.

2.8 React-Bootstrap:

React-Bootstrap is a popular library that brings the power of Bootstrap components to React applications, providing a comprehensive set of UI components that adhere to the Bootstrap design system. This ensures consistency and responsiveness across different devices and screen sizes. React-Bootstrap includes a wide range of pre-built components such as buttons, modals, forms, and navigation bars, which can be easily customized and used in projects. Its integration with Bootstrap makes building responsive and mobile-friendly user interfaces straightforward. The library also allows for extensive customization of component styles to match the application's branding, ensuring a visually appealing and accessible user experience.

2.9 React-Recharts:

React-Recharts is a composable charting library built on React components, designed to be simple to use, highly customizable, and performant. It supports various types of charts, including line charts, bar charts, pie charts, and scatter plots, making it ideal for creating data visualizations in React applications. React-Recharts offers extensive customization options for each chart type, including colors, legends, tooltips, and animations. Its composability allows for the combination of different chart components to create complex and interactive data visualizations. The library ensures that charts are responsive and adapt to different screen sizes and devices, making it suitable for displaying business metrics, scientific data, and interactive dashboards.

2.10 Javascript:

JavaScript is a high-level, interpreted programming language primarily used for web development.

- **Dynamic:** JavaScript is dynamic, meaning it can adapt and change as the program runs. This makes it well-suited for creating interactive web pages.
- **Client-Side Scripting:** JavaScript is mainly used as a client-side scripting language in web browsers, allowing developers to create dynamic and interactive web pages by

manipulating the HTML and CSS content.

- **Event-Driven:** JavaScript is event-driven, meaning it can respond to user actions such as mouse clicks, keyboard inputs, and form submissions. This enables developers to create responsive and interactive user interfaces.
- **Functional and Object-Oriented:** JavaScript supports both functional and object-oriented programming paradigms, allowing developers to write code in a variety of styles. It also features first-class functions, closures, and prototypes, which contribute to its flexibility and expressiveness.
- **Cross-Platform:** JavaScript is supported by all major web browsers and can run on various platforms, including desktops, mobile devices, and servers. This makes it a versatile language for building a wide range of applications.
- **Libraries and Frameworks:** JavaScript has a rich ecosystem of libraries and frameworks, such as React, Angular, and Vue.js, which streamline the development process and provide additional features and functionalities for building web applications.

2.11 GitHub:

GitHub is a web-based platform and version control system that enables developers to collaborate on projects, host code repositories, and manage software development workflows. Below is a brief overview of its core features:

- **Version Control:** GitHub uses Git, a distributed version control system, to track changes to files and manage different versions of a codebase. Developers can create branches to work on features or bug fixes independently, merge changes into the main branch, and revert to earlier versions when necessary.
- **Code Hosting:** GitHub offers a centralized platform for hosting Git repositories, making it easy to share code. Repositories can be public or private, and GitHub provides features such as issue tracking, project boards, and wikis to support collaboration and project management.
- **Collaboration:** GitHub supports collaborative development by allowing users to fork repositories, make changes, and submit pull requests for review and integration. De-

developers can discuss code changes, review proposed modifications, and provide feedback through comments and code reviews.

- **Community and Open Source:** GitHub is a central hub for the open-source community, enabling contributions to public projects, discovery of new tools and frameworks, and interaction with other developers. It hosts millions of public repositories across diverse fields such as web development, data science, machine learning, and gaming.

3. System Analysis

3.1 System Features

3.1.1 Front-End Development

- **Responsive Design:**

- The system will be built using React.js, ensuring it works smoothly on desktops, tablets, and mobile devices.
- The UI will automatically adjust to different screen sizes for a consistent and user-friendly experience.

3.1.2 Dynamic Content and Interactions

- **Real-Time Updates:**

- Users will get instant notifications and updates about projects, tasks, and team activities.
- The system ensures that all changes, like adding or updating tasks, reflect in real time.

- **Interactive Features:**

- Buttons, pop-ups, and animations will improve user interaction and navigation.
- Users can easily add, edit, and manage projects, tasks, and members with interactive elements.

3.1.3 Form Management and Validation

- **Easy-to-Use Forms:**

- Project Managers can create projects, assign tasks, and update statuses using simple forms.
- Team Members can update their task progress effortlessly.

- **Secure Login and Signup:**

- The system will include a secure authentication process for login and registration.

- Passwords will be hashed and stored securely in the MySQL database.

- **Error Checking:**

- Forms will automatically validate user inputs, checking for missing fields or incorrect data.
- Instant feedback will help users correct errors before submission.

3.1.4 Personalization and User Experience

- **Custom Dashboards:**

- Each user will have a personalized dashboard showing their assigned tasks, project status, and team members.

- **Role-Based Access Control:**

- **Admins:** Manage user accounts and system settings.
- **Project Managers:** Oversee projects, assign tasks, and manage team members.
- **Team Members:** View and update their tasks.

- **Smooth and Responsive UI:**

- Fast navigation, clean design, and easy accessibility for all users.

3.1.5 Single-Page Application (SPA) Approach

- **Fast and Smooth Navigation:**

- The system is built as a Single-Page Application (SPA) using React.js.
- This ensures quick loading and seamless transitions between different sections without full page reloads.

3.2 Feasibility Study

3.2.1 Technical Feasibility

- **TechnologyStack:**

The system uses React.js for the front end to create a smooth and interactive user experience. The backend is developed using Java (Spring Boot) to handle server-side logic efficiently. MySQL is used for securely storing and managing project, task, and user data. JWT (JSON Web Token) is implemented for secure user authentication and session management.

- **Infrastructure:**

The system is hosted on a cloud-based platform like AWS or DigitalOcean, ensuring scalability and reliability. The MySQL database is structured to handle large volumes of data efficiently, ensuring smooth performance. The backend server using Java (Spring Boot) ensures a fast and stable connection between users and the database.

- **DevelopmentResources:**

React.js, Java (Spring Boot), and MySQL are widely used technologies, making it easier to find developers for future updates. The component-based structure of React.js simplifies maintenance and feature upgrades. Java provides a robust backend solution, ensuring the system runs smoothly even with multiple users.

3.2.2 Operational Feasibility

- **User Accessibility:**

The Project Management System is a web-based platform, accessible from any device with an internet connection, including desktops, laptops, tablets, and mobile phones. The responsive design ensures a smooth user experience across all screen sizes. Role-based access ensures that **Admins, Project Managers, and Team Members** only see the features relevant to them.

- **Maintenance:**

The system is built using a component-based architecture in React.js, making it easy to update and maintain. Backend updates in Java and database modifications

in MySQL can be done without affecting the entire system. Regular updates will ensure bug fixes, performance improvements, and security enhancements.

- **Training and Support:**

The system is designed to be user-friendly, minimizing the need for extensive training. However, user guides, tutorials, and customer support will be available to help users navigate and effectively use the platform. Future updates can include an in-app help section or chatbot for real-time assistance.

3.2.3 Economic Feasibility

- **Development**

The Project Management System is built using open-source technologies like React.js, Java (Spring Boot), and MySQL, which reduces development costs. Initial expenses include server hosting, domain registration, and developer resources, but these are manageable with efficient planning.

- **Operational**

Since the system is web-based, there is no need for expensive hardware. Cloud-based hosting services like AWS, DigitalOcean, or Firebase ensure cost-effective scalability. Regular maintenance, security updates, and minor improvements will have minimal ongoing costs.

- **Long-Term**

The system will help increase productivity, reduce miscommunication, and improve project tracking, leading to cost savings in management and efficiency. The automation of task assignments and tracking reduces manual effort, making project execution smoother and reducing operational overhead.

3.3 System Software and Hardware Requirements for Web App

3.3.1 Development Environment

- **Operating System:**
 - Recommended: Windows 10 or 11, macOS 10.10 or later, Ubuntu 16 or later (for best compatibility and performance)
 - Minimum: Most modern operating systems with a compatible web browser should work (may require additional configuration)
- **Hardware:**
 - Minimum:
 - Processor: Dual-core processor (e.g., Intel Pentium 4 or equivalent)
 - RAM: 4 GB
 - Storage: 10 GB free disk space
 - Recommended:
 - Processor: Quad-core processor (e.g., Intel Core i5 or equivalent)
 - RAM: 8 GB or more
 - Storage: 50 GB or more (depending on project complexity and additional tools)
- **Software:**
 - **Backend:** Java for handling business logic.
 - **Frontend:** React.js for a responsive user interface.
 - **Database:** MySQL for storing and managing project, task, and user data.
 - **Development Tools:** Visual Studio Code, IntelliJ IDEA, or Apache netbea for coding.
 - **Package Managers:** npm (for React.js) and Glassfish server (for Java).
 - **Version Control:** Git/GitHub for code collaboration and tracking.

3.3.2 Deployment Environment

The deployment environment (where the React application runs after development) will vary depending on the chosen hosting provider. However, some general requirements include:

- **Web server:** A web server that can serve static files and handle backend functionality if needed (e.g., Apache, Nginx)
- **Node.js runtime (optional):** If the application requires server-side rendering or other Node.js functionality

4 System Design

4.1 Use Case Diagram:

Use Case Diagram represents the main functionalities of the Project Management System and shows the interactions between different types of users (Admin, Project Manager, Team Member) and the system.

In the diagram:

- **Admin** can manage users, view system analytics, and oversee project and task management.
- **Project Manager** can create projects, assign tasks, manage team members, and track project progress.
- **Team Member** can view assigned tasks, update task statuses, and collaborate with other members.

The diagram helps in identifying all the system requirements from a user's point of view and clearly outlines the various services the system provides to its users.

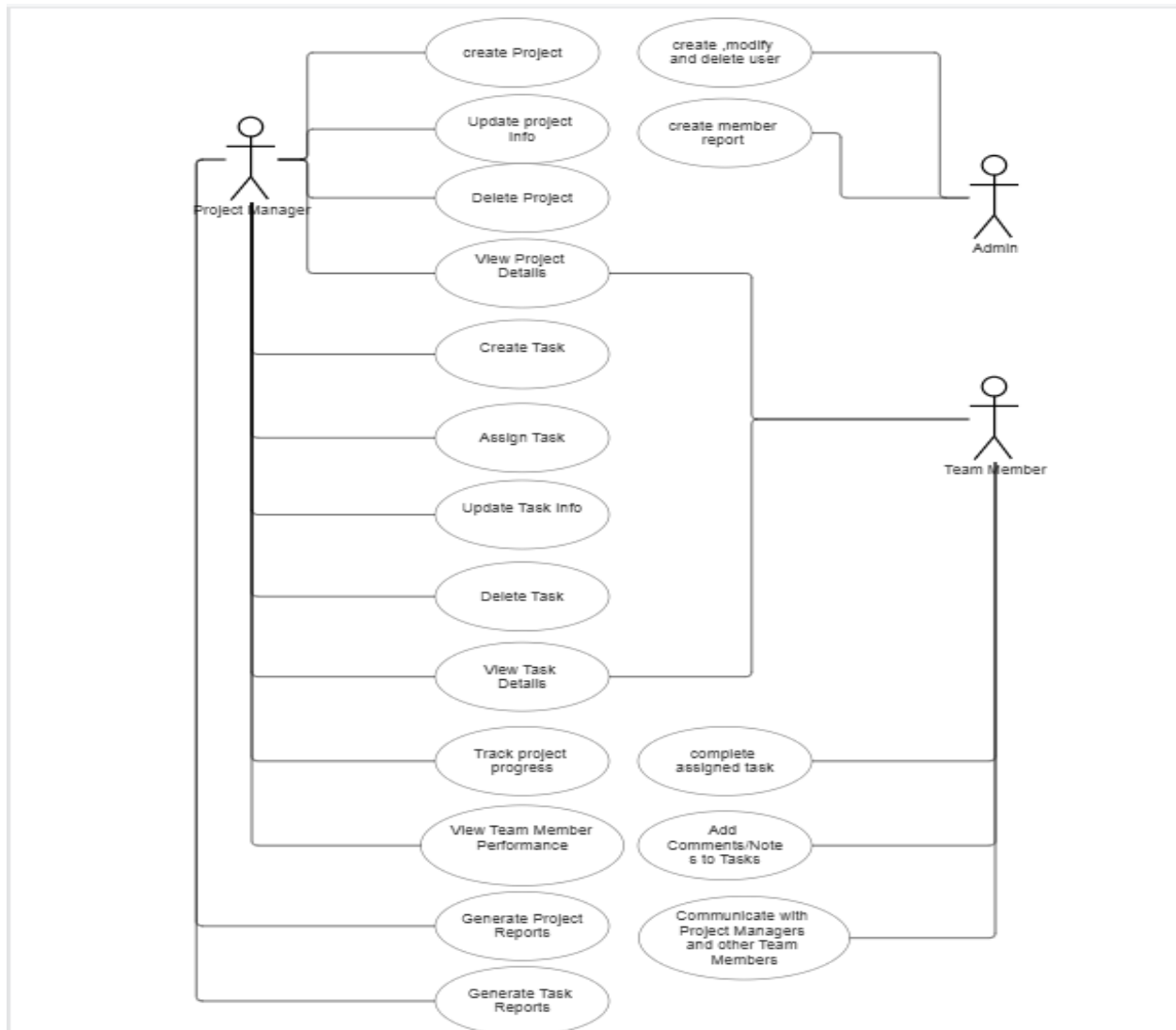


Figure 4.1 Use Case Diagram

4.2 ER Diagram :

The Entity-Relationship (ER) Diagram represents the core structure of the Project Management System, showing the relationships between users, projects, tasks, files, comments, and events.

- **Users** can create projects, upload files, post comments, and create events.
- **Projects** are created by users and have multiple tasks, files, events, and comments associated with them.
- **Tasks** are assigned to users and linked to specific projects.
- **Files** are uploaded by users and linked to projects.
- **Comments** are posted by users on projects.
- **Events** are created by users for particular projects.

- **Project Members** table manages the assignment of users to projects.

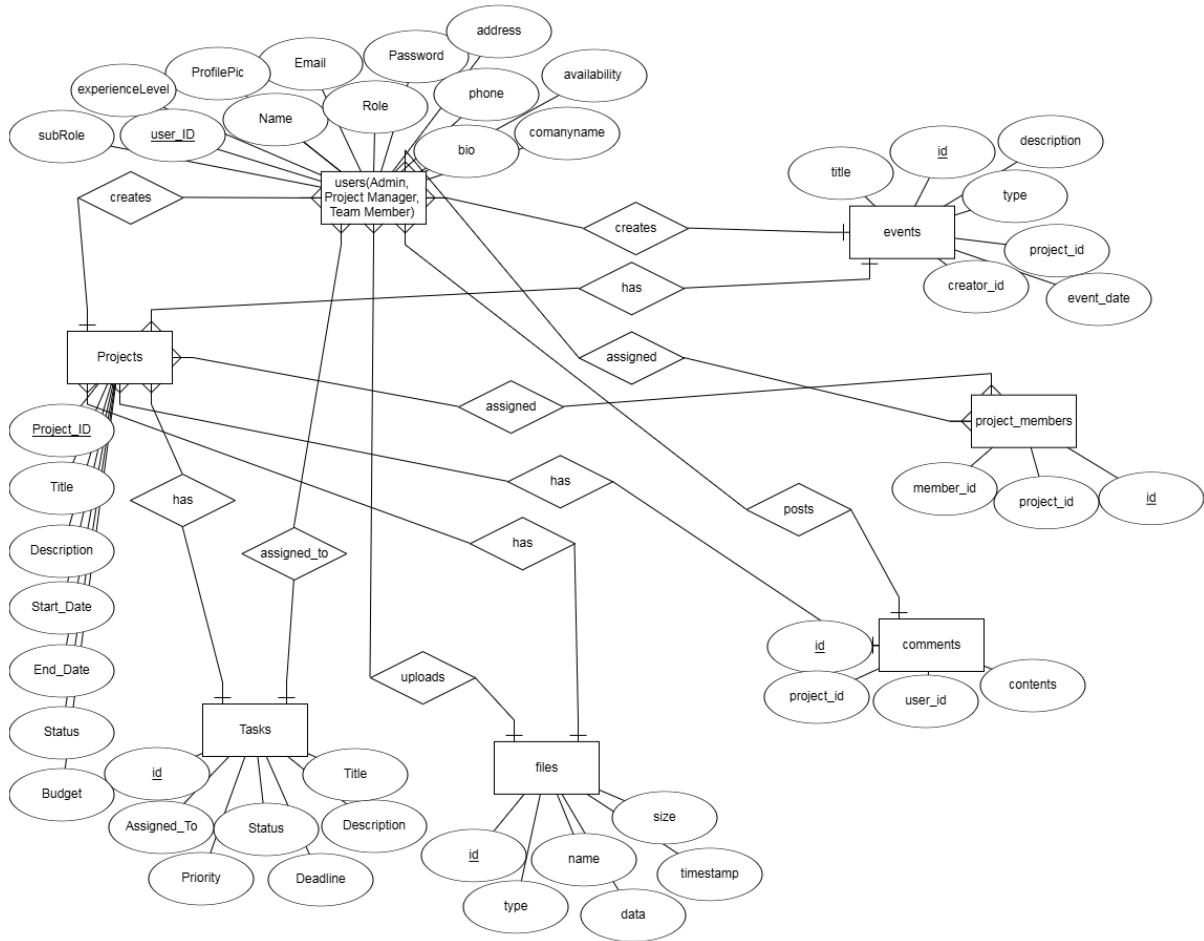


Figure 4.2 E-R Diagram

4.3 Class Diagram:

The Class Diagram shows the main entities of the Project Management System and their relationships.

- **Users** can create projects, tasks, comments, upload files, and participate in events.
- **Projects** are created by users and contain tasks, events, files, and comments.
- **Tasks** are assigned to users and linked to projects.
- **Team Members** link users to projects, managing team assignments.
- **Events** are project-related activities created by users.
- **Files** are uploaded by users for specific projects.

- **Comments** allow users to communicate within projects.
- **Admin** users manage the system separately.

This design ensures smooth project management, user collaboration, and secure data handling.

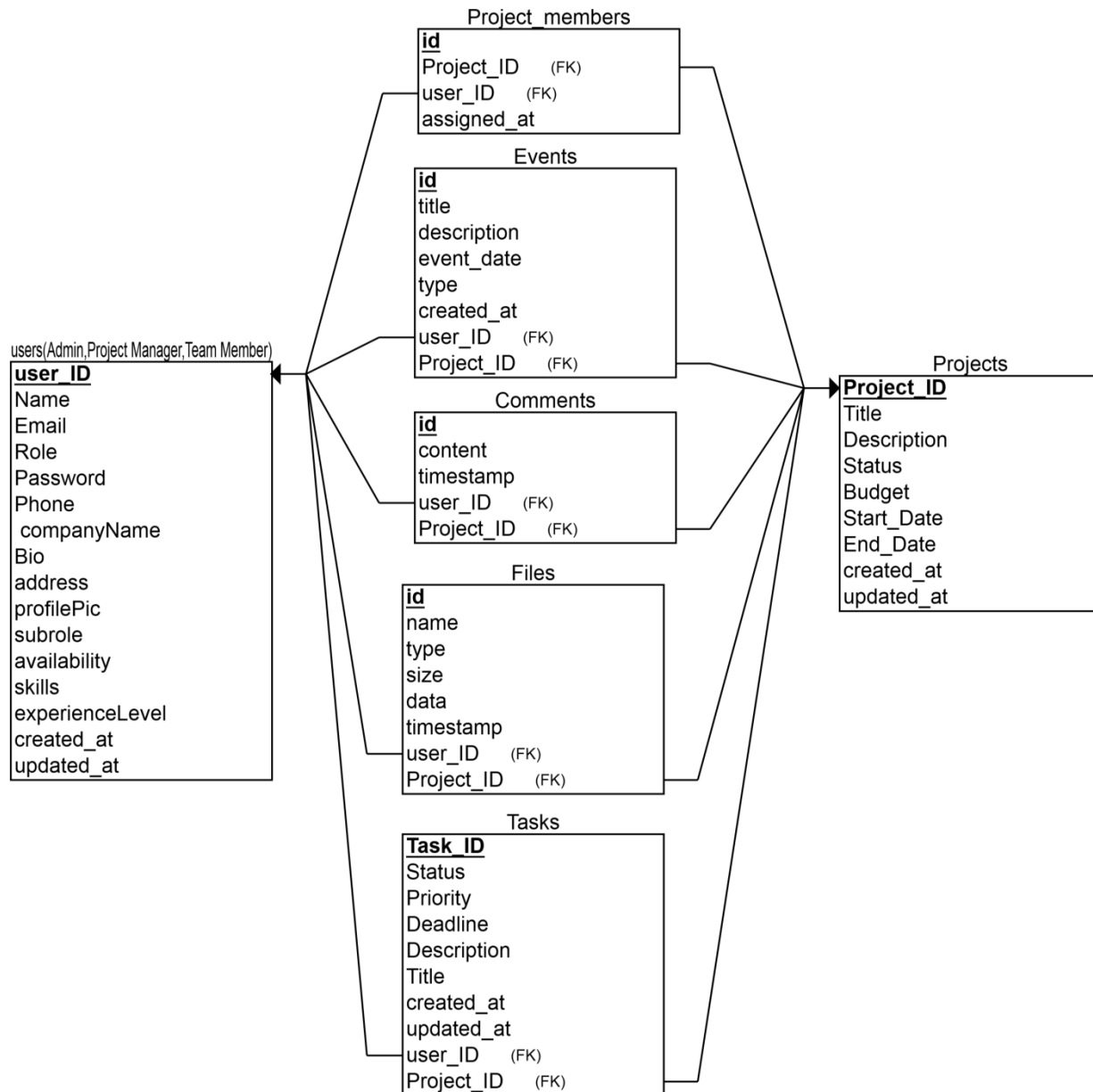


Figure 4.3 Class Diagram

4.4 Data Dictionary:

4.4.1 User Table

Field Name	Data Type	Constraints	Description
id	INT	Primary Key, Auto-Increment	Unique identifier for each user
name	VARCHAR(255)	NOT NULL	Full name of the user
email	VARCHAR(255)	NOT NULL, UNIQUE	User's email address
password	VARCHAR(255)	NOT NULL	Encrypted user password
role	VARCHAR(50)	NOT NULL	User's role (e.g., sponsor, influencer, admin)
phone	VARCHAR(20)	NULL	Phone number of the user
company	VARCHAR(255)	NULL	Company name (for sponsors)
address	TEXT	NULL	Physical address of the user
bio	TEXT	NULL	Short biography or profile description
profilePic	VARCHAR(255)	NULL	File path or URL to user's profile picture
subrole	VARCHAR(100)	NULL	Sub-category or specialization of the user
availability	VARCHAR(100)	NULL	Availability status (e.g., available, busy)
skills	TEXT	NULL	User's skills or areas of expertise
experienceLevel	VARCHAR(100)	NULL	Experience level (e.g., Beginner, Intermediate, Expert)
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Date and time when the account was created
updated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	Date and time of the last update to the account

4.4.2 Projects table

Field Name	Data Type	Constraints	Description
Project_ID	INT	Primary Key, Auto-Increment	Unique identifier for each project
Title	VARCHAR(255)	NOT NULL	Title or name of the project
Description	TEXT	NULL	Detailed description of the project
Start_Date	DATE	NULL	Project start date
End_Date	DATE	NULL	Project end date
Budget	DECIMAL(10,2)	NULL	Budget allocated for the project
userId	INT	Foreign Key (users.id)	ID of the user who created or owns the project
Status	VARCHAR(100)	NULL	Current status (e.g., active, completed, pending)
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Date and time when the project was created
updated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	Date and time of the last update to the project

4.4.3 tasks table

Field Name	Data Type	Constraints	Description
Id	INT	Primary Key, Auto-Increment	Unique identifier for each task
Title	VARCHAR(255)	NOT NULL	Title or name of the task
description	TEXT	NULL	Detailed description of the task
Status	VARCHAR(50)	NOT NULL	Current status (e.g., pending, in progress, completed)
deadline	DATE	NULL	Task deadline date
assigned_to	INT	Foreign Key (users.id)	User ID to whom the task is assigned
Priority	VARCHAR(50)	NULL	Task priority level (e.g., low, medium, high)
project_id	VARCHAR(255)	Foreign Key (projects. Project_ID)	Id of the related project
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Date and time when the task was created
updated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	Date and time of the last update to the task

4.4.4 project_members table

Field Name	Data Type	Constraints	Description
Id	INT	Primary Key, Auto-Increment	Unique identifier for each project-member assignment
project_id	INT	Foreign Key (projects.Project_ID), NOT NULL	ID of the project the member is assigned to
member_id	INT	Foreign Key (users.id), NOT NULL	ID of the user assigned to the project
assigned_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Date and time when the member was assigned to project

4.4.5 events table

Field Name	Data Type	Constraints	Description
Id	INT	Primary Key, Auto-Increment	Unique identifier for each event
Title	VARCHAR(255)	NOT NULL	Title or name of the event
description	TEXT	NULL	Detailed description of the event
event_date	DATE	NOT NULL	Scheduled date of the event
Type	VARCHAR(100)	NULL	Type of event (e.g., meeting, deadline, reminder)
creator_id	INT	Foreign Key (users.id)	ID of the user who created the event
project_id	INT	Foreign Key (projects.Project_ID)	ID of the project associated with the event (if applicable)
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Date and time when the event was created

4.4.6 Comments table

Field Name	Data Type	Constraints	Description
Id	INT	Primary Key, Auto-Increment	Unique identifier for each comment
project_id	INT	Foreign Key (projects.id), NOT NULL	ID of the project the comment is associated with
user_id	INT	Foreign Key (users.id), NOT NULL	ID of the user who posted the comment
content	TEXT	NOT NULL	The actual content or message of the comment
timestamp	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Date and time when the comment was posted

4.4.7 files table

Field Name	Data Type	Constraints	Description
Id	INT	Primary Key, Auto-Increment	Unique identifier for each file
Name	VARCHAR(255)	NULL	Name of the uploaded file
Type	VARCHAR(100)	NULL	MIME type or file extension (e.g., image/png, pdf)
Size	BIGINT(20)	NULL	Size of the file in bytes
Data	LONGBLOB	NULL	Binary data of the file
uploadedBy	INT	Foreign Key (users.id), NULL	ID of the user who uploaded the file
projectId	INT	Foreign Key (projects.id), NOT NULL	ID of the associated project
timestamp	DATETIME	NULL	Date and time when the file was uploaded

5. Implementation

5.1 Home Page:

The **Home Page** serves as the initial entry point for users. It introduces the platform and offers easy navigation to the **Login Page**, **Admin Page**, and additional platform features. The page provides an overview of the system's functionalities, ensuring that users can quickly access the services they need.

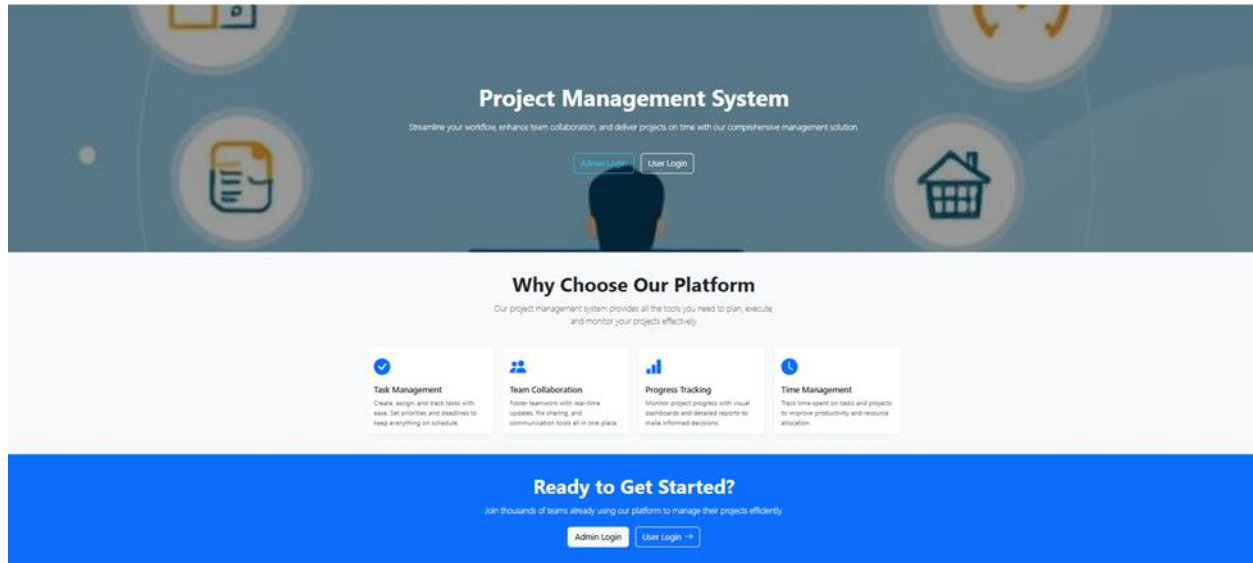


Figure 5.1 Home Page

5.2 Login Page:

The Login Page allows users to access the system by entering their credentials. Users will enter their registered email and password to log in. If the login credentials are correct, they will be redirected to their respective dashboards based on their roles (Admin, Project Manager, or Team Member). This page has a clean, user-friendly design with clear fields for email and password, ensuring an easy and secure login experience.

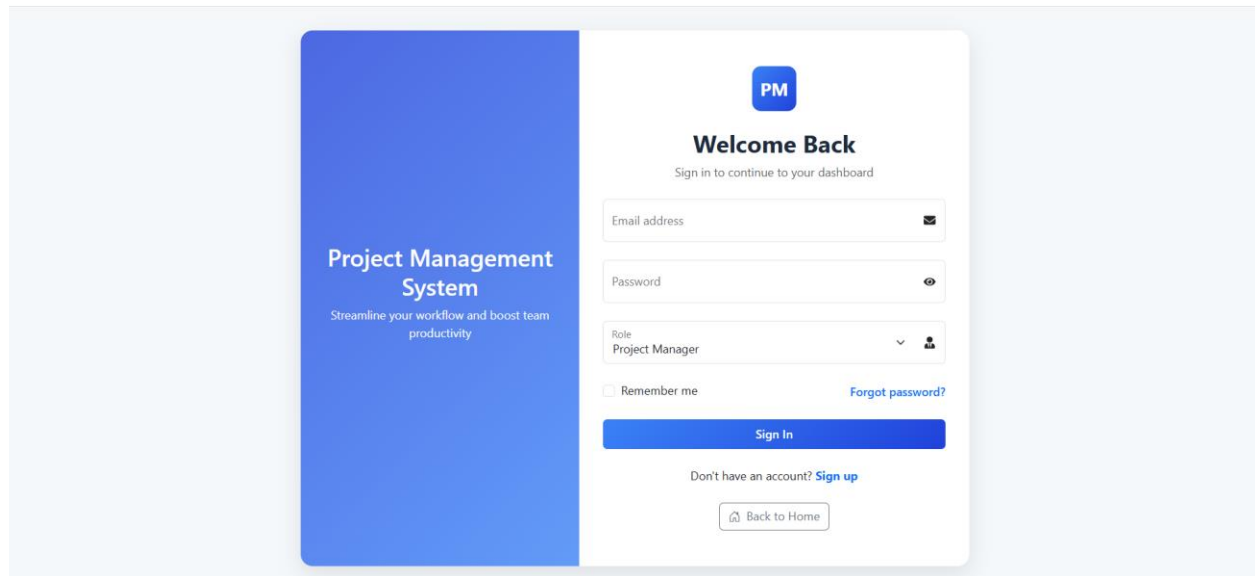


Figure 5.2 Login Page

5.3 Registration page:

The Registration Page allows new users to create an account. Users will be asked to provide basic information such as name, email, password, and role (Project Manager, or Team Member). Once the registration form is completed, users are redirected to a profile page where they can enter more detailed information (e.g., job title, team role). This process ensures that each user has a complete profile upon registration.

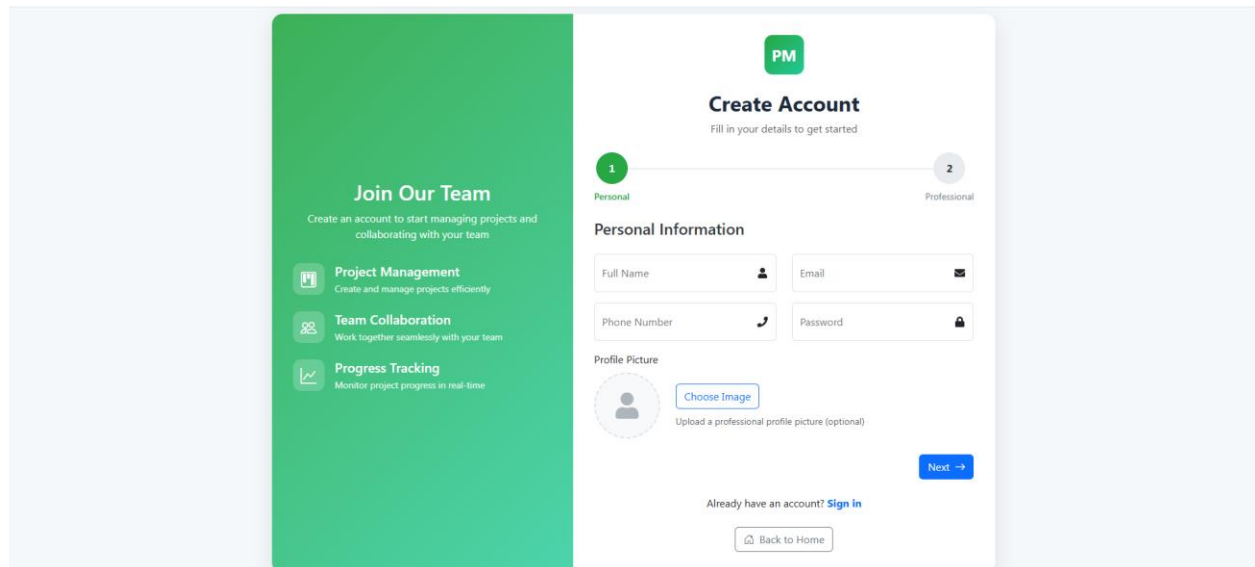


Figure 5.3 Registration Page

5.4 Admin Login page:

The **Admin Login Page** is specifically designed for administrators to access the backend of the platform. The page includes a login form where admins can enter their credentials (email and password). Once logged in, admins gain access to powerful tools to manage user accounts, monitor system performance, and review analytics. The design of this page is simple and secure, focusing on easy access to admin-specific features.

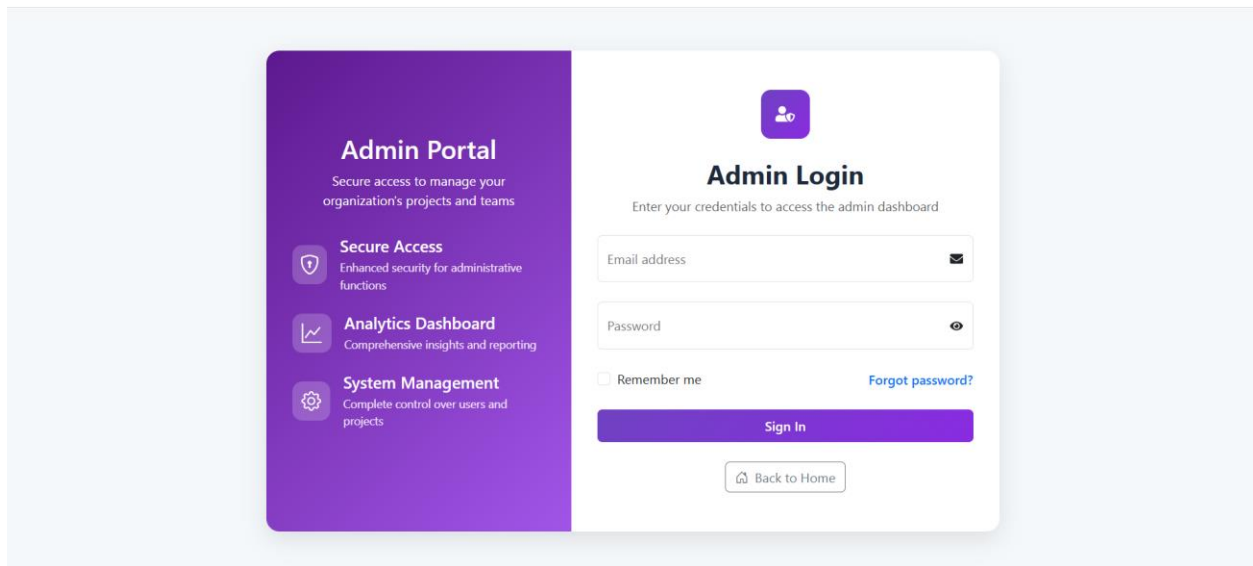


Figure 5.4 Admin Login Page

5.5 Project manager dashboard:

The Project Manager Dashboard serves as the central hub for Project Managers. On this page, Project Managers can create, edit, or delete projects, assign tasks to team members, and monitor the progress of their projects. The dashboard provides real-time updates on the status of various projects, with easy-to-use navigation for managing multiple projects at once.

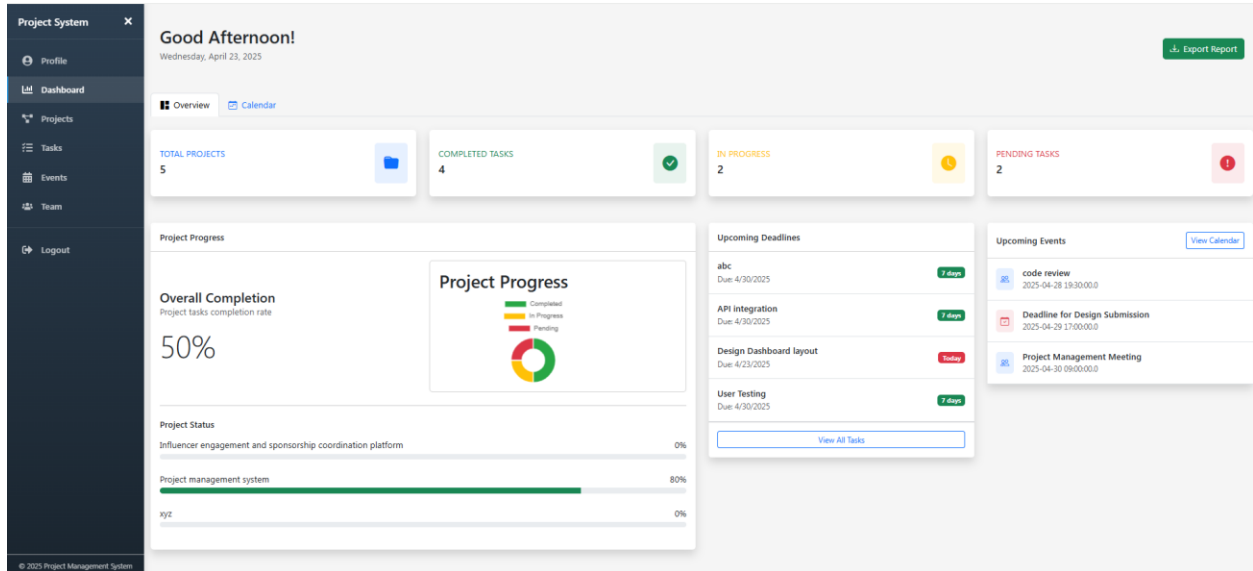


Figure 5.5 Project Manager Dashboard Page

5.6 Profile web page:

The Profile Page allows users to view and update their personal information, including username, email, role, and project preferences. Users can also edit their contact information and add a brief bio or description about their professional background. This page ensures that each user's profile is accurate and up to date.

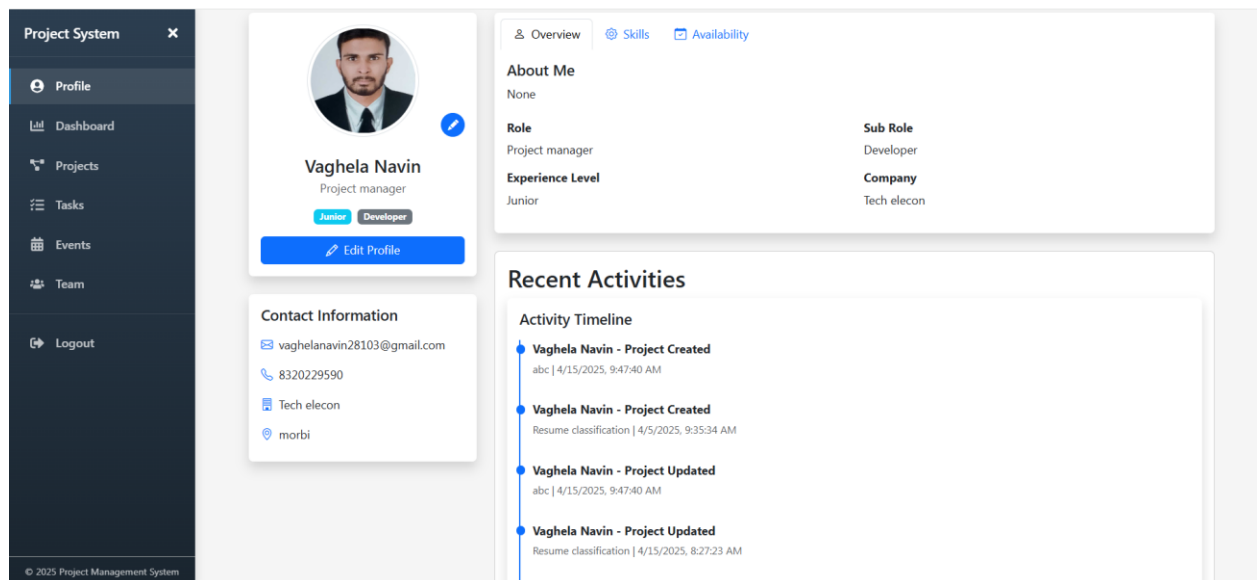
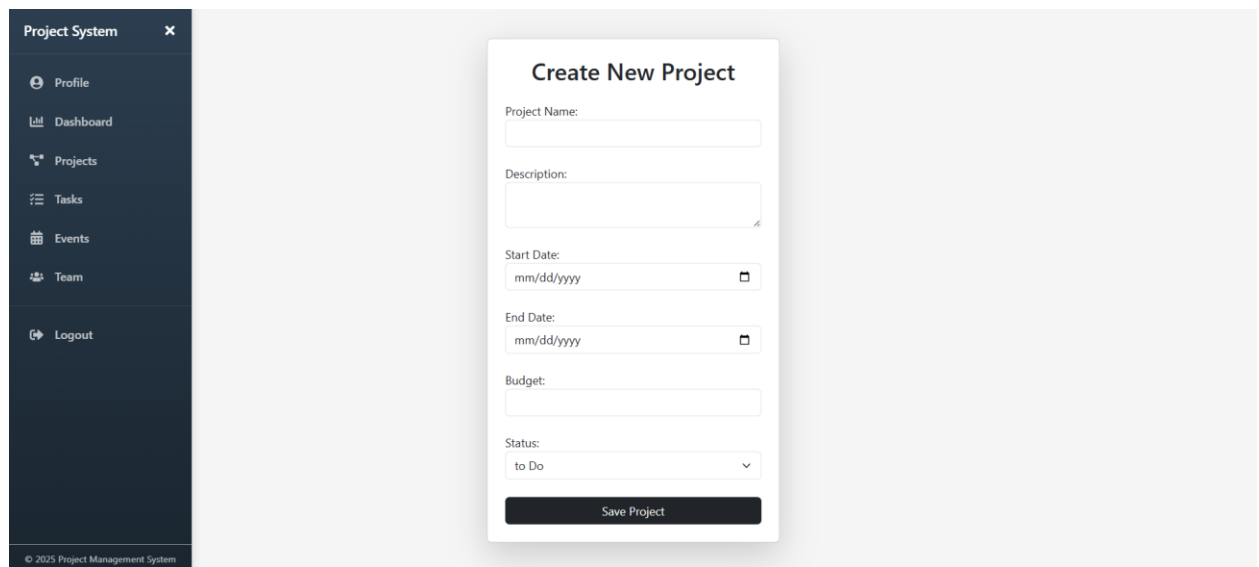


Figure 5.6 Project Manager Profile Page

5.7 Project form Page:

The Project Form Page enables Project Managers to create new projects. Project Managers will fill out a form detailing the project name, description, timeline, team members, and specific goals. The form includes clear labels and instructions, guiding users through the process of entering the project details. Once completed, users can submit the form to create the project.



The screenshot displays the 'Create New Project' form within a 'Project System' interface. On the left is a dark sidebar with navigation links: Profile, Dashboard, Projects, Tasks, Events, Team, and Logout. The main content area features a white form titled 'Create New Project'. The form contains the following fields: 'Project Name' (text input), 'Description' (text area), 'Start Date' (calendar picker with 'mm/dd/yyyy' format), 'End Date' (calendar picker with 'mm/dd/yyyy' format), 'Budget' (text input), and 'Status' (dropdown menu currently showing 'to Do'). A 'Save Project' button is located at the bottom of the form. The footer of the sidebar indicates '© 2025 Project Management System'.

Figure 5.7 Project form Page

5.8 Project list Page:

The Project List Page displays a comprehensive overview of all projects currently being managed. Project Managers can search, filter, and sort projects based on various criteria such as project status, priority, or deadline. Each project in the list includes relevant details such as its name, status, and assigned team members, enabling Project Managers to easily track progress.

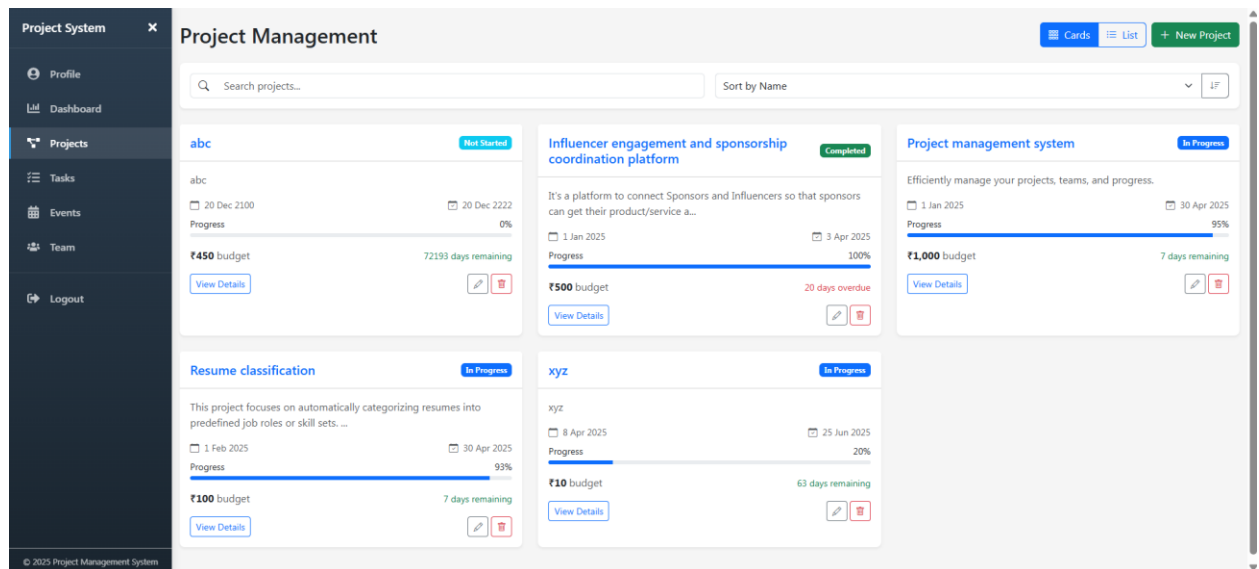


Figure 5.8 Project list Page

5.9 Task form page:

The Task Form Page allows Project Managers to create tasks associated with a project. Here, they can assign specific tasks to Team Members, set deadlines, and provide task descriptions. The page also includes options for adding task priorities and status (e.g., In Progress, Completed, Pending). Clear labels and instructions are provided to guide users through the process.

The screenshot shows the 'Add New Task' form. The sidebar is identical to the previous page. The form fields are:

- Title:** Text input field.
- Description:** Text area with a clear button.
- Project Name:** Dropdown menu with 'Select a project'.
- Status:** Dropdown menu with 'To Do'.
- Deadline:** Text input with a date picker icon and placeholder 'mm/dd/yyyy'.
- Assigned To:** Dropdown menu with 'Select a team member'.
- Priority:** Dropdown menu with 'Medium'.

A 'Save Task' button is at the bottom of the form. The footer indicates '© 2023 Project Management System'.

Figure 5.9 Task form Page

5.10 Task list page:

The **Task List Page** provides an overview of all tasks within a project, displaying important details like deadlines, assigned team members, and current status. **Project Managers** and **Team Members** can easily filter and search tasks based on various parameters, ensuring quick access to critical tasks. This page is designed to facilitate efficient task management and status tracking.

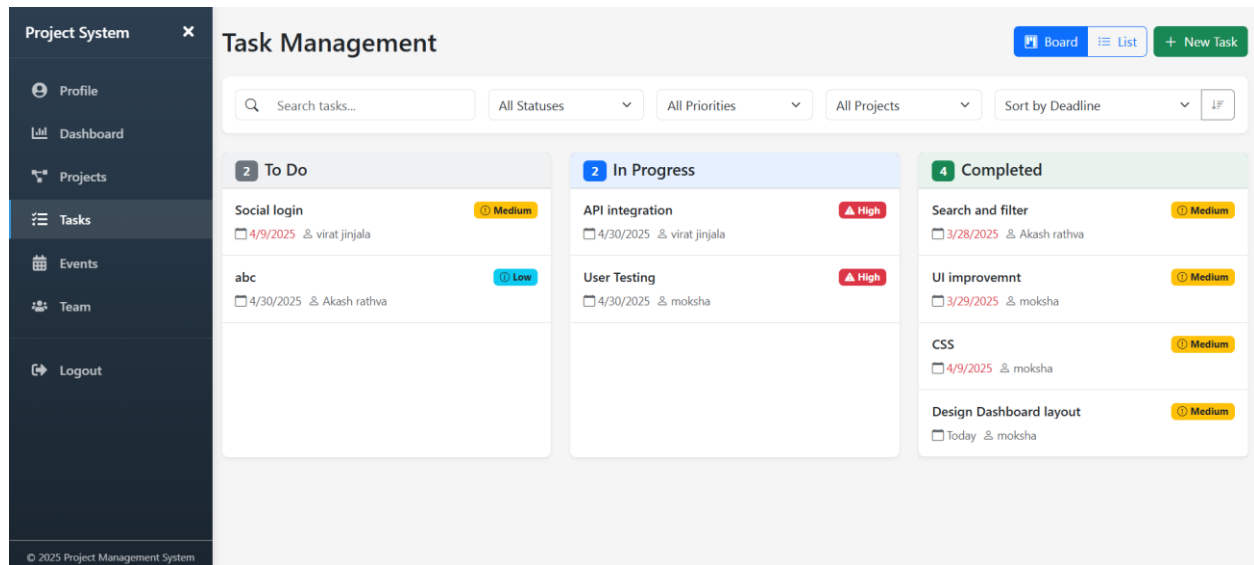


Figure 5.10 Task list Page

5.11 Team members page:

The **Team Members Page** allows **Project Managers** to view all team members associated with a particular project. The page displays the name, role, and assigned tasks for each team member, helping **Project Managers** monitor team participation and performance. Additionally, **Project Managers** can add or remove team members from the project.

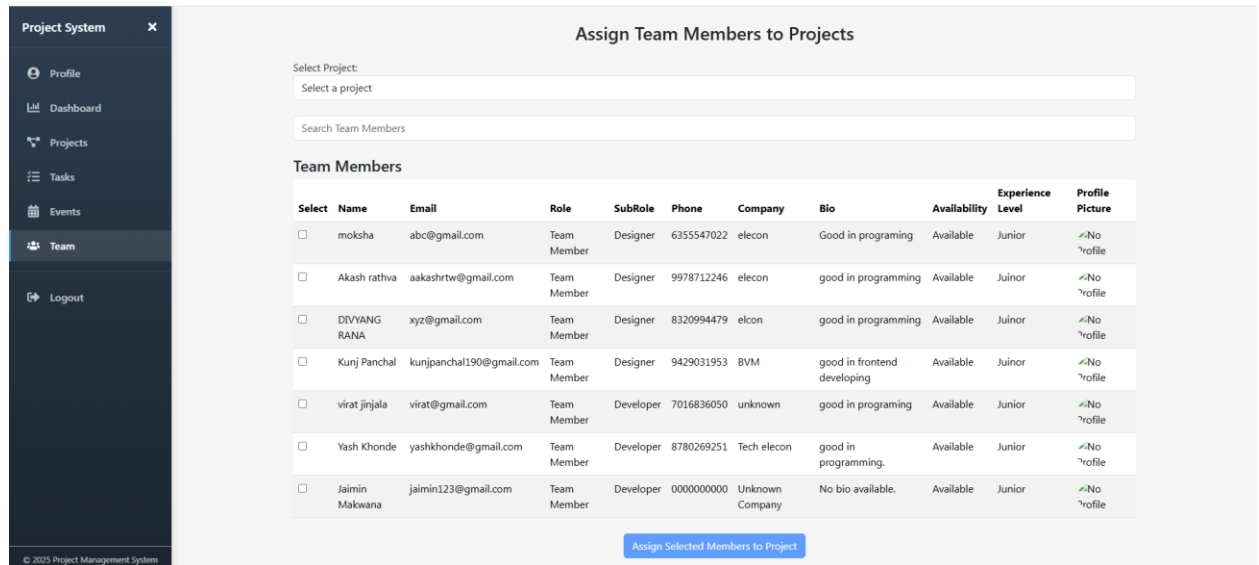


Figure 5.11 Team members Page

5.12 Team member's profile page:

The **Team Member's Profile Page** is where each **Team Member** can manage their personal information, such as name, email, and role within the project. **Team Members** can update their contact details and add a short biography. This page serves as a central location for **Team Members** to keep their information current and relevant to the project.

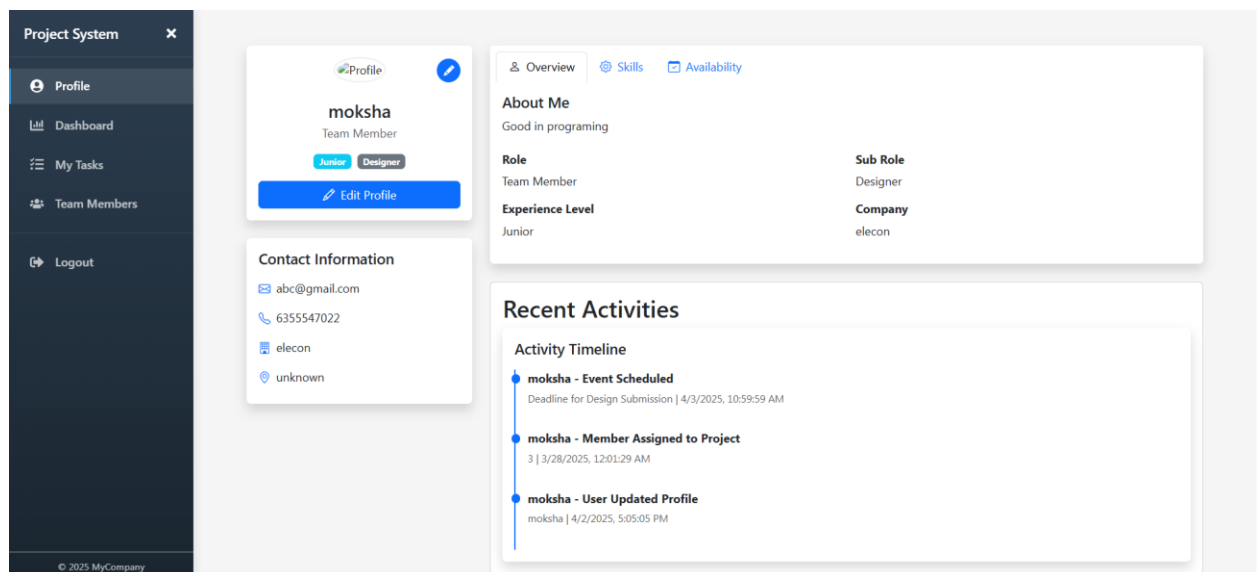


Figure 5.12 Team member's Profile Page

5.13 Team member's dashboard:

The **Team Member's Dashboard** displays a summary of all tasks assigned to the user. This dashboard is designed to help **Team Members** track their ongoing work, upcoming tasks, and deadlines. It includes a list of active projects, along with the ability to filter and prioritize tasks based on their importance or due date. The dashboard promotes effective task management and allows **Team Members** to stay organized.

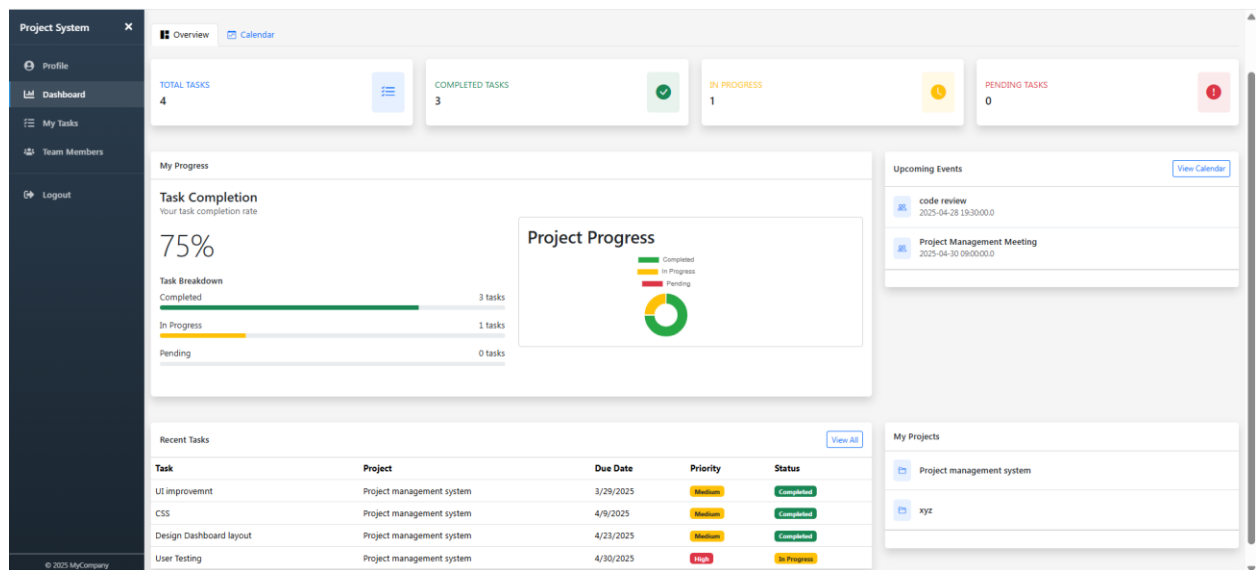


Figure 5.13 Team member Dashboard Page

5.14 Team member's task list:

The **Team Member's Task List** provides an overview of the tasks assigned to a specific **Team Member**. This page allows users to view task details, including deadlines, task descriptions, and priorities. **Team Members** can update their task status (e.g., In Progress, Completed, Pending) directly from this page, keeping everyone updated on their work progress.

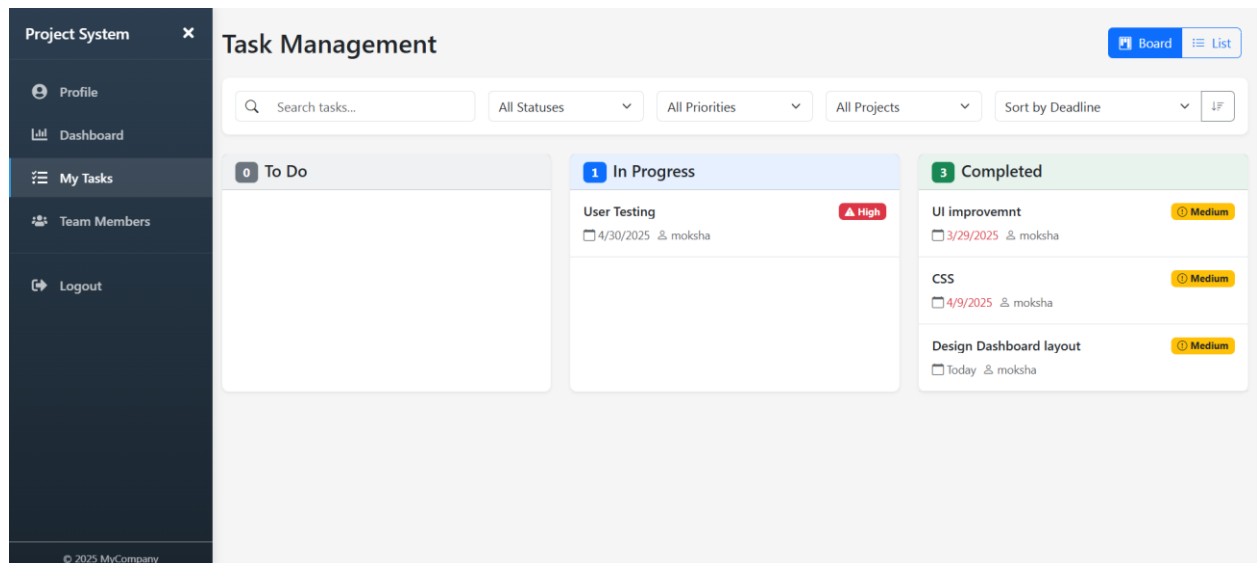


Figure 5.14 Team member's task list Page

5.15 Team member's team:

The **Team Member's Team Page** shows a list of all team members involved in the same project. This page includes relevant information about each team member, such as their role, tasks, and progress. The **Team Member's Team** page enables collaboration and communication by giving an overview of who is working on what within the project

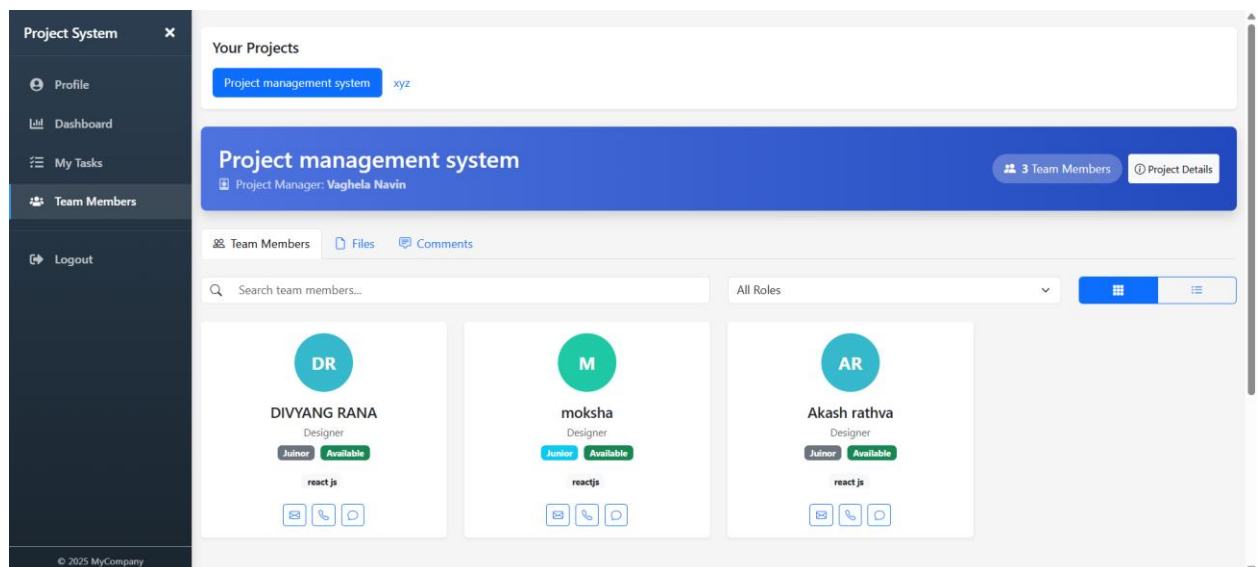


Figure 5.15 Team member's Team Page

5.16 Event form:

The **Event Creation** form allows project managers or team members to schedule new events related to their projects. Users can input the event title, description, date and time, and select the type of event (e.g., Meeting). They must also choose the associated project from a dropdown list. Once all details are filled in, the event can be created with a single click. This feature helps streamline team coordination and ensures all important events are tracked within the project system.

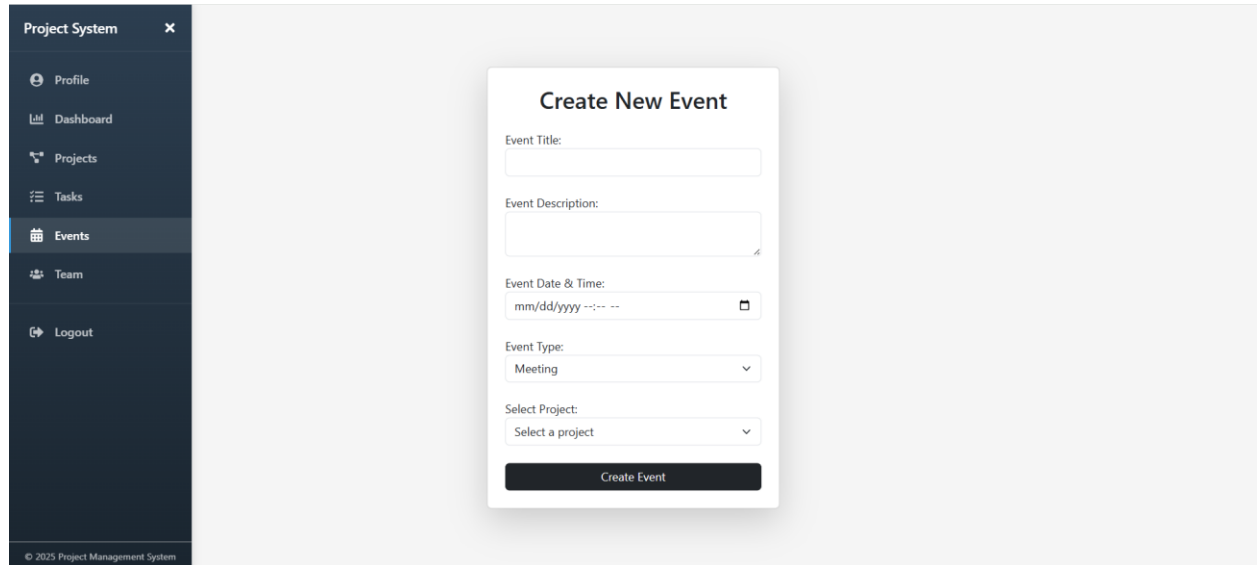
The image shows a web application interface for a 'Project System'. On the left is a dark sidebar with navigation links: Profile, Dashboard, Projects, Tasks, Events (highlighted), Team, and Logout. The main content area is light gray and features a white 'Create New Event' form. The form has the following fields: 'Event Title' (text input), 'Event Description' (text area), 'Event Date & Time' (datetime picker showing 'mm/dd/yyyy --:-- --'), 'Event Type' (dropdown menu with 'Meeting' selected), and 'Select Project' (dropdown menu with 'Select a project' as the placeholder). A dark 'Create Event' button is at the bottom of the form. The footer of the sidebar reads '© 2025 Project Management System'.

Figure 5.16 Event Form page

5.17 Admin Dashboard:

The **Admin Dashboard** of the Project Management System provides a quick and clear overview of the platform's key metrics. It displays total users, projects, tasks, and the task completion rate in visually distinct cards. A line chart shows monthly trends of tasks created and completed, helping track progress over time. A pie chart presents user distribution between project managers and team members. The dashboard also includes a sidebar for easy navigation to user, project, and task management sections.

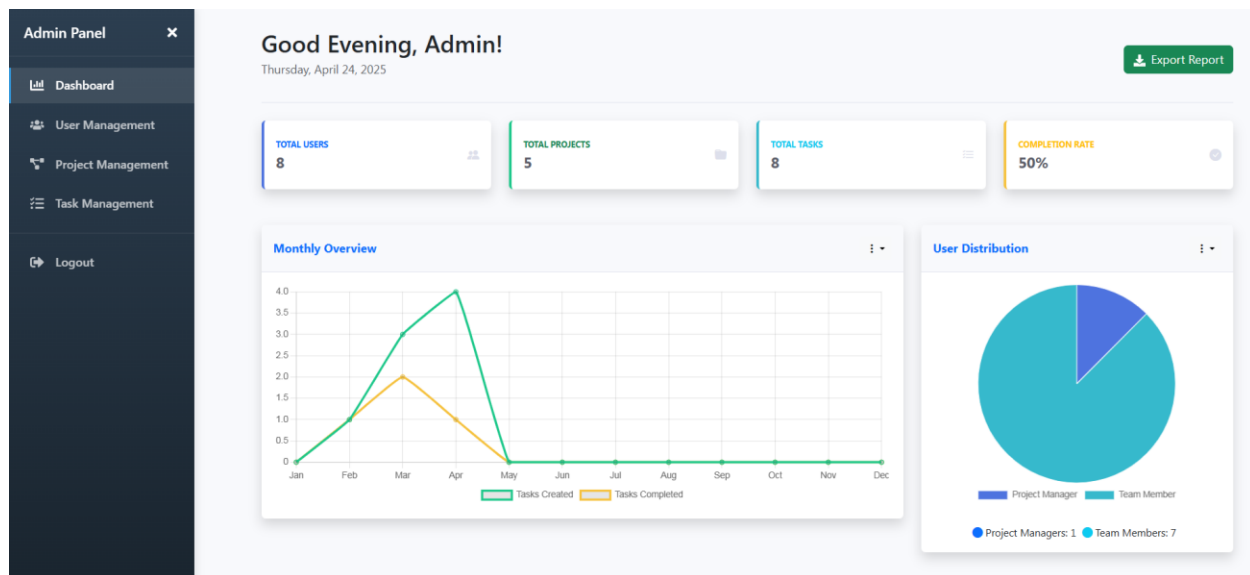


Figure 5.17 Admin Dashboard Page

5.18 Admin's Project management:

The **Project Management** page in the admin panel of the Project Management System allows administrators to efficiently oversee all ongoing projects. It provides a search bar and status filter to quickly locate projects. A bar chart shows the number of projects created each month, while a pie chart visualizes the distribution of project statuses such as "In Progress," "Completed," and "To Do." Below, a detailed table lists all projects with information like project name, manager, team size, status, deadline, and quick action buttons for view, edit, and delete. This page helps the admin track progress and manage project workflows effectively.

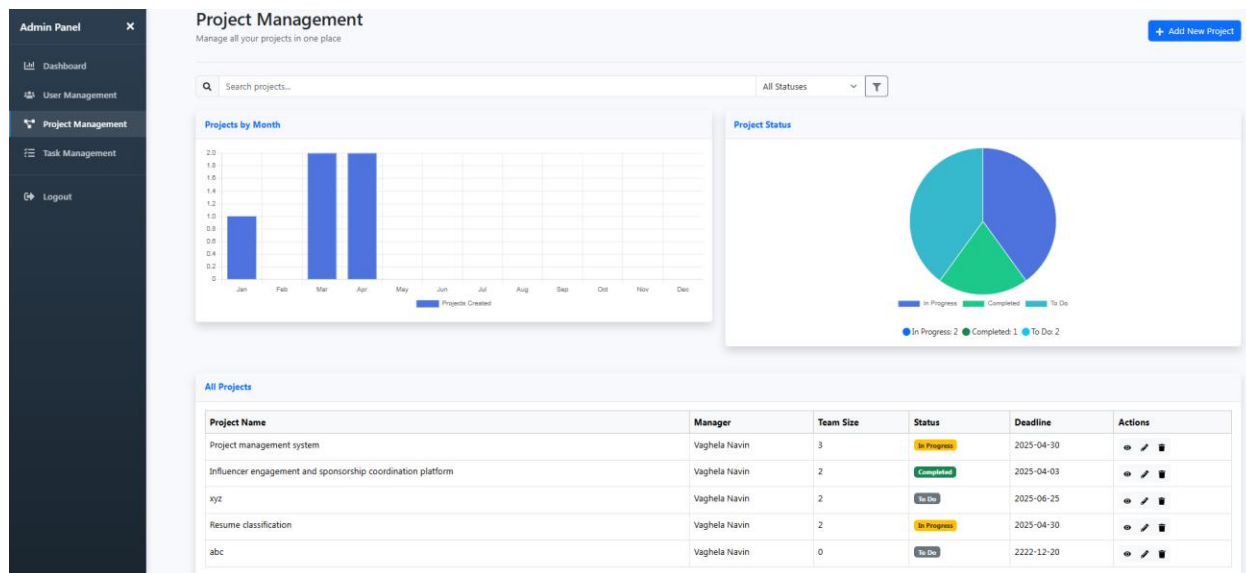


Figure 5.18 Admin Project management Page

5.19 Admin's Task management :

The Task Management page in the admin panel helps manage and track all tasks within the system. It features a search bar along with filters for status and priority, making it easy to find specific tasks. A pie chart shows the current distribution of task statuses—Completed, In Progress, and To Do. A line graph displays the trend of tasks created versus completed over the months. Below, a task table lists each task with its title, associated project, assignee, status, priority, deadline, and action buttons for viewing, editing, or deleting tasks. This page provides a clear overview of task progress and helps streamline task tracking.

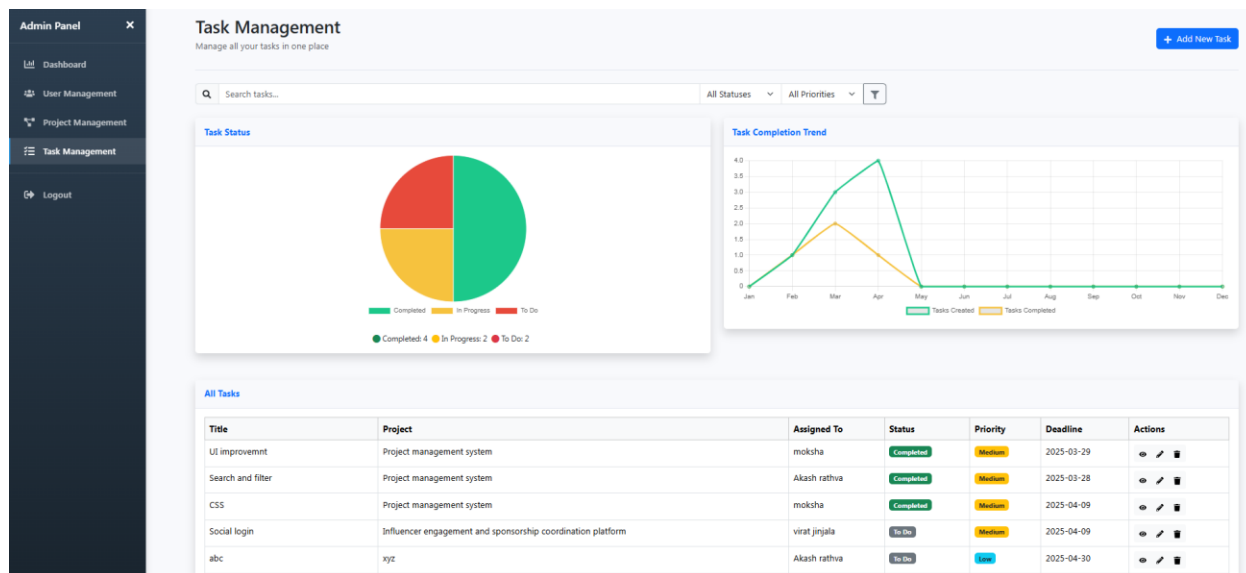


Figure 5.19 Admin Task management Page

5.20 Admin's User Management:

The **User Management** page allows the admin to oversee all registered users in the system. It includes a search bar and role filter for easy navigation. A pie chart displays the distribution of users by role—Project Manager and Team Member. The statistics panel shows the total number of users, available users, total assigned projects, and average projects per user. Below, a table lists each user's details, including name, email, role, department, project count, status, and options to view, edit, or delete user profiles. This page helps the admin manage user roles and project involvement efficiently.

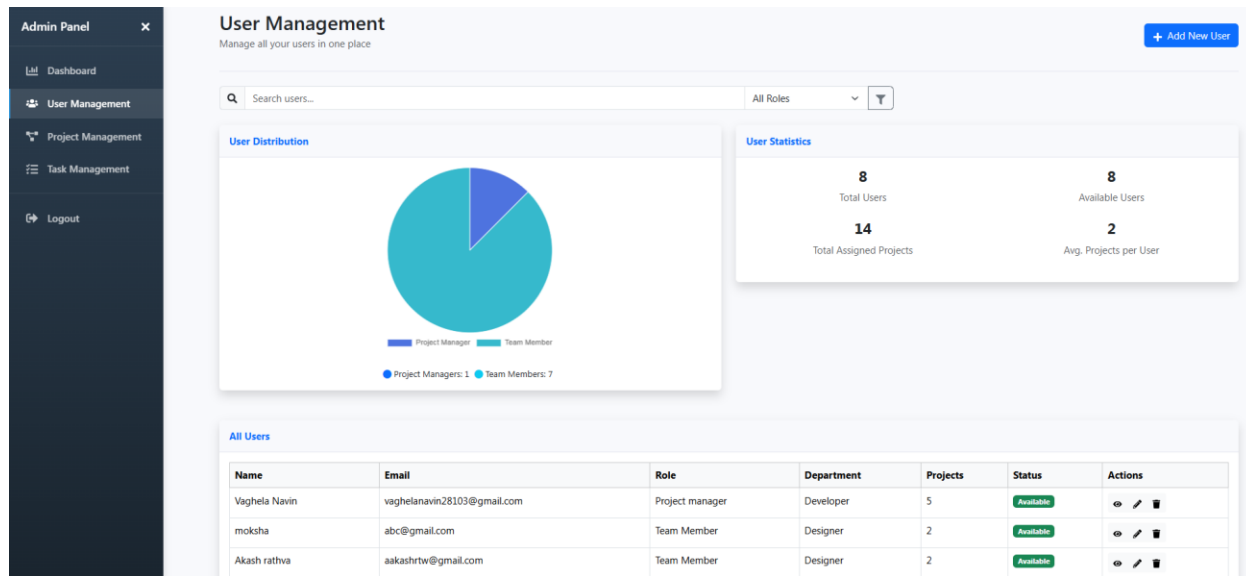


Figure 5.20 Admin User management Page

6. Conclusion

Developing a Project Management System as a single-page web application using React.js for the frontend and Java with MySQL for the backend provides multiple benefits. This system is designed to streamline project planning, task allocation, and team collaboration efficiently. Here's a breakdown of why this technology stack is ideal for our Project Management System, focusing on task management, real-time collaboration, and system scalability.

1. Fast and Responsive Navigation

Our system is built using React.js, ensuring that users can switch between different sections, such as dashboard, projects, tasks, and team members, without experiencing page reloads. This single-page application (SPA) approach enhances user experience by making navigation smooth, reducing load times, and providing a more interactive interface.

2. Seamless User Experience

With real-time updates, project managers and team members receive instant notifications about task assignments, deadlines, and project progress. This ensures that all users stay informed, reducing delays and miscommunication. The system also allows easy task tracking, providing visibility into project status and improving collaboration between team members.

3. Reusable and Scalable Components

The system is built using a component-based architecture in React.js, meaning UI elements like task lists, project cards, and team member dashboards are reusable across different sections. This modular approach ensures consistency, easier maintenance, and future scalability. Any improvements or updates to components automatically reflect throughout the system, ensuring a uniform user experience.

4. Easy Maintenance and Updates

Using Java with MySQL for backend development ensures secure and efficient data management. Each system feature is built as an independent module, making it easy to update functionalities without affecting the entire application. Whether it's adding a new project, modifying task priorities, or enhancing security, updates are seamless, ensuring that the system remains reliable and future-proof.

7. References

1. React.js, "React.js Documentation." Available: <https://react.dev/>
2. HTML (Markup Language), "HTML Documentation." Available: <https://www.w3schools.com/html/>
3. CSS, "CSS Documentation." Available: <https://www.w3schools.com/css/>
4. Tailwind CSS, "Tailwind CSS Installation Guide." Available: <https://tailwindcss.com/docs/installation>
5. Material UI, "Material UI Documentation." Available: <https://mui.com/material-ui/>
6. React Router DOM, "React Router Documentation." Available: <https://reacttraining.com/react-router>
7. React Hooks, "React Hooks Introduction." Available: <https://legacy.reactjs.org/docs/hooks-intro.html>
8. JavaScript, "JavaScript Documentation." Available: <https://www.javascript.com/>
9. GitHub, "GitHub Platform." Available: <https://github.com/index>
10. Font Awesome, "Font Awesome Icons." Available: <https://fontawesome.com/>
11. Draft.js, "Draft.js Documentation." Available: <https://draftjs.org/>
12. React-Recharts, "React-Recharts Documentation." Available: <https://recharts.org/>
13. React Modal, "React Modal GitHub Repository." Available: <https://github.com/reactjs/react-modal>