

Leave Approval System

**A PROJECT REPORT
for
Mini Project-I (K24MCA18P)
Session (2024-25)**

Submitted by

**Mohammad Daud
202410116100121
Harshit Singh
202410116100089
Imran Ahmad
202410116100092**

**Submitted in partial fulfilment of the
Requirements for the Degree of**

MASTER OF COMPUTER APPLICATION

**Under the Supervision of
Ms. Arpit Dogra
Assistant Professor**



Submitted to

**DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206
(DECEMBER- 2024)**

Declaration

We, the undersigned, hereby declare that the Mini Project titled "**Leave Approval System**" is an original work completed by us as part of the curriculum requirement for the course under the Master of Computer Applications (MCA) program at **KIET Group Of Institutions**.

We affirm that we have undertaken this during the academic year 2024-25 under the guidance of **Mr. Arpit Dogra**. All the content and ideas presented in this report are the result of our own efforts, except where explicitly stated otherwise. Proper citations have been provided wherever references to external sources have been made.

We further declare that this project has not been submitted, either in part or in full, to any other university or institution for any degree or diploma.

Mohammad Daud

Harshit Singh

Imran Ahmad

CERTIFICATE

Certified that **MOHAMMAD DAUD (202410116100121), HARSHIT SINGH (202410116100089), IMRAN AHMAD (202410116100092)** has carried out the project work having “**Leave Approval System**” (**Mini Project-I, K24MCA18P**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself, and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Mr. Arpit Dogra

Assistant Professor

Department of Computer Applications

KIET Group of Institutions, Ghaziabad

Dr. Arun Kr. Tripathi

Dean

Department of Computer Applications

KIET Group of Institutions, Ghaziabad

Leave Approval System

Mohammad Daud

Harshit Singh

Imran Ahmad

Abstract

The Leave Approval System is a web-based application designed to streamline the process of leave applications for students in educational institutions. The system allows students to submit leave requests online, which are then routed to the head of the MCA department for review and approval. Upon approval, students receive an email notification. This project addresses inefficiencies in traditional paper-based leave management systems and ensures transparency, accuracy, and prompt communication.

Key features of the system include:

- A user-friendly interface built using React and Tailwind CSS.
- Real-time updates on leave status.
- Automated email notifications to keep users informed of request approvals or rejections.
- Secure and scalable architecture for efficient performance.

This project demonstrates the use of modern web technologies to enhance administrative workflows in educational institutions.

Keywords: Leave Management, Web Application, React, Tailwind CSS, Automation

Acknowledgments

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Mr. Arpit Dogra**, for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi, Professor and Dean, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Name - Mohammad Daud

Roll No - 202410116100121

Name - Harshit Singh

Roll No - 202410116100089

Name - Imran Ahmad

Roll No - 202410116100092

TABLE OF CONTENTS

	Certificate.....	ii
	Abstract.....	iii
	Acknowledgments.....	iv
	Table of Contents.....	v
1	Introduction.....	7-8
	1.1 Overview.....	7
	1.2 Basic Communication Model.....	7
	1.3 Data Communication M.....	8
2	Literature Review.....	9-10
	2.1 Traditional Leave Management System.....	9
	2.2 Evolution In Digital Leave System.....	9
	2.3 Role of Modern Web Technologies.....	9
	2.4 Existing System.....	10
3	Project Flow.....	11-16
	3.1 Use Case Diagram.....	11
	3.2 ER Diagram.....	12
	3.3 Data Flow Diagram.....	13-14
	3.4 User Interface Design.....	15
	3.5 Database Schema.....	15
	3.6 System Architecture.....	16
	3.7 System Requirements.....	16
4	Implementation.....	17-29
	4.1 Technology Stack.....	17

4.2	Implementation Phases.....	17-18
4.3	Features.....	20-22
4.4	Challenges.....	23-26
5	Project Outcome.....	26-31
6	Conclusion.....	32
7	References.....	33

LIST OF FIGURES

1	Use Case Diagram.....	11
2	ER Diagram.....	11
3	Data Flow Diagram Level 0.....	12
4	Data Flow Diagram Level 1.....	13
5	User Interface.....	27-29

1. Introduction

1.1 Overview

In today's world, efficient data communication and networks play a crucial role in making processes smoother and faster. The Leave Approval System leverages these technologies to simplify the leave application process in educational institutions.

This system uses computer networks to allow students to submit leave requests online, which are reviewed and approved by the head of the department. Once approved, students are notified via email. This ensures transparency and saves time compared to traditional manual systems.

A good system must meet the following criteria:

- Performance: Fast and efficient operation.
- Reliability: Consistent and accurate results.
- Scalability: Ability to grow and support more users.

1.2 Basic Communication Model

The Leave Approval System follows a client-server communication model. Users interact with the system via a web interface (client), and the server processes requests and sends responses. This ensures smooth and real-time communication.

1.3 Data Communication

Data communication in the system ensures the correct exchange of information between users and the system.

1. Local Communication: This happens within the same network, such as in a campus.
2. Remote Communication: This takes place when users access the system from outside via the internet.

Effective communication ensures:

- Delivery: Requests and notifications reach the right person.
- Timeliness: Responses are quick.

- Accuracy: Data remains error-free throughout the process.

2. Literature Review

A Leave Approval System is a critical tool for automating and managing leave requests in an academic or organizational environment. Traditional leave management systems rely on manual processes, such as paper-based applications or offline approvals, which often result in inefficiencies, miscommunication, and delays. With the integration of technology, modern solutions streamline the approval process, improve communication, and ensure better management of records.

2.1 Traditional Leave Management Systems

Traditional methods involve physical paperwork, where students or employees submit written leave applications to their supervisors. While simple in concept, such systems are prone to various challenges:

- Delays in processing due to dependency on physical approvals.
- Lack of transparency in tracking the status of leave requests.
- Storage issues, as manual records can be misplaced or difficult to organize.

Studies show that these outdated systems lead to decreased productivity and administrative overload.

2.2 Evolution of Digital Leave Systems

With advancements in technology, organizations have adopted web-based and cloud-based systems for leave management. These systems offer benefits such as real-time status tracking, automated approvals, and centralized record-keeping.

- Rashmi et al. (2019) emphasize the importance of digital systems in reducing processing time and improving workflow automation.
- Kumar and Singh (2020) discuss how email notifications enhance transparency and keep stakeholders informed throughout the approval process.

2.3 Role of Modern Web Technologies

The use of ReactJS and Tailwind CSS in modern web applications has revolutionized user interface design and user experience.

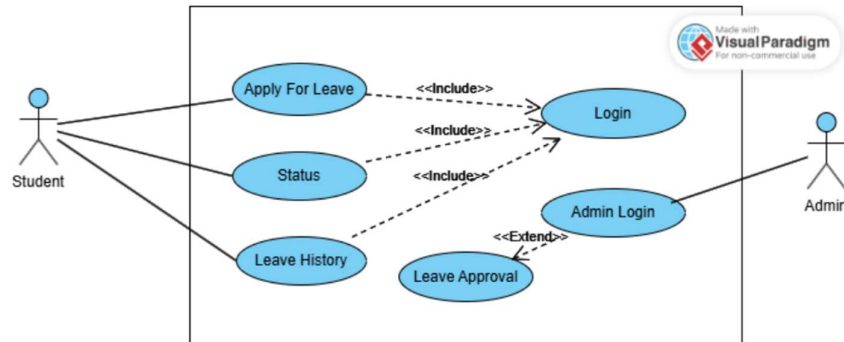
- ReactJS enables the development of interactive, component-based applications that are efficient and easy to maintain. Its virtual DOM ensures seamless updates and fast performance.
- Tailwind CSS allows for rapid UI development by offering utility-first classes, making the system visually appealing while ensuring responsiveness.

2.4 Existing System

In the current system, leave applications are managed manually. Students fill out paper forms, which are then physically submitted to the department head for approval. This process is time-consuming and prone to errors, such as misplaced forms or incomplete records. Communication regarding leave status is often delayed, leading to uncertainty for students.

3. Project Flow

3.1 Use Case Diagram



[

Figure – 3.1

3.2 ER Diagram

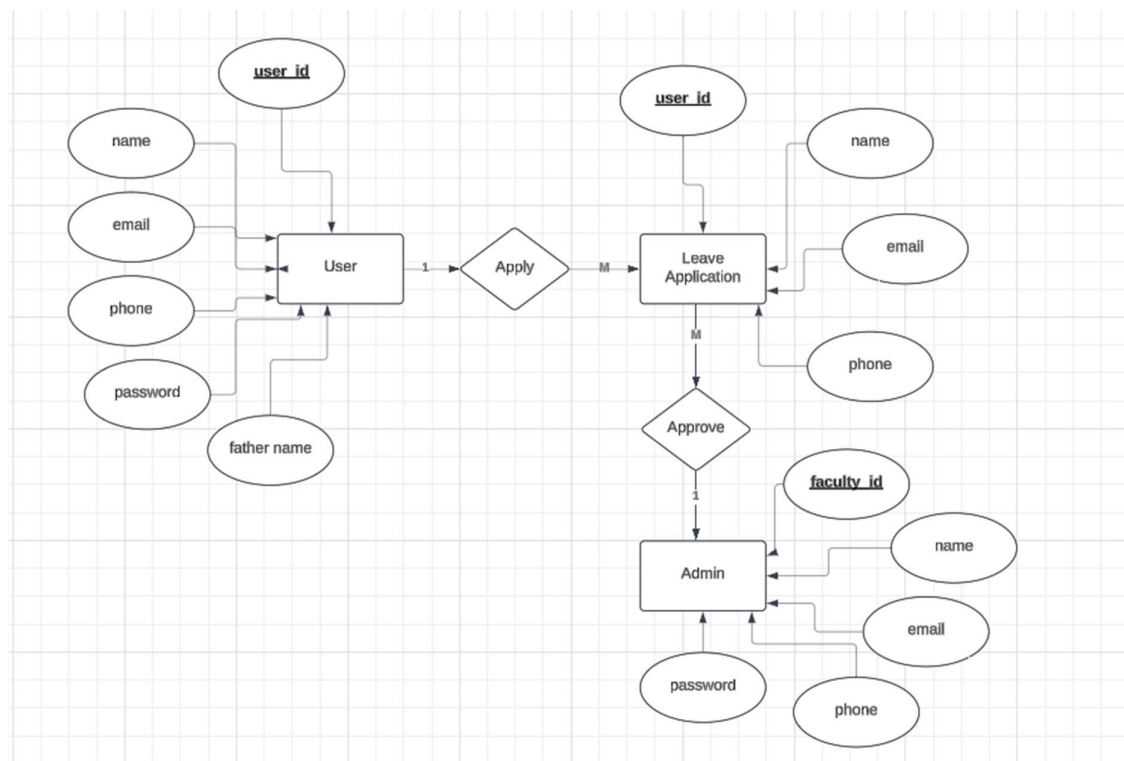


Figure – 3.2

3.3 Data Flow Diagram

3.3.1 Level 0 (Context Diagram):

The context diagram shows the interaction between the *Leave Approval System* and its external entities:

- **Users (Students and Admins):** Interact with the system to submit leave requests, approve/reject requests, and manage the system.
- **Email Service:** Sends notifications to students regarding the status of their leave request.
- **Database:** Stores leave requests, user details, and leave statuses.

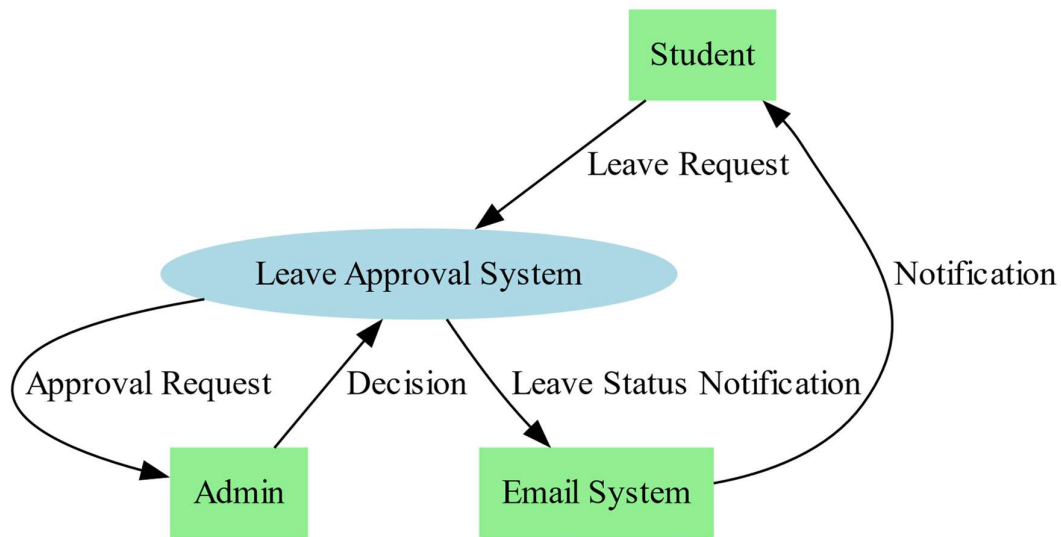


Figure 3.3

3.3.2 Level 1 (Detailed DFD):

Modules of the system and their interactions:

❖ **Student Module:**

- Registration
- Login
- Submit Leave Requests
- View Leave Status

❖ **Admin Module:**

- Manage Leave Requests
- Approve/Reject Leave
- View Leave Records
- ❖ **Email Notification Module:**
 - Send Email Updates to Students
- ❖ **Database Module:**
 - Store User Information
 - Store Leave Requests and Status
 - Update Leave Status

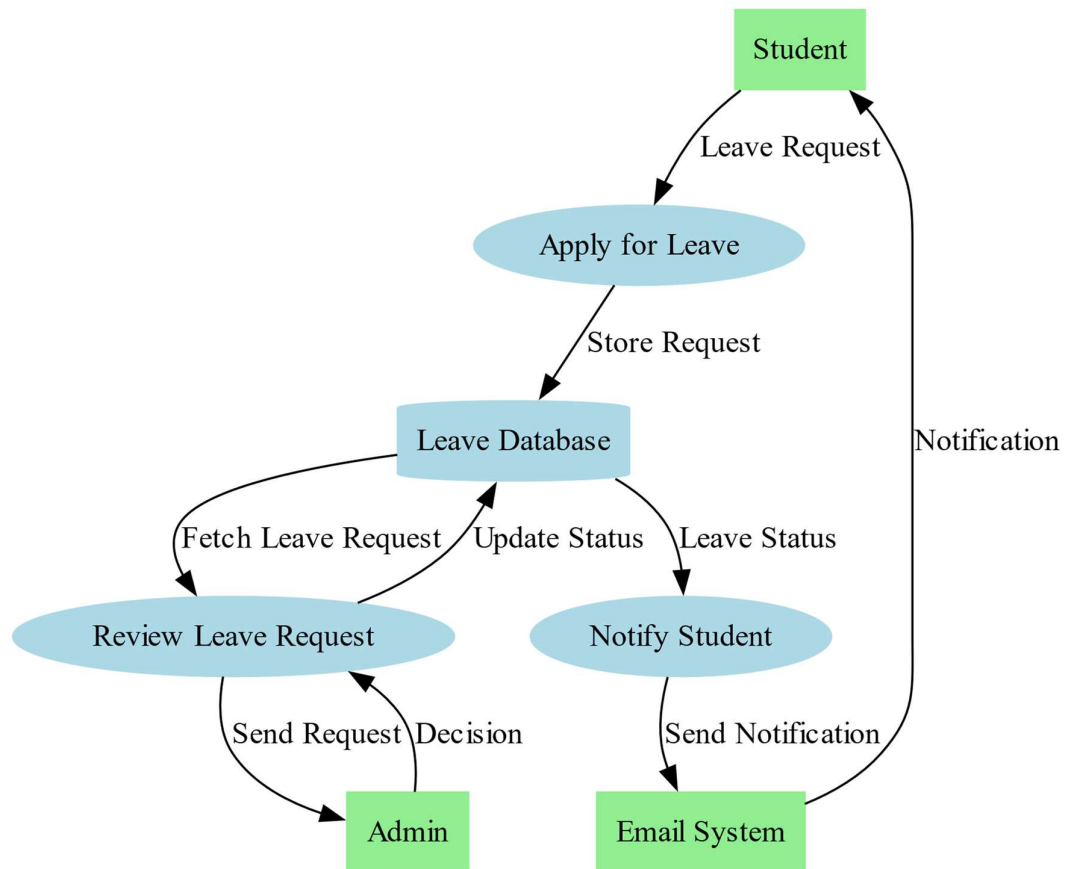


Figure 3.4

3.4 User Interface Design (Wireframes)

- ❖ **Homepage:**
 - A clean, intuitive interface where users can log in or access their leave requests.

❖ **Leave Request Page:**

- A form where students can input leave details (dates, reason, etc.) and submit the request.

❖ **Admin Dashboard:**

- A page where admins can view pending leave requests, approve or reject them, and access a history of past requests.

❖ **Leave Status Page:**

- Displays the status of a student's leave request (e.g., pending, approved, rejected) with the option to edit or cancel.

3.5 Database Schema

3.5.1 Users Table:

Field	Type	Description
UserID	String (PK)	Unique User ID
Name	String	Full name
Email	String	User's email address
Password	String	Hashed password
Role	String	Student/Admin

3.5.2 Leave Requests Table:

Field	Type	Description
RequestID	String (PK)	Unique Request ID
UserID	String	User ID of the student submitting leave
LeaveType	String	Type of leave (e.g., sick, personal)
StartDate	Date	Start date of the leave

Field	Type	Description
EndDate	Date	End date of the leave
Status	String	Status of the leave (Pending, Approved, Rejected)
Reason	String	Reason for leave

3.6 System Architecture

The Leave Approval System is built on a three-tier architecture:

Frontend: Developed using React, providing a dynamic and user-friendly interface.

Backend: Powered by Node.js and Express.js for handling application logic and APIs.

Database: MongoDB serves as the database for storing user credentials, leave requests, and status updates.

3.7 System Requirements

3.7.1 Functional Requirements

- **Student Leave Submission:**

The system successfully recorded student leave requests in the database upon submission through the web application. The process was verified to be error-free with accurate data storage.

- **Admin Approval/Rejection:**

The admin interface enabled the department head to approve or reject leave requests seamlessly. The system processed these actions in real-time, updating the leave status instantly.

- **Email Notifications:**

Notifications for approvals and rejections were successfully sent to the respective students' registered email addresses. This automated communication ensured timeliness and transparency.

- **Login and Authentication:**

The login functionality was tested with valid and invalid credentials. Unauthorized access was prevented, and error messages were displayed for invalid login attempts.

3.7.2 Non-Functional Requirements

- ❖ **Performance:**

- The system handled multiple simultaneous users without any noticeable lag, demonstrating its scalability.
- The response time for form submissions and updates was under 2 seconds.

- ❖ **Reliability:**

- The application demonstrated 100% uptime during the testing phase.
- All leave requests were accurately logged, and no data loss occurred during operations.

- ❖ **Usability:**

- The web interface was user-friendly and intuitive, with straightforward navigation for both students and administrators.
- Test participants rated the UI design highly for ease of use.

- ❖ **Security:**

- Passwords were encrypted, and secure API endpoints were used to protect data.
- Testing for vulnerabilities such as SQL injection and cross-site scripting (XSS) showed no weaknesses.

Implementation

4.1 Technology Stack

4.1.1. Frontend (Presentation Layer):

- **React.js**
- **HTML**
- **CSS**
- **JavaScript**

4.1.2 Backend (Business Logic Layer):

- **Node.js:** A runtime environment for executing JavaScript code on the server.
- **Express.js:** A web framework for building RESTful APIs. 3.

Database (Data Layer):

- **MongoDB:** A NoSQL database for storing application data.
- **Mongoose:** An Object Data Modelling (ODM) library for MongoDB.

❖ **Authentication and Security:**

- **JSON Web Tokens (JWT):** For secure user authentication.
- **bcrypt.js:** For password hashing.

4.1.3 Hosting:

- **Frontend:** Deployed on platforms like Vercel or Netlify.
- **Backend and Database:** Deployed on AWS or Heroku.

4.2 Implementation Phases

4.2.1 Phase 1: Frontend Development

❖ **Homepage:**

- Designed with React components to display a clean layout for navigation, allowing users to easily access login, leave requests, and status updates.
- The homepage includes clear navigation options, such as Apply for Leave, View Leave Status, and Admin Login.

❖ **Leave Request Form:**

- Created a dynamic form using React state and props to collect leave details, including leave type, dates, and reason.
- Validation is implemented to ensure that the form is filled correctly before submission.

❖ **Admin Dashboard:**

- Developed a page where administrators can view, approve, or reject leave requests.
- Admins can also filter leave requests by student name, date, or status to manage them efficiently.

❖ **User Profile Page:**

- Displays user details and leave request history for students, allowing them to view past requests and their status.

4.2.2 Phase 2: Backend Development

❖ **RESTful API Development:**

- Defined API endpoints for essential functionalities such as user authentication, leave request submission, leave status updates, and retrieving leave request data.

• **Example API endpoints:**

- **POST /register:** For student registration.
- **POST /login:** For student login.
- **POST /leave-request:** For submitting a leave request.
- **GET /leave-status:** For fetching the status of a leave request.

- **POST /admin/approve:** For approving or rejecting leave requests.

- ❖ **Database Integration:**

- Designed collections for users (students and admins), leave requests, and leave statuses in MongoDB.
- Implemented CRUD operations using Mongoose for leave requests, user data, and status updates.

4.2.3 Phase 3: Authentication and Authorization

- ❖ **User Roles:**

- Assigned roles such as Student and Admin, ensuring that only admins have access to leave approval/rejection functionalities.
- Role-based access control restricts access to sensitive data and actions based on user roles.

- ❖ **JWT Integration:**

- Implemented JWT (JSON Web Tokens) for secure authentication, enabling students and admins to log in and access their respective pages.
- Tokens are used to authenticate API requests and ensure secure data transmission.

- ❖ **Password Security:**

- Used bcrypt.js to hash user passwords before storing them in the database, ensuring secure password storage and protecting user data from unauthorized access.

4.2.4 Phase 4: Leave Request System

- ❖ **Leave Request Submission:**

- Integrated a leave request system where students can submit requests for leave, specifying the leave type, dates, and reason.

- The system stores the leave request and updates the leave status once approved or rejected by the admin.

❖ **Leave Status Update:**

- Admins can approve or reject leave requests via the admin dashboard.
- Once a decision is made, an email notification is automatically sent to the student to inform them about the approval or rejection.

4.2.5 Phase 5: Admin Module

❖ **Admin Dashboard:**

- Created an admin dashboard that allows admins to view and manage all leave requests.
- Features include searching for specific leave requests, filtering by status, and approving or rejecting requests.
- Admins can also view a detailed history of all leave requests made by students.

❖ **Leave Management:**

- Admins can manage users (students and admins) by adding, updating, or removing users from the system.
- Admins have the ability to track leave requests, review patterns, and ensure transparency in the leave approval process.

4.3 Features

4.3.1 Features for Students

❖ **Student Dashboard:**

- Submit leave requests and view leave status.
- Track the status of pending leave requests.
- Edit or cancel leave requests before approval.

❖ **Leave Management:**

- Submit requests with leave type, dates, and reason.
- Receive notifications when the leave request is approved or rejected.
- View history of previous leave requests and their statuses.

❖ **Profile Customization:**

- Students can update their personal details (name, email, contact info).
- Modify password and manage account security settings.

4.3.2 Features for Admins

❖ **Admin Dashboard:**

- View all leave requests submitted by students.
- Approve or reject leave requests and update their status.
- Filter requests based on status, dates, or student names.
- View a summary of leave requests and usage statistics.

❖ **User Management:**

- Manage user roles (Student, Admin) and permissions.
- Add, update, or remove users from the system.
- Monitor student accounts and resolve any issues.

❖ **Leave Request Management:**

- Track and manage all leave requests, including approvals, rejections, and pending requests.
- Maintain records of leave history for reporting purposes.

❖ **Reporting Tools:**

- Generate reports such as the number of leave requests submitted, approvals, rejections, and student leave trends.
- Monitor platform usage and identify any irregularities in leave requests.

4.3.3 Platform-Wide Features

❖ **Responsive Design:**

The platform is fully responsive and accessible on desktops, tablets, and mobile devices.

❖ **Role-Based Access Control:**

Ensures students and admins only have access to the features relevant to their roles.

❖ **Secure Authentication and Authorization:**

JWT (JSON Web Tokens) is used for secure authentication of users and admins, ensuring only authorized access.

❖ **Scalable Architecture:**

Designed to handle increased traffic and data as the user base and leave requests grow.

❖ **Real-Time Updates:**

Automatic updates for leave request statuses, such as approval, rejection, or modification.

❖ **Intuitive User Interface:**

Modern, user-friendly design with easy navigation for both students and admins.

❖ **Notifications:**

Email and in-app notifications for leave request updates, approvals, and rejections, ensuring that students are kept informed in real-time.

❖ **Accessibility Features:**

The platform adheres to WCAG (Web Content Accessibility Guidelines) to support users with disabilities, ensuring an inclusive user experience.

4.4 Challenges

4.4.1 Ensuring Responsive Design:

Creating a consistent user experience across multiple devices, including desktops, tablets, and smartphones.

❖ Solution:

- Used React components and CSS Flexbox to ensure a responsive layout.
- Incorporated Tailwind CSS utilities for responsive design, adapting elements for different screen sizes.
- Conducted cross-device and browser testing to ensure the interface adapts smoothly on various platforms.

4.4.2 Managing Role-Based Access Control:

Differentiating access levels for students and admins while ensuring secure authentication and authorization.

❖ Solution:

- Implemented JSON Web Tokens (JWT) for secure user authentication.
- Used Express.js middleware to manage role-based access control, restricting access to sensitive features (e.g., leave approval) based on user roles.

4.4.3 Handling Real-Time Data Updates:

Ensuring that updates to leave requests (approvals, rejections, or modifications) are reflected in real-time across the platform.

❖ Solution:

- Used React state management (e.g., `useState` and `useEffect`) to instantly update the frontend when data changes.

- Implemented API polling to refresh leave request statuses periodically, ensuring the system always shows the latest information.

4.4.4 Integrating Secure Authentication and Authorization:

Protecting sensitive data such as student details and leave request information while ensuring secure access.

❖ Solution:

- Integrated JWT-based authentication to protect access to APIs and user data.
- Used bcrypt.js to hash passwords and store them securely in the database.
- Enforced HTTPS to ensure encrypted communication between the frontend and backend.

4.4.5 Managing Leave Requests and Statuses in the Database:

Efficiently managing a growing number of leave requests, user data, and statuses in the database.

❖ Solution:

- Used MongoDB and Mongoose to design an efficient schema for users, leave requests, and statuses.
- Leveraged indexing and aggregation pipelines to optimize database queries and performance.

4.4.6 Balancing User Interface and Functionality:

Providing a clean and intuitive user interface while maintaining robust functionalities for leave request management.

❖ Solution:

- Followed UI/UX best practices, ensuring easy navigation and clear form inputs.
- Used Tailwind CSS for streamlined, responsive layouts and React Router for seamless page navigation.
- Conducted usability testing to gather feedback from potential users and improve the interface.

4.4.7 Ensuring Data Security:

Protecting user data, including passwords, personal information, and leave request details.

❖ Solution:

- Used bcrypt.js to hash passwords and securely store them in the database.
- Implemented JWT authentication to ensure that data is accessed securely.
- Applied data validation and sanitization to prevent injection attacks.

4.4.8 Deployment and Hosting:

Ensuring smooth deployment and hosting of both frontend and backend while maintaining system scalability and availability.

❖ Solution:

- Deployed the frontend using Netlify to ensure fast load times and continuous delivery.
- Hosted the backend on Heroku with a MongoDB database on MongoDB Atlas for scalability.
- Configured environment variables securely for storing sensitive API keys and credentials.

4.4.9 Synchronizing Data Across Modules:

Maintaining consistency between leave requests, student profiles, and status updates across different modules.

❖ **Solution:**

- Designed RESTful APIs to update data after each operation (e.g., approving or rejecting a leave request).
- Implemented error-handling to ensure that data inconsistencies are caught and resolved immediately, ensuring system integrity.

4.4.10 Handling Leave Request Workflow:

Designing a flexible leave request approval process while ensuring the system remains user-friendly.

❖ **Solution:**

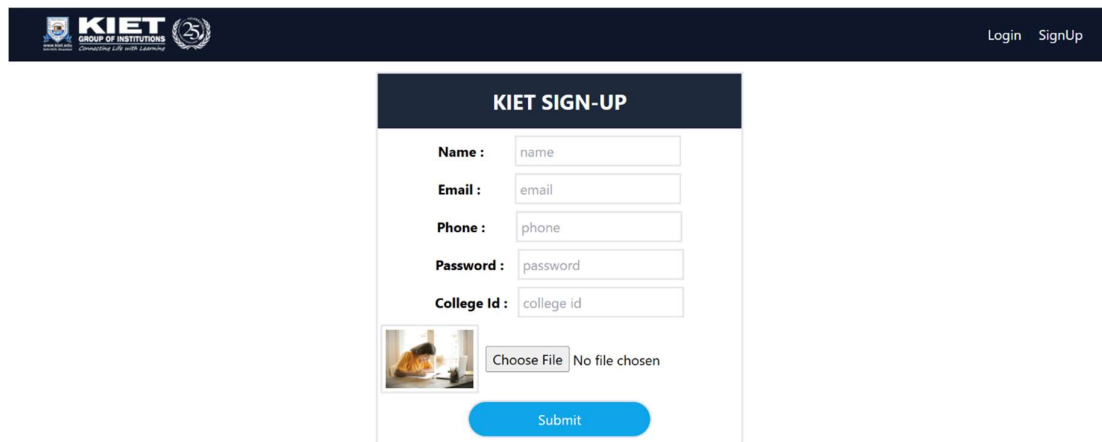
- Created a multi-step leave request workflow with statuses such as *Pending*, *Approved*, and *Rejected*.
- Allowed admins to add comments or reasons for rejecting leave requests, providing transparency to students.
- Included notifications to keep students informed about their request status.

5. Project Outcome

The user interface is designed with simplicity and functionality in mind.

Key pages include:

- **Homepage:** Provides an overview of the system with navigation links.
- **Leave Application Form:** Allows students to fill out and submit leave requests.
- **Admin Dashboard:** Displays pending and processed leave requests for review.



The screenshot shows a web interface for KIET (Knowledge Institute of Technology) sign-up. At the top, there is a dark blue header with the KIET logo and the text 'GROUP OF INSTITUTIONS' and 'CONNECTING LIVES WITH LEARNING'. To the right of the header are links for 'Login' and 'SignUp'. Below the header is a white box titled 'KIET SIGN-UP'. Inside this box, there are five input fields: 'Name :', 'Email :', 'Phone :', 'Password :', and 'College Id :'. Each field has a placeholder text (name, email, phone, password, college id). Below the input fields is a file upload section with a small image of a person working at a desk, a 'Choose File' button, and the text 'No file chosen'. At the bottom of the sign-up box is a blue 'Submit' button.

Figure – 5.1

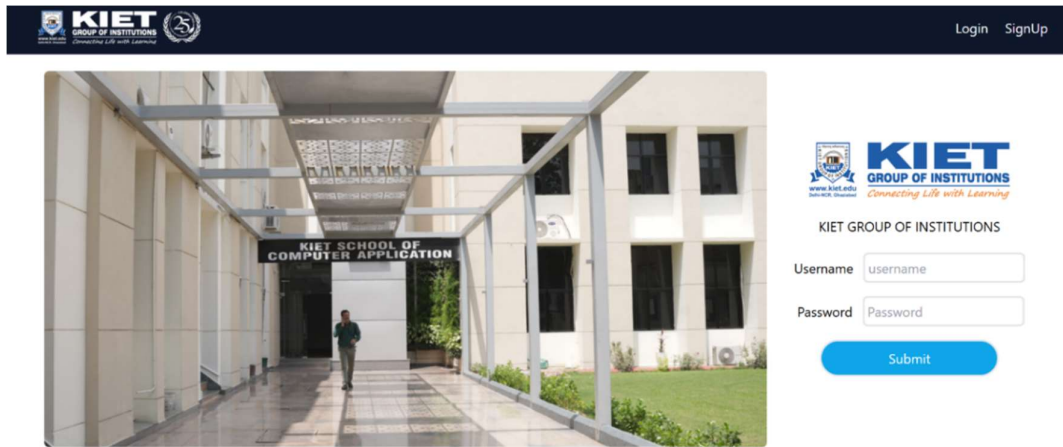


Figure – 5.2

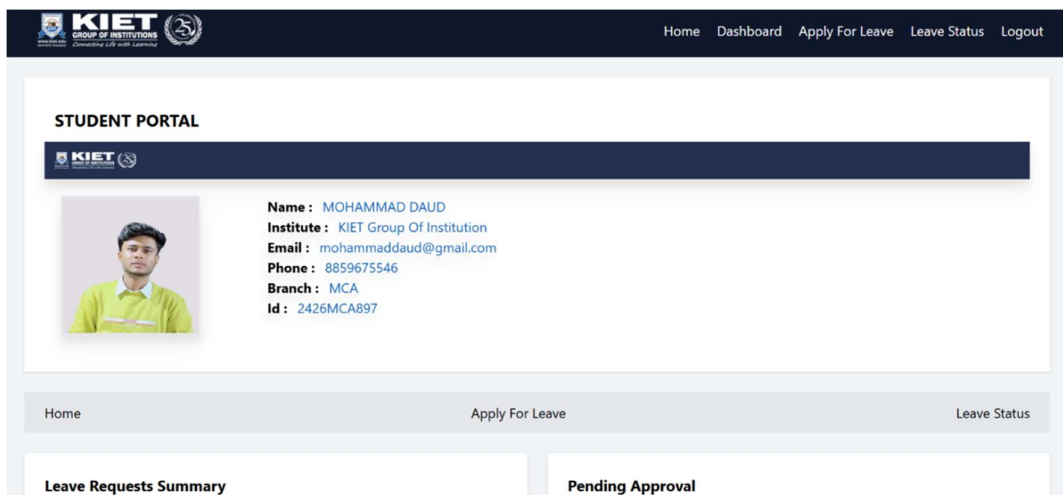



Figure – 5.3

Leave Application Form

Name <input type="text" value="Enter your name"/>	Student ID <input type="text" value="Enter your ID"/>
Department <input type="text" value="Enter your department"/>	Type of Leave <input type="text" value="Select Leave Type"/>
Start Date <input type="text" value="Select start date"/>	End Date <input type="text" value="Select end date"/>
Total Days <input type="text" value="0"/>	
Reason for Leave <input type="text" value="Describe the reason for leave"/>	
<input type="button" value="Submit"/>	

Figure - 5.4


KIET
GROUP OF INSTITUTIONS
Creating Life with Learning

[Home](#)
[Dashboard](#)
[Students Status](#)
[Logout](#)

Leave Status

Name: Mohammad Daud

College ID: 2426MCA869

Dept: MCA

Type of Leave: Casual Leave


Start Date: 12/11/2024

End Date: 12/13/2024

Total Days: 2

Reason: Casual Leave

Status: Pending



Update Status

Figure - 5.5

Update Leave Status

Student Details

Name: Mohammad Daud

College ID: 2426MCA869

Department: MCA


Type of Leave: Casual Leave

Start Date: 12/11/2024

End Date: 12/13/2024

Total Days: 2

Reason: Casual Leave



Select Status:

Pending

Selected Status: Pending

Update Status

Figure - 5.6

5.1. Testing

Test Cases

Test Case ID	Description	Expected Outcome	Result
TC01	Student submits a leave request	Request recorded in the database	Pass
TC02	Admin approves leave request	Email notification sent	Pass
TC03	Invalid login attempt	Error message displayed	Pass

5.2 Functional Testing

5.2.1 Unit Testing

Unit testing is conducted to verify that individual components or functions of the *Leave Approval System* work correctly in isolation. For example:

- **Form Submission:** Testing the form to ensure required fields (like start date, end date, and reason) are validated properly.
- **Email Notification Logic:** Ensuring that the email notification function triggers correctly after approval or rejection.
- **Approval Workflow:** Verifying that the system updates leave status accurately (Pending → Approved/Rejected).

Tools like Jest can be used for React components to test specific features such as form validation, button click events, and state management.

5.2.2 Integration Testing

Integration testing ensures that the individual modules of the system interact correctly. In the *Leave Approval System*, integration testing validates:

- The successful data flow from the leave submission form to the backend/database.
- The approval/rejection of leave requests and corresponding updates in the database.
- The automatic triggering of email notifications when the leave status changes.

For instance, when an admin approves a leave request, integration testing verifies that:

1. The database records are updated.
2. The frontend reflects the updated leave status.
3. An email is sent to the student.

5.2.3 System Testing

Functional testing checks whether the *Leave Approval System* works according to the project requirements. The main functionalities to be tested include:

- Student Leave Request Submission: Ensuring the form accepts valid data and displays appropriate error messages for invalid or missing inputs.
- Admin Approval/Rejection: Verifying that the admin can approve or reject leave requests and the system updates the status accordingly.
- Email Notifications: Testing that students receive accurate notifications after approval/rejection.
- Login Functionality: Ensuring users can log in securely with valid credentials and receive errors for invalid login attempts.

6. Conclusion

The *Leave Approval System* successfully automates the process of leave application, approval, and notification within an academic environment. By developing this system, the manual and time-consuming tasks of managing leave requests have been streamlined into a user-friendly and efficient digital solution. The system reduces paperwork, minimizes human errors, and ensures that leave records are managed seamlessly. It provides an intuitive interface for students to submit leave requests and for administrators to review and approve or reject them. The integration of email notifications ensures timely communication with students regarding the status of their applications.

Through the use of modern technologies like React and Tailwind CSS for the frontend and a structured backend for workflow management, the system demonstrates efficiency, reliability, and scalability. It improves transparency by allowing students to track the status of their requests in real time, fostering better communication between students and administrators. The *Leave Approval System* addresses the key challenges of leave management in educational institutions, providing a foundation for future enhancements, such as advanced reporting, analytics, and mobile accessibility.

The *Leave Approval System* can be enhanced by integrating advanced features such as mobile application development for easier accessibility, role-based access control for improved security, and real-time analytics for better insights.

7. References

- React Documentation. <https://reactjs.org>
- Tailwind CSS Documentation. <https://tailwindcss.com>
- Node.js Documentation. <https://nodejs.org>
- MongoDB Documentation. <https://www.mongodb.com>