A Major Project Synopsis on

Employees Task Management Tool

Submitted to Manipal University, Jaipur

Towards the partial fulfillment for the Award of the Degree of

MASTER OF COMPUTER APPLICATIONS

2023-2025

by

Rupa Kumari

23FS20MCA00062

MANIPAL UNIVERSITY
JAIPUR

Under the guidance of

Dr. Arpana sinhal

Department of Computer Applications

School of AIML, IoT&IS, CCE, DS and Computer Applications

Faculty of Science, Technology and Architecture

Manipal University Jaipur

Jaipur, Rajasthan

2025

Employee Task Management System

I. Introduction

The **Employee Task Management System** is a web-based application designed to streamline task assignment, tracking, and management. The frontend is built using **React, Next.js, Tailwind CSS, and MUI**, ensuring a clean and responsive user interface.

With a focus on user experience, the system provides a well-structured layout, intuitive navigation, and dynamic components for managing tasks efficiently. The integration of **MUI tables** allows for easy data visualization, while modals simplify task creation and updates. Tailwind CSS ensures a modern and adaptable design, making the interface seamless across different devices.

By leveraging these technologies, the system enhances productivity by offering a fast, interactive, and user-friendly platform for employees and managers alike.

II. Motivation

Efficient task management is crucial for maintaining productivity in any organization. Traditional methods, such as manual tracking or scattered communication channels, often lead to mismanagement, missed deadlines, and reduced efficiency.

The motivation behind developing this **Employee Task Management System** is to provide a structured, digital solution that simplifies task allocation, monitoring, and reporting. By creating a user-friendly and visually appealing interface using **React, Next.js, Tailwind CSS, and MUI**, this system aims to:

- **Enhance Productivity:** Enable employees to track their tasks in real-time, reducing confusion and delays.
- **Improve Collaboration:** Provide a seamless way for employees and administrators to communicate and manage work progress.
- **Automate Workflows:** Reduce manual effort with automated task updates, notifications, and daily email reports.
- **Ensure Accessibility:** A responsive, modern UI that works across devices, improving usability for all employees.

By leveraging the latest frontend technologies, this system offers an intuitive and efficient way to manage tasks, ensuring that organizations can operate smoothly with minimal administrative overhead.

III. Problem Statement

In many organizations, task management is often handled through fragmented methods such as spreadsheets, emails, or verbal communication. These traditional approaches lead to several challenges, including:

- **Lack of Centralization:** Employees and administrators struggle to keep track of assigned tasks, leading to mismanagement and inefficiencies.
- **Poor Task Visibility:** Without a structured system, employees may not have a clear understanding of their responsibilities, priorities, and deadlines.
- **Inefficient Communication:** Task updates and progress tracking become cumbersome when handled through multiple, disconnected channels.
- **Manual Reporting:** Administrators need to spend extra time generating task reports, leading to delays and increased workload.
- **Limited Accessibility:** A non-responsive or outdated task management system restricts employees from efficiently accessing their tasks on different devices.

To address these challenges, the **Employee Task Management System** is designed with a **modern frontend interface** using **React, Next.js, Tailwind CSS, and MUI**. This system provides a centralized, user-friendly, and responsive platform for managing tasks, ensuring efficiency and transparency in the workplace.

IV. Methodology & Work Plan

Methodology

The development of the **Employee Task Management System** follows a structured approach to ensure a seamless and efficient frontend experience. The methodology involves the following key steps:

1. **Requirement Analysis:** Understanding the needs of employees and administrators to define essential features like task creation, assignment, tracking, and reporting.
2. **UI/UX Design:** Designing wireframes and prototypes to create an intuitive and user-friendly interface using modern frontend technologies.
3. **Frontend Development:**
    - Setting up the project using **React 19** and **Next.js** for optimal performance.
    - Styling with **Tailwind CSS** to achieve a responsive and visually appealing design.
    - Integrating **MUI components** for structured layouts, tables, forms, and pagination.
4. **State Management:** Implementing the **Context API** to manage global states, such as modal visibility and user interactions.
5. **Feature Development:**
    - Creating and managing tasks through an interactive UI.
    - Implementing **real-time notifications** using **React-Toastify**.
    - Enhancing task descriptions with **Quill Editor**.
    - Using a **global modal system** for consistency across the application.
    - Automating **daily task reports via email**, ensuring proper role-based distribution.
6. **Testing & Debugging:** Conducting thorough UI testing to identify and fix design inconsistencies, ensuring smooth user interactions.
7. **Deployment & Maintenance:** Hosting the frontend on a reliable platform, gathering user feedback, and making iterative improvements for enhanced performance.

Planning of Work

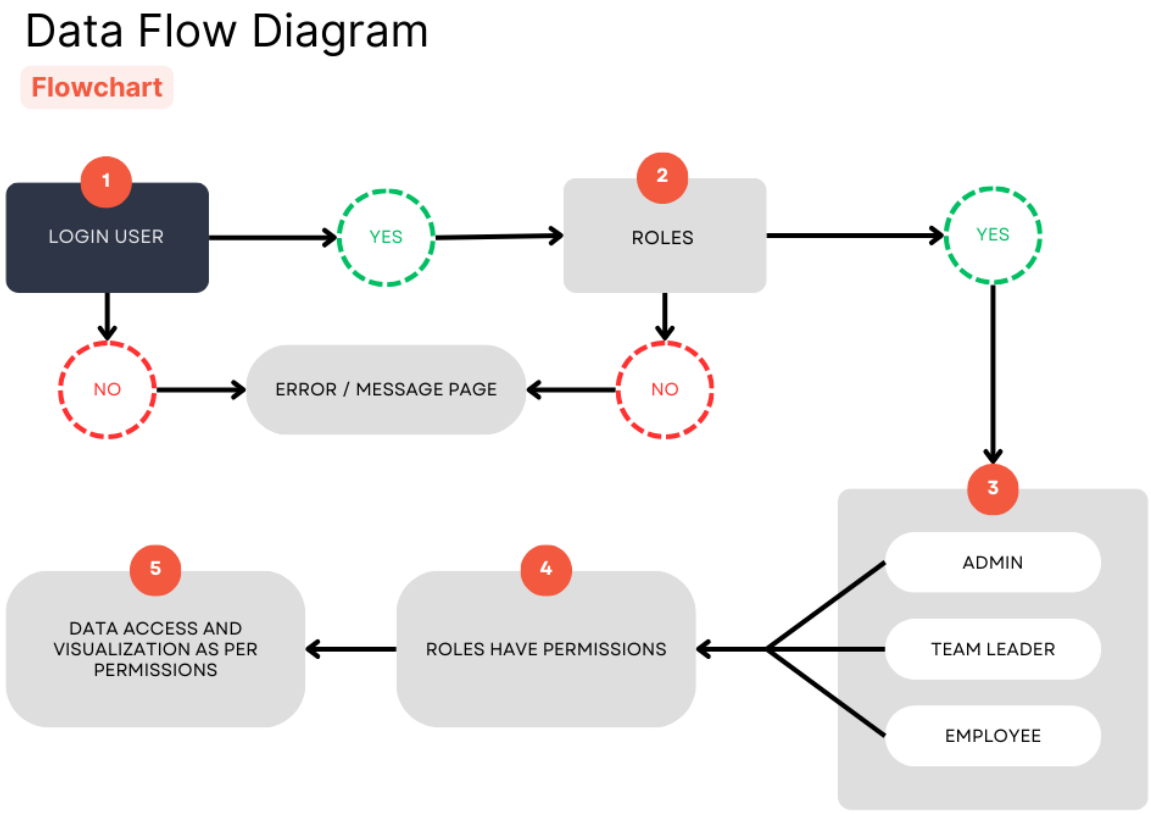
The development process is divided into different phases to ensure a structured workflow:

- **Phase 1 – Planning & Requirement Analysis:** Gathering requirements, defining the scope, and designing UI wireframes.
- **Phase 2 – Frontend Development:** Setting up the project, developing core UI components, and integrating the task management system.
- **Phase 3 – Feature Implementation:** Adding advanced functionalities like notifications, rich text editing, modal management, and email reports.
- **Phase 4 – Testing & Optimization:** Conducting rigorous UI/UX testing, debugging issues, and optimizing performance.
- **Phase 5 – Deployment & Maintenance:** Deploying the application, collecting feedback, and continuously improving the frontend experience.

System Design (DFD & ER Diagram)

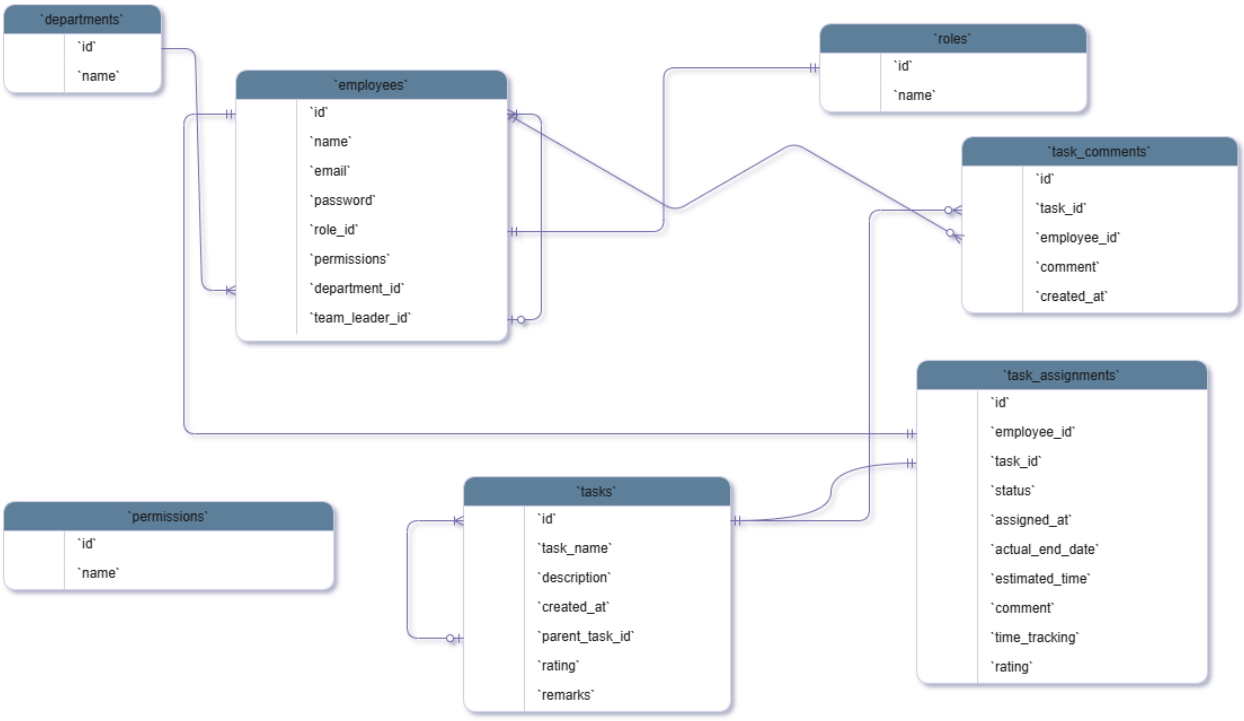
DFD (Data Flow Diagram):

The Data Flow Diagram outlines the flow of information:
1. User Interactions: Employees and Managers log in and perform actions.
2. Task Assignment: Managers create and assign tasks with deadlines.
3. Task Updates: Employees update task progress, tracking time spent.
4. Notifications & Reports: System generates alerts and task performance reports.



ER Diagram:

The Entity-Relationship diagram includes the following entities:
• User (_id, name, email, password, role, createdAt, updatedAt)
•Task (_id, title, description, assigned_to, status, start_date, end_date, estimate_time, actual_end_date, createdAt, updatedAt)
• Comment (_id, task_id, user_id, message, createdAt, updatedAt)
• TimeTracking (_id, task_id, user_id, time_spent, time_remaining, createdAt, updatedAt)
• Notification (_id, user_id, message, createdAt, updatedAt)


`departments`
`id`
`name`
`employees`
`id`
`name`
`email`
`password`
`role_id`
`permissions`
`department_id`
`team_leader_id`
`roles`
`id`
`name`
`task_comments`
`id`
`task_id`
`employee_id`
`comment`
`created_at`
`task_assignments`
`id`
`employee_id`
`task_id`
`status`
`assigned_at`
`actual_end_date`
`estimated_time`
`comment`
`time_tracking`
`rating`
`tasks`
`id`
`task_name`
`description`
`created_at`
`parent_task_id`
`rating`
`remarks`
`permissions`
`id`
`name`

V. Requirements for Proposed Work

Develop an efficient and user-friendly Employee Task Management System, the following requirements are necessary:

1. Technical Requirements
    - Frontend Framework: React 19 and Next.js for a dynamic and optimized UI.
    - Styling & Design: Tailwind CSS for responsive and modern UI components.
    - UI Components Library: MUI for tables, forms, pagination, and modals.
    - State Management: Context API for handling global states, such as modal visibility.
    - Rich Text Editing: Quill Editor for detailed task descriptions.
    - Notifications: React-Toastify for real-time user alerts.
    - Data Handling: API integration with the backend for task management, user roles, and reports.
    - Email Reports: Automated daily task reports sent to employees based on their roles.
2. Hardware Requirements
    - A system with minimum 8GB RAM and a multi-core processor for smooth development.
    - A stable internet connection for accessing libraries, APIs, and deployment.
3. Software Requirements
    - Code Editor: VS Code for development.
    - Version Control: Git and GitHub for project management.
    - Package Manager: npm or yarn for installing dependencies.
    - Browser: Google Chrome or Firefox for testing UI responsiveness.
4. Human Resource Requirements
    - Frontend Developer: Responsible for UI/UX design, component development, and API integration.
    - Backend Developer: Handling data storage, authentication, and API development using Node.js and SQL.
    - QA Tester: Ensuring bug-free and smooth user interactions.

By fulfilling these requirements, the proposed Employee Task Management System will provide an efficient and seamless task management experience for employees and administrators.

VI. Bibliography & References

- *https://www.npmjs.com/package/react*
- *https://mui.com/material-ui/react table/?srsltid=AfmBOoqvHDhrlZJ4J6hkPO4zEkfbigLLkM4G1LD4lVSK50XklmmQXQo-*

- *https://nextjs.org/docs/pages/building-your-application/routing/dynamic-routes*
- *https://nextjs.org/docs/pages/building-your-application/data-fetching/client-side*