RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

CS19442 SOFTWARE ENGINEERING CONCEPTS LAB

Laboratory Record
Note Book

Name :……. ……...

Year / Branch / Section : …….. …….

University Register No. : …….. ……

College Roll No. : …….……

Semester : …….……

Academic Year : …….

RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

Name:_____

Academic Year: _____ Semester: _____ Branch:_____

Register No: | | | | | | | | | | | | |

Certified that this is the bonafide record of work done by the above student in the CS19442-Software Engineering Concepts Laboratory during the year 2023- 2024

Signature of Faculty-in-charge

Submitted for the Practical Examination held on ----------

Internal Examiner External Examiner

ONLINE LEAVE MANAGEMENT SYSTEM

INDEX

OVERVIEW

The "Online Leave Management System" is a web-based application designed to streamline and automate the process of leave requests and approvals within an educational institution or organization. This system provides a centralized platform for students, class incharges, and heads of departments (HODs) to manage leave-related activities efficiently.

Functionality Abstract:

1. **User Authentication and Management:**
    - Users can register and log in to the system using unique credentials.
    - The system supports different user roles, including students, class incharges, and HODs.
2. **Leave Request Submission:**
    - Students can submit leave requests specifying the start date, end date, and reason for the leave.
    - Leave requests are validated based on predefined criteria such as attendance percentage.
3. **Leave Approval Workflow:**
    - Class incharges are responsible for approving or rejecting leave requests submitted by students.
    - If a student's attendance percentage falls below a certain threshold, the leave request is forwarded to the HOD for approval.
4. **Profile Management:**
    - Users can manage their profiles, including updating personal information such as email address and password.
5. **Leave Request Tracking:**
    - Students can track the status of their leave requests, including whether they have been approved, rejected, or are pending approval.
6. **Historical Leave Data:**
    - Students can view their past leave requests and their corresponding approval status.

7. **Class Incharge Functions:**
    - Class incharges have additional functionalities such as managing leave requests of students under their supervision and viewing attendance records.
8. **Head of Department Functions:**
    - HODs have access to a broader range of functionalities, including approving leave requests forwarded by class incharges and monitoring overall leave management within their department.

SOFTWARE REQUIREMENT SPECIFICATION

EXP NO:1: DATE: 20-02-24

Table of Contents

Introduction

1. Problem Statement

The current process for managing student leave requests is primarily paper-based, relying on physical forms and manual approvals. This manual approach presents several challenges:

- **Inefficiency:** Filling out forms, collecting signatures, and manually processing requests are time-consuming for both students and Class In-Charges.
- **Delays and Errors:** Delays occur due to the need for physical submissions, approvals, and potentially lost paperwork. Additionally, the manual process is prone to human error in data entry or approval decisions.
- **Lack of Transparency:** Students have limited visibility into the status of their leave requests, leading to frustration and unnecessary inquiries.
- **Data Management Challenges:** Maintaining paper records is cumbersome, making it difficult to track trends, generate reports, or analyze leave patterns.

1.1 Purpose

Efficiency and Convenience:

- **Streamlined application process:** Students can submit leave requests electronically, eliminating paper forms and long queues.
- **Real-time tracking:** Students can easily track the status of their leave applications and receive prompt notifications regarding approvals or rejections.
- **Accessibility:** The system allows students to request leave from anywhere with an internet connection, improving accessibility and convenience.

Transparency and Communication:

- **Clear leave policies:** The system can present clear and up-to-date information about leave policies, types of leave available, and limitations.
- **Improved communication:** Both students and faculty can access a centralized platform for communication regarding leave requests and approvals, avoiding confusion or miscommunication.

Record Keeping and Management:

- **Accurate records:** The system maintains a digital record of all leave requests and approvals, ensuring accuracy and transparency.
- **Data analysis:** Institutions can analyze leave data to identify trends and patterns, which can then inform policy decisions and resource allocation.

Reduced administrative burden: The system automates many aspects of leave management, freeing up valuable time for Class In-Charges and faculty. Overall, an online leave management system can significantly improve the efficiency, transparency, and management of student leave requests within a college setting.

1.2 Product Scope

The product is a software application which is accessed from a web browser and used Online system for requesting, tracking, and managing college student leave.

1.3 Overview

This project aims to develop an online leave management system for college students. The system will allow students to electronically submit leave requests, track their status, and view their remaining leave balance. It will also provide a platform for faculty and Class In-Charges to approve or reject leave requests, manage leave policies, and generate reports on student leave patterns. This system aims to improve efficiency, transparency, and communication in the leave management process for both students and college administration.

2. General Description

This project envisions an online leave management system designed specifically for college students. The system will operate in three distinct stages:

Stage 1: Student Leave Request: Students can access the Student-friendly platform using their college credentials. The system will offer a clear interface for submitting leave requests. Students can choose from predefined leave categories (e.g., medical, personal, academic) and specify details like reason for leave, duration, and supporting documents (if applicable). The system will automatically display their remaining leave balance for each category, ensuring responsible and informed decision-making.

Stage 2: Administrative Review and Approval: Once submitted, the leave request enters the review stage. Class In-Charges, typically Class In-Charges or designated staff, can access the request through a secure dashboard. They can review the student's details, leave type, supporting documents, and verify their eligibility against predefined criteria (e.g., attendance record, previous leave history). Based on this review, they can either **approve** the request, granting the student leave, or **reject** it for valid reasons.

Stage 3: Department Head Intervention and Emergency Leave: In special cases, the system caters to emergencies and situations requiring additional authority. For **emergency leaves**, students can submit their request through a designated emergency leave category, often requiring a brief explanation and immediate attention. These requests bypass the regular review stage and directly reach the **Department Head**, typically a higher-level Class In-Charge, Dean, or Provost. They can then assess the urgency, approve or reject the request, or potentially forward it back to the relevant Class In-Charge for further investigation and potential approval. This ensures prompt response to unforeseen situations while maintaining a structured leave management process.

This three-stage approach aims to streamline the leave management process, empower students with self-service capabilities, and ensure responsible leave handling by providing clear guidelines and appropriate levels of approval within the college administration.

.

2.1 Product Perspective

From a product perspective, this online leave management system offers a Student-friendly and centralized platform for both students and college administration to manage student leave requests efficiently.

Students benefit from a streamlined process: submitting requests electronically, tracking their status in real-time, and viewing their remaining leave balance promotes transparency and removes the hassle of paper

forms. Class In-Charges gain a centralized system for reviewing requests, managing policies, and generating reports, leading to improved decision-making and reduced administrative burden. Overall, the system fosters clear communication, eliminates confusion, and ensures responsible leave management for a more productive and well-organized college environment.

2.1.1 System interfaces

The system interface caters to three distinct Student groups - students, Class In-Charges, and the Department Head - offering a unique experience for each. Students will have a Student-friendly dashboard to submit leave requests, check history, and view their remaining balance. Class In-Charges will access a dedicated portal to review pending requests, manage policies, and generate reports. The superadmin dashboard prioritizes emergency requests, allowing them to directly approve, reject, or delegate further investigation. Each interface emphasizes clear communication and Student-friendliness, ensuring a smooth and efficient leave management experience for all stakeholders

2.1.2 Student interfaces

Target Students:

- Students
- Class In-Charges (Staff)
- Department Head (Class In-Charges)

2.1.3 Interface Components:

1. **Student Dashboard:**
    - Login using college credentials.

- **Leave Request Panel:**
    - Select leave type (sick leave, personal leave, etc.).
    - Enter specific dates and duration of leave.
    - Provide a clear and concise reason for leave.
    - Option to upload supporting documents (doctor's note, travel itinerary).
    - View remaining leave balance for each category.
    - Submit button with confirmation message upon successful submission.

- **Leave History:**
    - View a chronological list of past leave requests.
    - See status of each request (approved, rejected, pending).
    - Access detailed information for each request.

2. Class In-Charge Dashboard:

- Login using designated credentials.

- **Pending Leave Requests:**
    - A list of all pending leave requests from students within their department.
    - Each request shows student details, leave type, dates, reason, and uploaded documents.
    - Option to approve, reject, or request additional information from the student.
    - Option to forward the request to the Department Head for emergency or special cases.

- **Approved/Rejected Requests:**
    - Access and manage records of previously approved or rejected requests.
    - Option to filter by student, date range, or leave type.

3. Department Head Dashboard:

- Login using designated credentials.

- **Emergency Leave Requests:**
    - Dedicated section displaying all emergency leave requests submitted by students.
    - Similar information as regular requests with emphasis on urgency and reason.
    - Option to directly approve, reject, or re-route the request to a specific Class In-Charge for further investigation and potential approval.
    - View and manage existing leave policies for different categories.
    - Ability to update or modify policies with appropriate permissions.

2.2 Product Functions

This online leave management system empowers both students and the college administration with a centralized and efficient platform for managing student leave requests. Students benefit from a streamlined process, allowing them to submit requests electronically, track their status in real-time, and view their remaining leave balance through a Student-friendly interface. This eliminates the need for paper forms and fosters transparency. Class In-Charges gain a central hub for reviewing requests, managing leave policies, and generating data-driven reports, leading to improved decision-making and reduced administrative burden. Overall, the system promotes clear communication, eliminates confusion, and ensures responsible leave management for a more productive and well-organized college environment. This unique combination of features empowers both students and Class In-Charges, streamlining the leave management process and fostering a more efficient and transparent college environment

2.2.1 Enterprise Requirements

This online leave management system caters to several enterprise requirements within the college environment:

Compliance and Governance:

- **Policy Management:** The system facilitates the creation, maintenance, and distribution of clear and consistent leave policies aligned with college regulations.
- **Audit Trail and Reporting:** A comprehensive audit trail tracks all leave requests and actions, ensuring accountability and compliance with internal policies and external regulations. Additionally, the system generates reports on leave trends and usage patterns, enabling informed decision-making regarding future policy adjustments.

Security and Data Management:

- **Secure Access Control:** Student roles and permissions ensure that only authorized individuals can access sensitive leave information. Secure login protocols and data encryption safeguard the confidentiality of student data and comply with data privacy regulations.

Workflow Automation and Efficiency:

- **Automated Notifications:** The system sends automated notifications to students and Class In-Charges regarding the status of leave requests, eliminating manual communication and promoting transparency.

- **Streamlined Approval Process:** The system automates the routing of leave requests to appropriate Class In-Charges based on pre-defined criteria, expediting the approval process and reducing administrative workloads.

Scalability and Integration:

- **Scalable Architecture:** The system is designed to accommodate the growing needs of the college, allowing for seamless integration with existing student information systems or future applications.

Overall, this online leave management system addresses various enterprise requirements by fostering compliance, enhancing data security, automating workflows, and promoting efficiency within the college environment.

2.3 Student Characteristics

This college student online leave management system caters to three distinct Student groups, each with their own unique characteristics:

1. Students:

- **Tech-savvy:** Comfortable using online platforms and familiar with contemporary Student interface design.
- **Time-conscious:** Juggle various academic and personal commitments, valuing quick and easy access to the system.
- **Privacy-conscious:** Concerned about the security and confidentiality of their personal information.

2. Class In-Charges (Staff):

- **Organized and detail-oriented:** Responsible for reviewing and managing student leave requests in accordance with college policies.
- **Efficient and time-managed:** Juggle multiple administrative tasks, requiring a system that facilitates quick processing of leave requests.
- **Data-driven:** Appreciate access to reports and analytics to understand leave trends and make informed decisions.

3. Department Head (Class In-Charges):

- **Strategic and decision-making:** Responsible for overseeing the overall leave management process and making crucial decisions concerning leave policies and exceptional circumstances.
- **Time-sensitive:** Requires a clear and concise overview of critical information, particularly regarding emergency leave requests.
- **Security-focused:** Prioritizes data security and ensures compliance with relevant regulations.

Understanding these Student characteristics is crucial for designing a Student-friendly and effective system that caters to the specific needs and expectations of each Student group.

2.4 Constraints

The college student online leave management system faces several potential constraints:

Technical Constraints:

- **Integration Challenges:** Integrating the system with existing college information systems like student databases or calendars can be complex and require technical expertise.
- **Data Security and Privacy:** Ensuring robust data security measures and complying with data privacy regulations like FERPA (Family Educational Rights and Privacy Act) are crucial considerations.

- **Accessibility:** The system needs to be accessible to Students with disabilities, adhering to accessibility standards and guidelines.

Organizational Constraints:

- **Change Management:** Implementing a new system can lead to resistance from Students accustomed to existing processes. Careful planning, communication, and training are essential to ensure smooth adoption.
- **Policy Alignment:** The system needs to be designed in alignment with existing college leave policies and regulations to avoid confusion and ensure compliance.
- **Resource Constraints:** Developing and maintaining the system requires ongoing resources, including personnel, budget, and technical expertise.

Student Constraints:

- **Technical Literacy:** Not all Students may possess the same level of technical literacy. The system needs to be Student-friendly and intuitive to accommodate varying skill levels.
- **Internet Connectivity:** Reliable internet access is essential for Students to access and utilize the system effectively.
- **Time Constraints:** Students, Class In-Charges, and the Department Head may have limited time to learn and use the new system. The system should be designed for efficiency and minimize additional time burdens.

By effectively addressing these constraints, the project can achieve its goals and deliver a valuable solution for both students and the college administration.

2.5 Assumptions and Dependencies

This college student online leave management system relies on several assumptions and dependencies:

Assumptions:

- **Standardized Leave Policies:** The college has established and clearly defined leave policies for different categories (e.g., medical, personal, academic) with clearly outlined eligibility criteria and approval processes.
- **Reliable Internet Access:** All Students (students, Class In-Charges, Department Head) have reliable internet access to utilize the system effectively.
- **Technical Expertise:** The college has the necessary technical expertise to implement and maintain the system, or has access to external resources for development and support.
- **Student Acceptance:** Students, Class In-Charges, and the Department Head are receptive to adopting the new system and are willing to undergo training.

Dependencies:

- **Existing Infrastructure:** The system's functionality may depend on successful integration with existing college information systems like student databases or calendars.
- **Data Security Measures:** The system relies on robust data security measures and protocols to safeguard sensitive student information, complying with relevant data privacy regulations.
- **Student Training and Support:** Effective Student training and ongoing support are crucial for successful system adoption and efficient utilization by all Student groups.
- **Clear Communication and Change Management:** Implementing the new system requires clear communication strategies and effective change management processes to address potential resistance and ensure a smooth transition from existing practices.

It's important to acknowledge and address these assumptions and dependencies throughout the project lifecycle to ensure successful development, implementation, and Student adoption of the online leave management system.

2.6 Apportioning of Requirements

Developing a successful online leave management system requires careful consideration of various factors and stakeholders. To ensure a well-rounded solution, here's a proposed apportionment of requirements:

1. **Functional Requirements(60%):**

- **Core Functionalities(40%)**

  - Student roles and permissions for students, Class In-Charges, and Department Head.
  - Secure login and data encryption.
  - Student ability to submit leave requests (various types with supporting documents).
  - Tracking and viewing leave request status.
  - Class In-Charge ability to review, approve, or reject requests.
  - Department Head ability to manage emergency leave and exceptional cases.
  - Generating reports on leave trends and usage patterns.

- **Advanced functionalities (20%):**
  - Integration with existing college information systems (optional).
  - Automated notifications for students and Class In-Charges regarding leave status updates.
  - Feedback mechanism for Students to provide comments or suggestions.
  - Accessibility features for Students with disabilities.

2. Non-Functional Requirements (20%)

- **Usability (40%):**
  - Student-friendly interface for all Student groups.
  - Intuitive navigation and clear instructions.
  - Responsive design for various devices (desktop, mobile, tablets).
- **Performance (30%):**
  - Fast loading times and responsiveness.
  - Scalability to accommodate a growing Student base.
  - System availability and uptime.
- **Security (30%):**
  - Secure Student authentication and authorization.
  - Data encryption and protection of sensitive information.
  - Compliance with relevant data privacy regulations.

3. Student Requirements (10%)

- **Needs and expectations of different Student groups:**
  - Students: ease of use, real-time status updates, privacy.
  - Class In-Charges: efficiency, data-driven insights, clear leave history.
  - Department Head: timely decision-making, emergency leave management, compliance.
- **Training and support resources for Students.**

4. Technical Requirements (10%)

- **Programming languages and frameworks suitable for the chosen architecture.**
- **Hardware and software infrastructure requirements for deployment.**
- **Integration protocols with existing systems (if applicable).**

This apportionment is a starting point and can be adjusted based on the specific needs and priorities of the college implementing the system. By allocating appropriate resources and addressing these diverse requirements, the project can deliver a valuable and efficient online leave management solution for the college community.

3. Specific Requirements

3.1 External Interfaces

This system interacts with several external entities, requiring well-defined interfaces to ensure smooth data exchange and functionality. Here's a breakdown of the key external interfaces:

1. Student Interface (UI):

- **Primary interface:** This is the web-based interface accessed by different Student groups (students, Class In-Charges, Department Head) to interact with the system. It handles Student login, navigation, displaying information, and performing actions like submitting leave requests, reviewing requests, generating reports, etc.

2. Database Interface:

- The system interacts with a database to store and manage student information, leave request details, leave policies, and other relevant data. The specific database interface depends on the chosen database management system (DBMS).

3. Authentication and Authorization Interface:

- This interface handles Student login, authentication, and authorization. It verifies Student credentials and grants access based on assigned roles and permissions within the system. This interface might interact with existing college authentication systems or utilize its own secure login mechanism.

4. (Optional) Integration Interface:

- This interface facilitates potential integration with existing college information systems like student databases or calendars. The specific protocols and data exchange formats depend on the target systems and their existing APIs (Application Programming Interfaces).

5. Reporting Interface:

- This interface allows the system to generate reports in PDF formats for different Student groups. The interface may involve external reporting tools or libraries depending on the chosen reporting solution.

6. Administrative Interface:

- (Optional) This interface might be relevant if system administration tasks require external tools or services. For example, backup and recovery procedures may involve interaction with separate backup software or cloud storage services.

Security Considerations:

- All external interfaces should be designed with robust security measures in place. This includes secure communication protocols (HTTPS), data encryption, and proper authorization controls to prevent unauthorized access and data breaches.

By carefully defining and managing these external interfaces, the college student online leave management

system can ensure efficient and secure data exchange with various external entities, contributing to the overall success of the project.

3.1 Functions

System functional requirements are specified by use cases and specific requirements. The use case helps understand system behavior, and the specific requirements extend the information from the use case.

3.1.1 Use Case: 1 Leave Request from student

Submitting a Leave Request:

1. **Login:** Students access the system using their college credentials (e.g., student ID and password).
2. **Leave Request Form:** A Student-friendly interface allows students to submit leave requests.
3. **Leave Details:** Students select the type of leave (e.g., medical, personal, academic) and specify the dates and duration of their absence.
4. **Reason and Documentation:** They can provide a concise explanation for their leave and upload supporting documents if required (e.g., doctor's note, travel itinerary).
5. **Leave Balance:** The system displays the student's remaining leave balance for each category, ensuring they stay within their allotted leave allowance.
6. **Submission:** Once all details are entered, students can submit the leave request electronically for review.

Tracking and Managing Requests:

- Students can access a dedicated section to view a list of their submitted leave requests.
- This list displays the status of each request (pending, approved, rejected) in real-time.
- Students can click on each request to view detailed information, including the reason for leave, submitted documents, and any comments from the Class In-Charge.

3.1.2 Use Case: 2 Approval or Reject or Forward Leave Request by Class In-Charges

Class In-Charges will have secure login credentials to access a dedicated web-based dashboard. This dashboard will provide a centralized view of all leave requests submitted by students within their designated department or area.

Reviewing Leave Requests:

- The Class In-Charge can view a list of all pending leave requests from students. Each request will present clear information, including:
    - Student details (name, ID)
    - Leave type (medical, personal, academic)
    - Requested dates and duration of leave
    - Reason for leave (optional)
    - Attached documents (if provided, e.g., doctor's note, travel itinerary)

Decision-Making and Action:

- Based on the information provided and pre-defined institutional policies (e.g., attendance

threshold, maximum leave allowance), the Class In-Charge can take the following actions:

- **Approve:** Grant the student's leave request.
- **Reject:** Decline the request with a reason and any specific instructions.
- **Request More Information:** Contact the student for additional details or clarifications before making a decision.
-

Additional Features:

- The Class In-Charge's dashboard might offer additional functionalities for better management:
  - **Search and Filter Requests:** Ability to search and filter requests by student name, leave type, date range, etc., for easier management of large workloads.
  - **Leave Policy Management:** Access to view and update existing leave policies or departmental guidelines (optional, might require higher-level permission).
  - **Leave History:** Ability to view past leave requests submitted by students within their department for reference and future decision-making.

1. **Accepting a Leave Request:**
   a. Once a student submits a leave request, it lands on the Class In-Charge's dashboard for review.
   b. The Class In-Charge can access the details of the request, including the student's information, leave type, dates, reason, and any uploaded documents (e.g., doctor's note).
   c. After reviewing the request and verifying the information against college leave policies and student eligibility criteria (e.g., attendance records, previous leave history), the Class In-Charge can click the "Approve" button.
   d. The system will automatically update the student's leave balance for the specific category and notify the student via email or in-app notification that their leave request has been approved.
2. **Declining a Leave Request:**
   a. In cases where the Class In-Charge cannot approve a leave request due to policy violations, insufficient information, or other reasons, they can choose the "Reject" option.
   b. The system will prompt the Class In-Charge to enter a reason for rejection (optional). This explanation can be helpful for the student to understand why their request was denied and potentially avoid similar issues in the future.
   c. The student will receive a notification informing them about the rejection and the reason provided by the Class In-Charge.
3. **Forwarding a Leave Request to the Department Head:**
   a. In specific situations, the Class In-Charge might encounter a leave request that requires a higher level of authorization or falls outside of their typical approval criteria. These situations might involve:
      i. **Emergency leave:** Requests marked as emergency by the student due to unforeseen circumstances.
      ii. **Complex cases:** Requests with unclear justifications or that require additional investigation before approval.
      iii. **Exceeding leave limits:** Requests exceeding the student's remaining leave balance for a specific category.

3.2.3 Use Case: 3 Approval or Reject and Manage Student and Class In-Charges information by Department Head

The online leave management system empowers the Department Head to manage exceptional leave requests

and oversee the overall leave approval process. Here's how the Department Head can handle situations involving requests forwarded by Class In-Charges and student appeals:

Admin-Forwarded Requests:

- **Functionality:** Class In-Charges can encounter student leave requests that fall outside their normal approval authority due to various reasons (e.g., complex medical cases, extenuating circumstances). The system provides the Class In-Charge with the option to **forward the request** to the Department Head for further review and decision-making.

AGILE SCRUM METHODOLOGY

EXP NO:2 DATE:12-03-24

Product Backlog:

Sprint 1:

Duration: 2 weeks

Sprint Goal: Implement basic leave request submission and view leave history functionality.

User Stories:

1. Student Leave Request Submission
2. View Leave History

Tasks:

1. Design database schema for leave requests and history.
2. Create UI forms for leave request submission and leave history display.
3. Implement backend logic for leave request submission and storage.
4. Implement backend logic for fetching and displaying leave history.

Sprint Review: Demonstrate functionality for leave request submission and leave history display.

Sprint 2:

Duration: 2 weeks

Sprint Goal: Implement leave request review for class in charge and emergency leave handling for department heads.

User Stories:

3. Class In Charge Leave Request Review
4. Department Head Emergency Leave Handling

Tasks:

1. Design database schema for pending leave requests and emergency leave handling.
2. Create UI for class in charge to review pending leave requests.
3. Implement backend logic for class in charge to approve, reject, or request additional information for leave requests.
4. Create UI for department heads to manage emergency leave requests.
5. Implement backend logic for department heads to handle emergency leave requests.
6. Set up notifications for department heads on new emergency leave requests.

Sprint Review: Demonstrate functionality for leave request review by class in charge and emergency leave handling by department heads.

Sprint 3:

Duration: 2 weeks

Sprint Goal: Implement leave policy management for department heads.

User Story: 5. Leave Policy Management

Tasks:

1. Design database schema for leave policies.
2. Create UI for department heads to view and manage leave policies.
3. Implement backend logic for department heads to update leave policies.
4. Log changes made to leave policies for audit purposes.
5. Set up permissions for who can view or modify leave policies.

Sprint Review: Demonstrate functionality for leave policy management by department heads.

This breakdown aligns with Agile principles by focusing on iterative development, delivering working functionality in short sprints, and incorporating feedback at each sprint review. Adjustments can be made based on team capacity and project priorities.

USER STORIES

EXP NO:3 DATE:19-03-24

User Story 1: Student Leave Request Submission

As a student,
I want to submit a leave request with specific details such as leave type, dates, reason, and supporting documents,
so that I can formally notify the college and get approval for my absence.

Acceptance Criteria:

1. The student can select the type of leave from a predefined list (e.g., sick leave, personal leave).
2. The student can enter the start and end dates for the leave.
3. The student can provide a reason for the leave in a text field.
4. The student can upload supporting documents (e.g., doctor's note, travel itinerary).
5. The system displays a confirmation message upon successful submission of the leave request.
6. The student can view the submitted leave request in their leave history.

User Story 2: View Leave History

As a student,
I want to view a chronological list of my past leave requests and their statuses,
so that I can track the history and outcomes of my leave applications.

Acceptance Criteria:

1. The student can access a leave history page from their dashboard.
2. The leave history page displays a list of past leave requests with details such as date, leave type, reason, and status (approved, rejected, pending).
3. The student can click on a specific leave request to view detailed information and any uploaded documents.
4. The system should display the status of each leave request prominently.

User Story 3: Class In Charge Leave Request Review

As a class in charge,
I want to review pending leave requests, including details such as student information, leave type, dates, reason, and supporting documents,
so that I can make informed decisions to approve, reject, or request additional information.

Acceptance Criteria:

1. The class in charge can access a list of all pending leave requests from their dashboard.
2. Each leave request displays student details, leave type, dates, reason, and any uploaded documents.
3. The class in charge can approve, reject, or request additional information for each leave request.
4. The system records and updates the status of the leave request based on the class in charge's decision.

5. The class in charge can forward the request to the department head for special or emergency cases.

User Story 4: Department Head Emergency Leave Handling

As a department head,
I want to review and manage emergency leave requests submitted by students,
so that I can prioritize and handle urgent cases appropriately, ensuring quick resolution.

Acceptance Criteria:

1. The department head can access a dedicated section for emergency leave requests from their dashboard.
2. The emergency leave section displays requests with details such as student information, leave type, dates, reason, and any uploaded documents.
3. The department head can directly approve, reject, or re-route the request to a specific class in charge for further investigation.
4. The system displays the urgency and reason for the leave prominently.
5. The department head receives notifications for new emergency leave requests.

User Story 5: Leave Policy Management

As a department head,
I want to view, update, and manage leave policies for different categories,
so that I can ensure the leave policies are up-to-date and reflect the institution's regulations and needs.

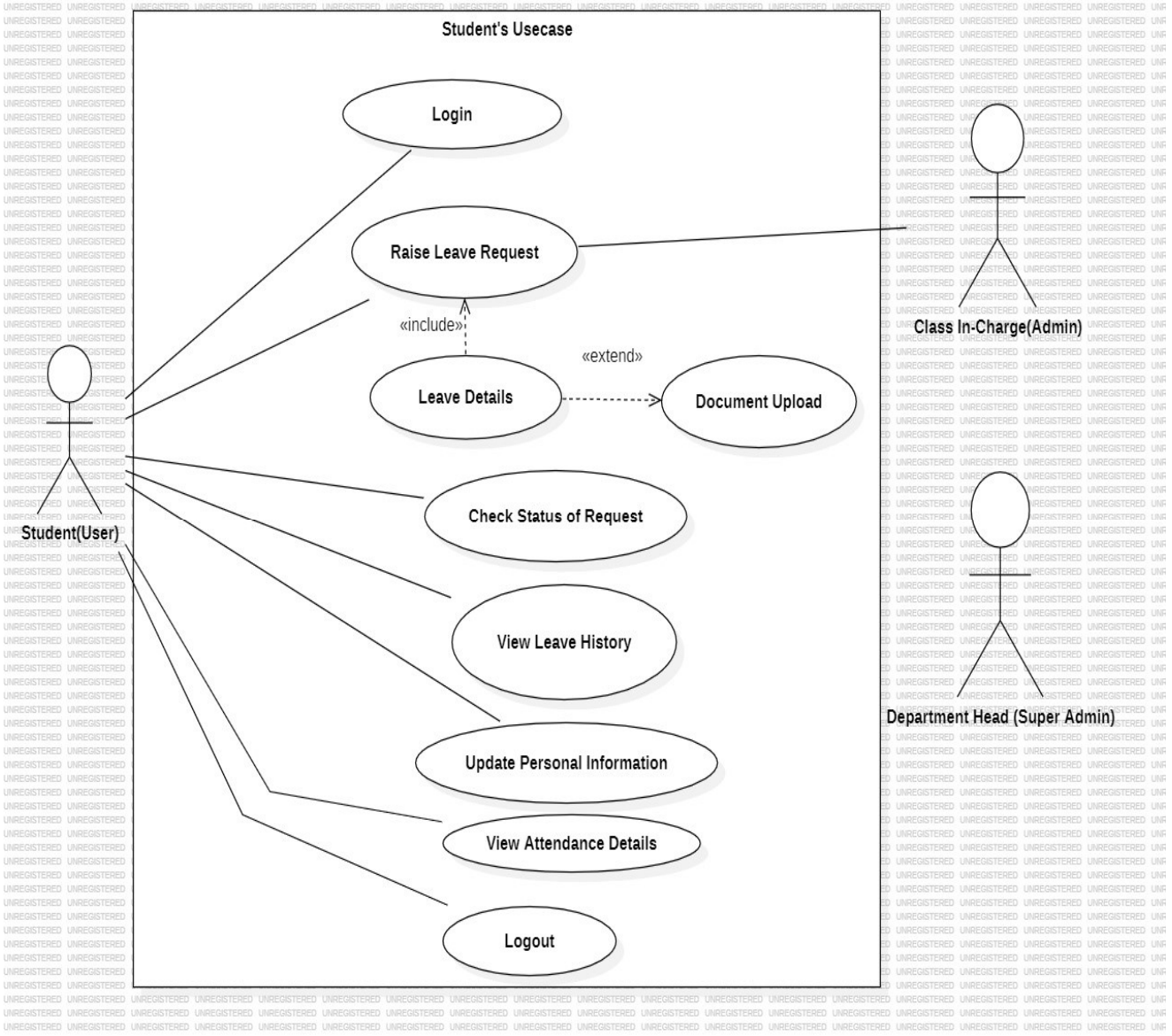
Acceptance Criteria:

1. The department head can access a leave policy management section from their dashboard.
2. The leave policy management section displays existing leave policies categorized by type.
3. The department head can update or modify leave policies, including rules and limits for each leave type.
4. The system logs all changes made to the leave policies for audit purposes.
5. The department head can set permissions for who can view or modify the leave policies.

USE CASE DIAGRAM

EXP NO:4 DATE:19-03-24

Student's Use Case Diagram:



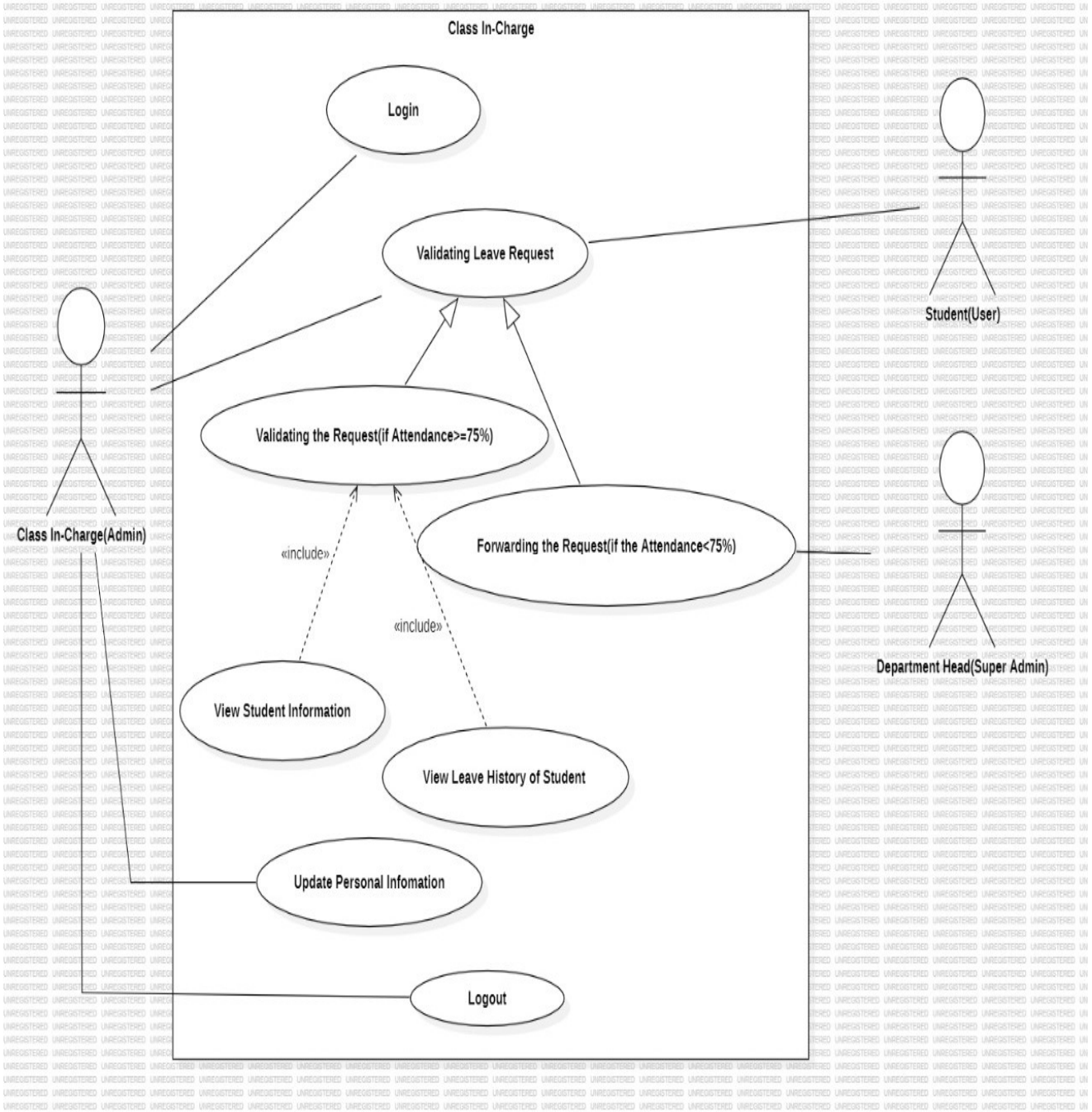
1. Student's Use Case:

- **Login**: Students log in to the system using their credentials.
- **Raise Leave Request:**
  - Students can request leave by providing necessary details (like leave type, dates, reason, etc.).
  - This use case has two extensions:
    - Document Upload: Students can attach supporting documents (e.g., medical certificates) if required.
    - Leave Details: Students can view additional information related to their leave request.
- **Check Status of Request:** Students can track the status of their leave requests.
- **View Leave History:** Students can see their past leave requests and approvals.
- **Update Personal Information:** Students can modify their personal details (e.g., contact info).
- **View Attendance Details**: Students can access attendance records.
- **Logout:** Students log out of the system.

2. Additional Actors:

- **Class In-Charge/Admin:**
  - Responsible for approving or rejecting leave requests.
  - Connected to the "Raise Leave Request" and "Check Status of Request" use cases.
- **Department Head/Super Admin:**
  - Has higher privileges.
  - Connected to the "Raise Leave Request" and "Check Status of Request" use cases.

Class In-Charge Use Case Diagram:



1. Class In-Charge's Use Cases:

a. **Login:** The class in-charge logs in to the system using their credentials.

b. Validating Leave Request:

i. The class in-charge reviews leave requests submitted by students.

ii. This use case has two extensions:

1. Forwarding the Request (if Attendance > 75%): If a student's attendance is above 75%, the request is forwarded for further processing.

2. Forwarding the Request (if Attendance < 75%): If a student's attendance is below 75%, the request is handled differently.

c. **View Student Information:** The class in-charge can access student details.

d. **View Leave History of Student:** The class in-charge can see a student's past leave requests and approvals.

e. **Update Personal Information:** The class in-charge can modify their own personal details.

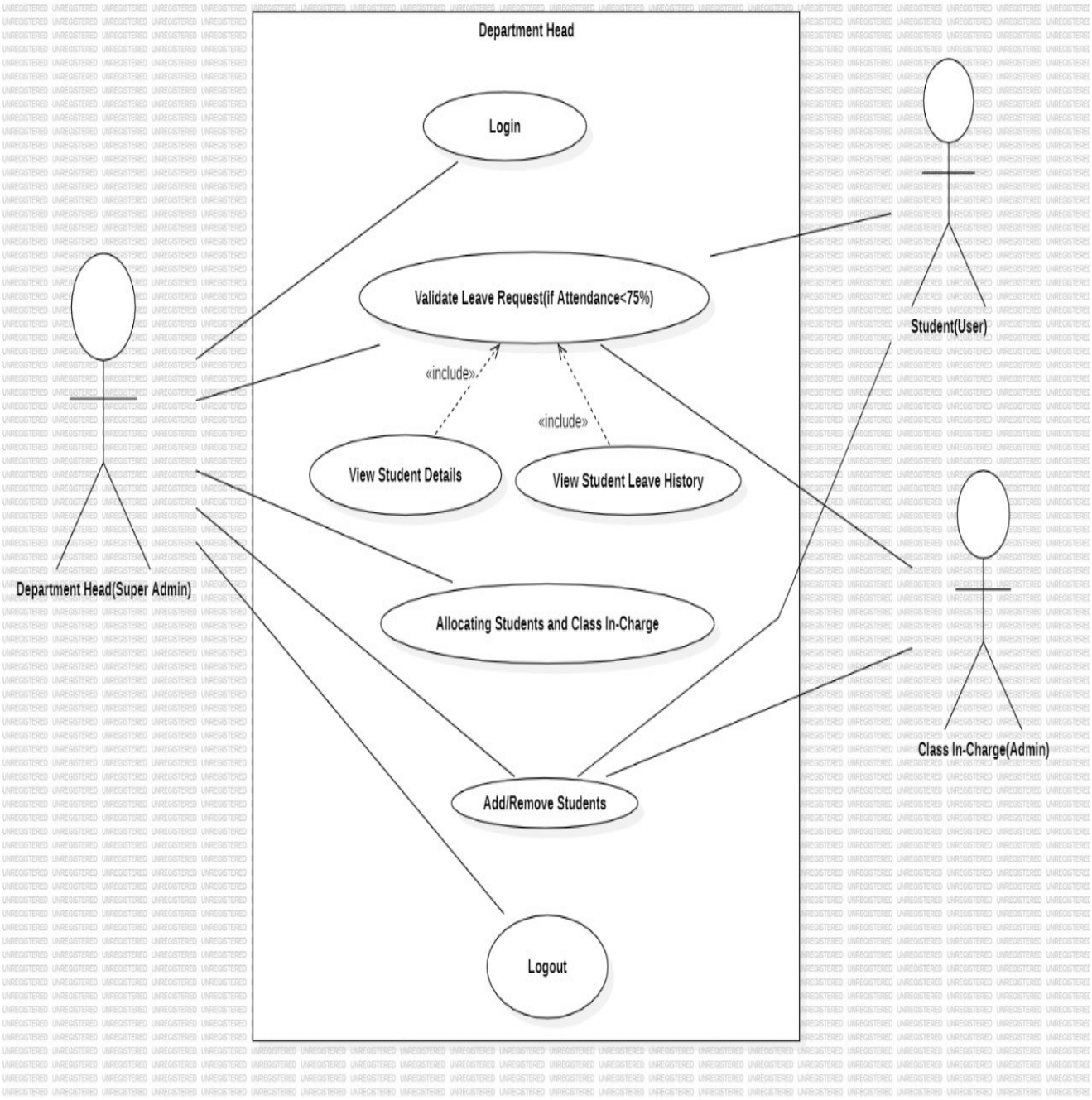
f. **Logout:** The class in-charge logs out of the system.

2. Other Actors:

I. Student (User): **The student submits leave requests.**

II. Department Head (Super Admin): **Has higher privileges and is connected to the "Validating Leave Request" use case.**

Department Head Use Case Diagram:



1. Department Head's Use Cases:

a. **Login:** The department head logs in to the system using their credentials.

b. Validate Leave Request (if Attendance < 75%):

i. The department head reviews leave requests submitted by students.

ii. If a student's attendance is below 75%, the request is handled differently.

c. **View Student Details:** The department head can access student information.

d. **View Student Leave History:** The department head can see a student's past leave requests and approvals.

e. Allocating Students and Class In-Charge:

i. After viewing student details, the department head allocates students to specific class in-charges.

f. **Add/Remove Students**: The department head manages student records.

g. **Logout:** The department head logs out of the system.

2. Other Actors:

a. Department Head Support Admin:

i. This actor is connected to the "Login" use case.

b. Student (User):

i. The student submits leave requests.

ii. Connected to the "Login" use case and an extended use case from the department head's "Validate Leave Request."

c. Class In-Charge (Admin):

i. Responsible for handling leave requests.

ii. Connected to the "Class In-Charge (Admin)" use case.

NFR

EXP NO:5 DATE:29-03-24

Performance

1. **Response Time:** The system should respond to user actions within 2 seconds under normal load conditions.
2. **Scalability:** The system should be able to handle up to 10,000 concurrent users without degradation in performance.
3. **Availability:** The system should be available 99.9% of the time, ensuring minimal downtime.
4. **Data Processing:** Leave requests and updates should be processed within 1 second of submission.

Security

1. **Authentication**: All users must authenticate using secure college credentials.
2. **Authorization:** Role-based access control (RBAC) should be implemented to ensure only authorized users can access specific features (students, administrators, super admins).
3. **Data Encryption:** Sensitive data, including login credentials and uploaded documents, should be encrypted both in transit and at rest.
4. **Audit Logs:** The system should maintain audit logs of all user activities, especially related to leave request submissions, approvals, and rejections.
5. **Session Management:** Secure session management should be enforced, including timeout policies to log out inactive users after 15 minutes of inactivity.

Usability

1. **User Interface:** The UI should be intuitive and user-friendly, allowing users to navigate easily through different sections.
2. **Accessibility:** The system should comply with accessibility standards (e.g., WCAG 2.1) to ensure it is usable by people with disabilities.
3. **Responsiveness:** The system should be responsive and work seamlessly on various devices, including desktops, tablets, and smartphones.
4. **Error Handling:** Clear and concise error messages should be provided to guide users in case of incorrect inputs or system errors.

Reliability

1. **Fault Tolerance:** The system should handle failures gracefully, ensuring data integrity and consistent state.
2. **Backup and Recovery:** Regular backups should be taken, and a recovery plan should be in place to restore data in case of failures.

Maintainability

1. **Code Quality:** The codebase should follow industry best practices for readability, modularity, and documentation.
2. **Configurability:** System configurations (e.g., leave types, policies) should be easily manageable through an admin interface without requiring code changes.
3. **Logging and Monitoring:** The system should include comprehensive logging and monitoring to detect and troubleshoot issues promptly.

Interoperability

1. **Integration:** The system should be able to integrate with existing college systems, such as the student information system (SIS) and human resources (HR) system.
2. **Data Export:** Provide options to export data in standard formats (CSV, Excel) for reporting and analysis.

Legal and Compliance

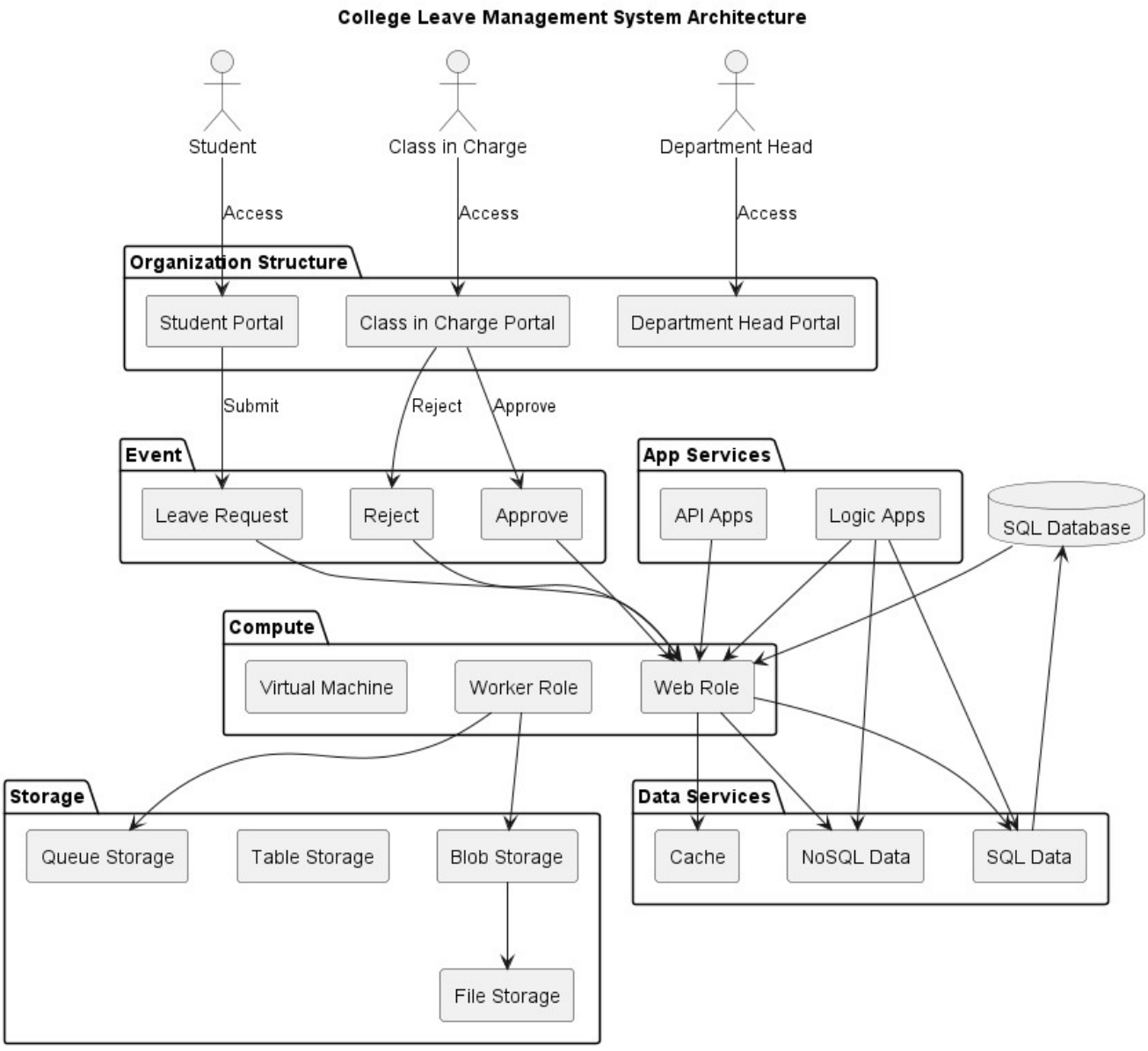
1. **Data Protection:** The system must comply with relevant data protection regulations (e.g., GDPR, FERPA) to ensure the privacy and security of student data.
2. **Record Retention:** Leave request records should be retained according to college policies and legal requirements.

Efficiency

1. **Resource Utilization:** The system should optimize resource usage, including CPU, memory, and network bandwidth, to ensure efficient operation.
2. **Power Consumption:** For any hardware components, power consumption should be minimized, especially for systems hosted in data centers.

OVERALL PROJECT ARCHITECTURE

EXP NO:6 DATE:09-04-24



```
@startuml
skinparam componentStyle rectangle
title College Leave Management System Architecture

actor Student
actor "Class in Charge" as ClassInCharge
actor "Department Head" as DepartmentHead

package "Event" {
    component "Leave Request"
    component "Approve"
    component "Reject"
}

package "Compute" {
    component "Web Role" as WebRole
    component "Worker Role" as WorkerRole
    component "Virtual Machine" as VirtualMachine
}

package "Data Services" {
    component "SQL Data" as SQLData
    component "NoSQL Data" as NoSQLData
    component "Cache"
}

package "Storage" {
    component "Blob Storage" as BlobStorage
    component "Table Storage" as TableStorage
    component "Queue Storage" as QueueStorage
    component "File Storage" as FileStorage
}

package "App Services" {
    component "API Apps" as APIApps
    component "Logic Apps" as LogicApps
}

package "Organization Structure" {
    component "Student Portal" as StudentPortal
    component "Class in Charge Portal" as ClassInChargePortal
    component "Department Head Portal" as DepartmentHeadPortal
}

database "SQL Database" as SQLDatabase

Student --> StudentPortal: Access
ClassInCharge --> ClassInChargePortal: Access
DepartmentHead --> DepartmentHeadPortal: Access
```

```
StudentPortal --> "Leave Request": Submit
ClassInChargePortal --> "Approve": Approve
ClassInChargePortal --> "Reject": Reject

"Leave Request" --> WebRole
"Approve" --> WebRole
"Reject" --> WebRole

WebRole --> SQLData
WebRole --> NoSQLData
WebRole --> Cache

WorkerRole --> QueueStorage
WorkerRole --> BlobStorage

APIApps --> WebRole
LogicApps --> WebRole
LogicApps --> SQLData
LogicApps --> NoSQLData

SQLData --> SQLDatabase
SQLDatabase --> WebRole

BlobStorage --> FileStorage

@enduml
```

Overall Structure

The diagram represents a high-level view of the College Leave Management System architecture, showing how different components interact with each other and with the users. It includes actors, events, compute resources, data services, storage options, app services, and the organization structure.

Actors

- **Student**: Represents the students who will use the system to submit leave requests.
- **Class in Charge**: Represents faculty members who review and approve or reject leave requests.
- **Department Head**: Represents the higher authority who can oversee leave requests and manage escalations.

Components and Packages

1. **Event Package**
    - **Leave Request**: The event triggered when a student submits a leave request.
    - **Approve**: The event triggered when a Class in Charge approves a leave request.
    - **Reject**: The event triggered when a Class in Charge rejects a leave request.
2. **Compute Package**

- **Web Role**: Manages the web interface and handles HTTP requests from the Student Portal, Class in Charge Portal, and Department Head Portal.
- **Worker Role**: Performs background processing tasks like sending notifications and handling bulk data processing.
- **Virtual Machine**: Provides dedicated computing resources for tasks requiring specific environments or configurations.

3. **Data Services Package**
    - **SQL Data**: Stores structured data such as student details, leave requests, and approval status.
    - **NoSQL Data**: Stores unstructured or semi-structured data such as logs and session information.
    - **Cache**: Provides fast access to frequently accessed data to improve system performance.
4. **Storage Package**
    - **Blob Storage**: Stores large binary objects such as documents or images submitted with leave requests.
    - **Table Storage**: Stores structured datasets that don't require complex querying, such as logs.
    - **Queue Storage**: Manages asynchronous message queuing between system components.
    - **File Storage**: Provides shared storage accessible by multiple components for configuration files and logs.
5. **App Services Package**
    - **API Apps**: Provide endpoints for various functionalities, enabling integration with other systems.
    - **Logic Apps**: Automate workflows such as the leave request process, including approval and rejection.
6. **Organization Structure Package**
    - **Student Portal**: Interface for students to submit leave requests and view their status.
    - **Class in Charge Portal**: Interface for faculty to review, approve, or reject leave requests.
    - **Department Head Portal**: Interface for department heads to oversee and manage leave requests.
7. **SQL Database**
    - Acts as the central repository for storing structured data.

Relationships

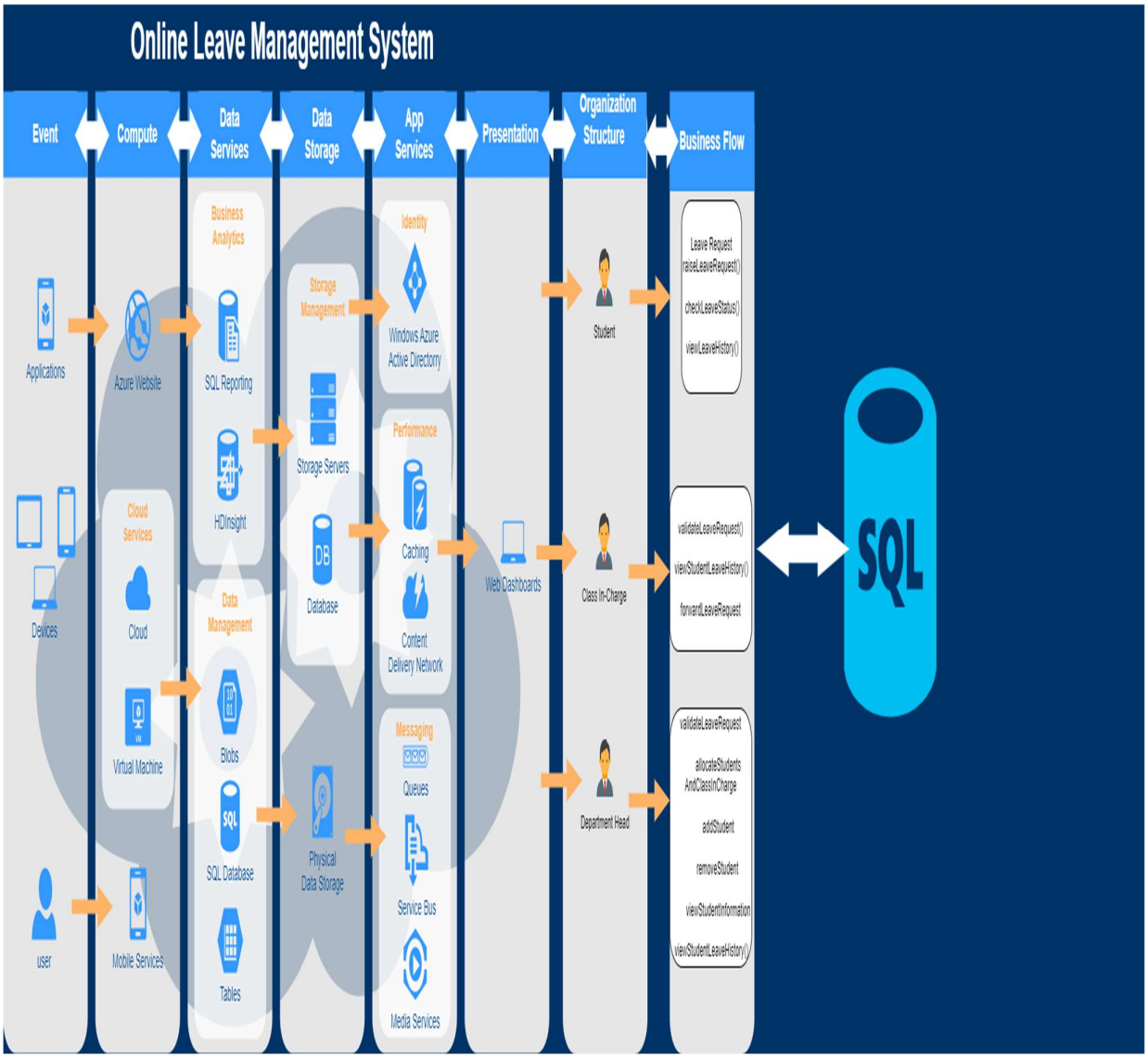
- **User Interactions**:
    - Students interact with the **Student Portal** to submit leave requests.
    - Class in Charge interacts with the **Class in Charge Portal** to approve or reject leave requests.
    - Department Head interacts with the **Department Head Portal** to manage leave requests.
- **Event Handling**:

    - **Leave Request**, **Approve**, and **Reject** events are handled by the **Web Role**.
- **Data Flow**:
    - **Web Role** interacts with **SQL Data**, **NoSQL Data**, and **Cache** to manage and retrieve data.
    - **Worker Role** processes tasks and interacts with **Queue Storage** and **Blob Storage**.
    - **API Apps** and **Logic Apps** interact with the **Web Role** to provide additional functionalities and automate workflows.
    - **SQL Data** is stored in the **SQL Database**, which interacts with the **Web Role**.
- **Storage**:
    - **Blob Storage** interacts with **File Storage** for storing large objects.

BUSINESS ARCHITECTURE

EXP NO:7 DATE:19-04-24

ARCHITECTURAL DIAGRAM



1. Event Column

Leave Request, Approve, Reject:

- **Leave Request:** A student initiates a leave request, which is submitted through the Student Portal.
- **Approve:** The class in charge (faculty) reviews the leave request and can approve it.
- **Reject:** The class in charge can also reject the leave request if it does not meet the criteria.

These events represent the primary actions taken by users within the leave management workflow.

2. Compute Column

Web Role, Worker Role, Virtual Machine:

- **Web Role:** Handles the web interface for students, faculty, and department heads. This includes the portals where students submit leave requests and faculty manage them.
- **Worker Role:** Manages background tasks such as sending email notifications to students and faculty about the status of leave requests, generating reports, and handling periodic data synchronization.
- **Virtual Machine:** Used for tasks that require more control over the environment or for running custom applications that support the leave management system.

These components provide the computational resources necessary to operate the web applications and background services.

3. Data Services Column

SQL Data, NoSQL Data, Cache:

- **SQL Data:** Stores structured data, such as student details, leave request records, approval statuses, and historical data.
- **NoSQL Data:** Stores unstructured data like logs, session information, and possibly other metadata that doesn't fit neatly into relational tables.
- **Cache:** Improves performance by caching frequently accessed data, such as student profiles, class schedules, and recent leave requests, to reduce database load and speed up response times.

These services manage the storage and retrieval of data, ensuring quick access and reliable data management.

4. Storage Column

Blob Storage, Table Storage, Queue Storage, File Storage:

- **Blob Storage:** Used for storing large files like scanned medical certificates, documents, or images submitted by students as part of their leave request.
- **Table Storage:** A NoSQL store for structured datasets that don't require complex querying, such as system logs or user preferences.
- **Queue Storage**: Manages messaging between components for asynchronous processing, such as queuing notifications to be sent to students and faculty.
- **File Storage:** Provides a shared storage solution for configuration files, static resources, or any other necessary files that need to be accessed by multiple components.

These storage solutions provide different ways to store various types of data required by the system.

5. App Services Column

API Apps, Logic Apps:

- **API Apps:** Provide endpoints for different functionalities of the leave management system, enabling integration with other systems or services (e.g., college's main student information system).
- **Logic Apps:** Automate the workflow processes. For example, a logic app could manage the leave request approval process, including sending emails to the class in charge for approval and notifying students of the decision.

These services automate and expose the system's functionality, making it easier to manage and integrate with other systems.

6. Organization Structure Column

Student Portal, Class in Charge Portal, Department Head Portal:

- **Student Portal:** Interface where students can submit leave requests, view their leave balance, and check the status of their requests.
- **Class in Charge Portal**: Interface for faculty to review, approve, or reject leave requests, and to view student leave histories.
- **Department Head Portal:** Interface for department heads to oversee leave requests, manage escalations, and generate reports on leave statistics.

This structure defines user roles and responsibilities, ensuring proper access control and workflow management.

7. Data Flow and SQL Database

Arrows and SQL Database:

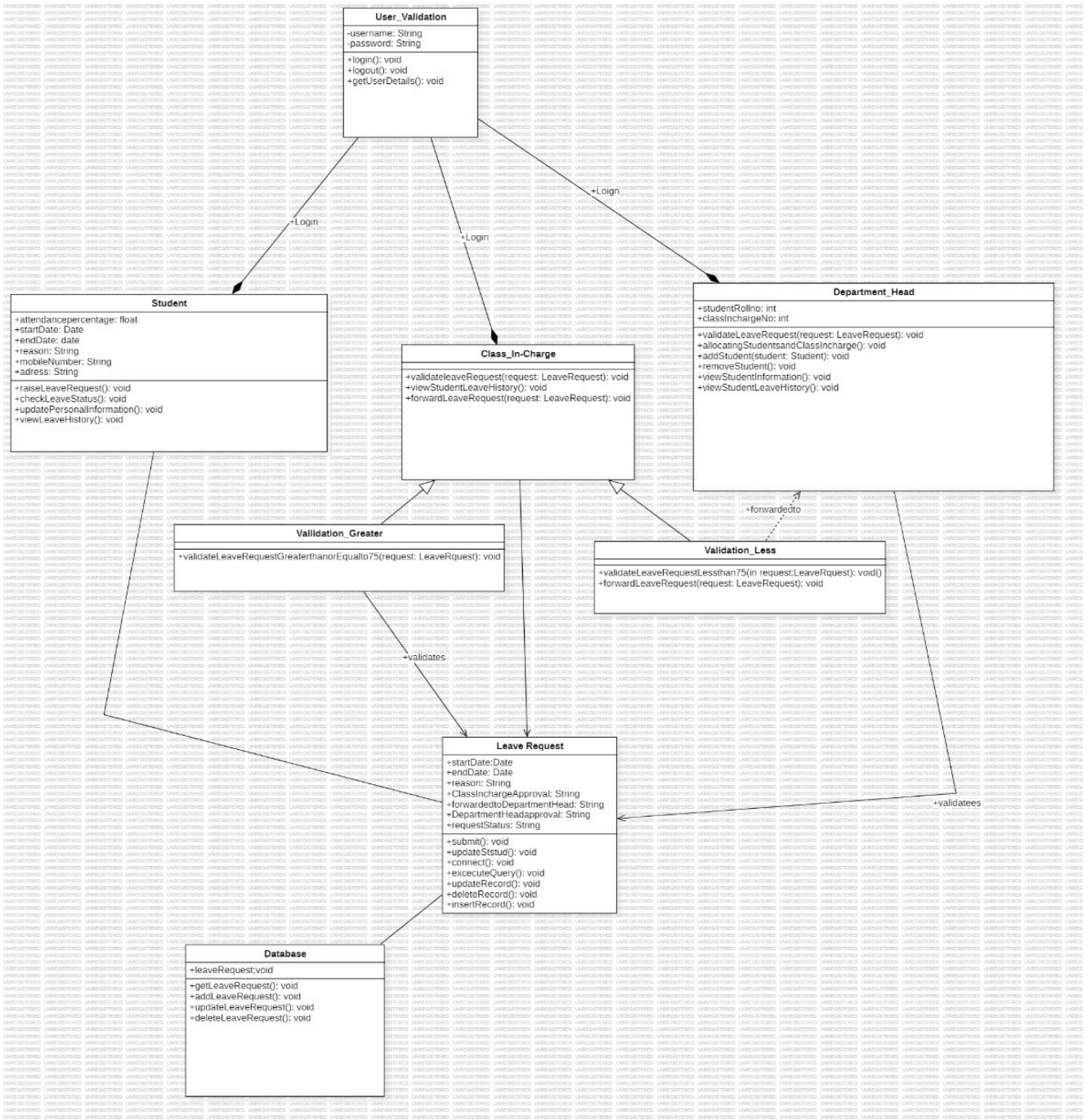
- **Arrows:** Indicate the flow of data between different components, showing how data moves from user actions through the compute resources, data services, and storage, and ultimately being stored in the SQL database.
- **SQL Database:** Acts as the central repository for structured data, ensuring that all leave requests, approvals, rejections, and related data are stored securely and can be queried efficiently.

CLASS DIAGRAM

EXP NO:8 DATE:30-04-24

.

Class Diagram



User Management:

1. **User Class:**
    - **Attributes:** ID, username, password, email, role, creation timestamp.
    - This class represents a generic user in the system. The role attribute distinguishes between different types of users (Student, ClassIncharge, HOD).
2. **Student Class (Extends User Class):**
    - **Additional Attributes:** attendance percentage, class incharge ID.
    - This class represents a student user, inheriting attributes from the User class and adding student-specific information like attendance and the ID of their class incharge.
3. **ClassIncharge Class (Extends User Class):**
    - This class represents a class incharge, inheriting attributes from the User class. It includes attributes specific to their role within the system.
4. **HOD Class (Extends User Class):**
    - Similar to ClassIncharge, this class represents a head of department with attributes specific to their role, inheriting from the User class.

Associations:

- The associations show that a User can be a Student, ClassIncharge, or HOD, indicating the different roles users can have within the system.

Leave Management:

1. **LeaveRequest Class:**
    - **Attributes:** ID, student ID, start date, end date, reason, status, approval status for class incharge and HOD, application timestamp.
    - This class represents a leave request made by a student.
    - It contains information about the request, such as dates, reason, status (pending, approved, rejected), and approval status by both the class incharge and HOD.
    - The student ID attribute establishes a link between a leave request and the student making the request.

Associations:

- The associations indicate that a LeaveRequest is associated with a Student, ClassIncharge, and HOD, reflecting the involvement of these entities in the leave request process. This association allows tracking which student made the request and which authorities are responsible for approving it.

MVC Architecture Overview:

In the Model-View-Controller (MVC) architecture:

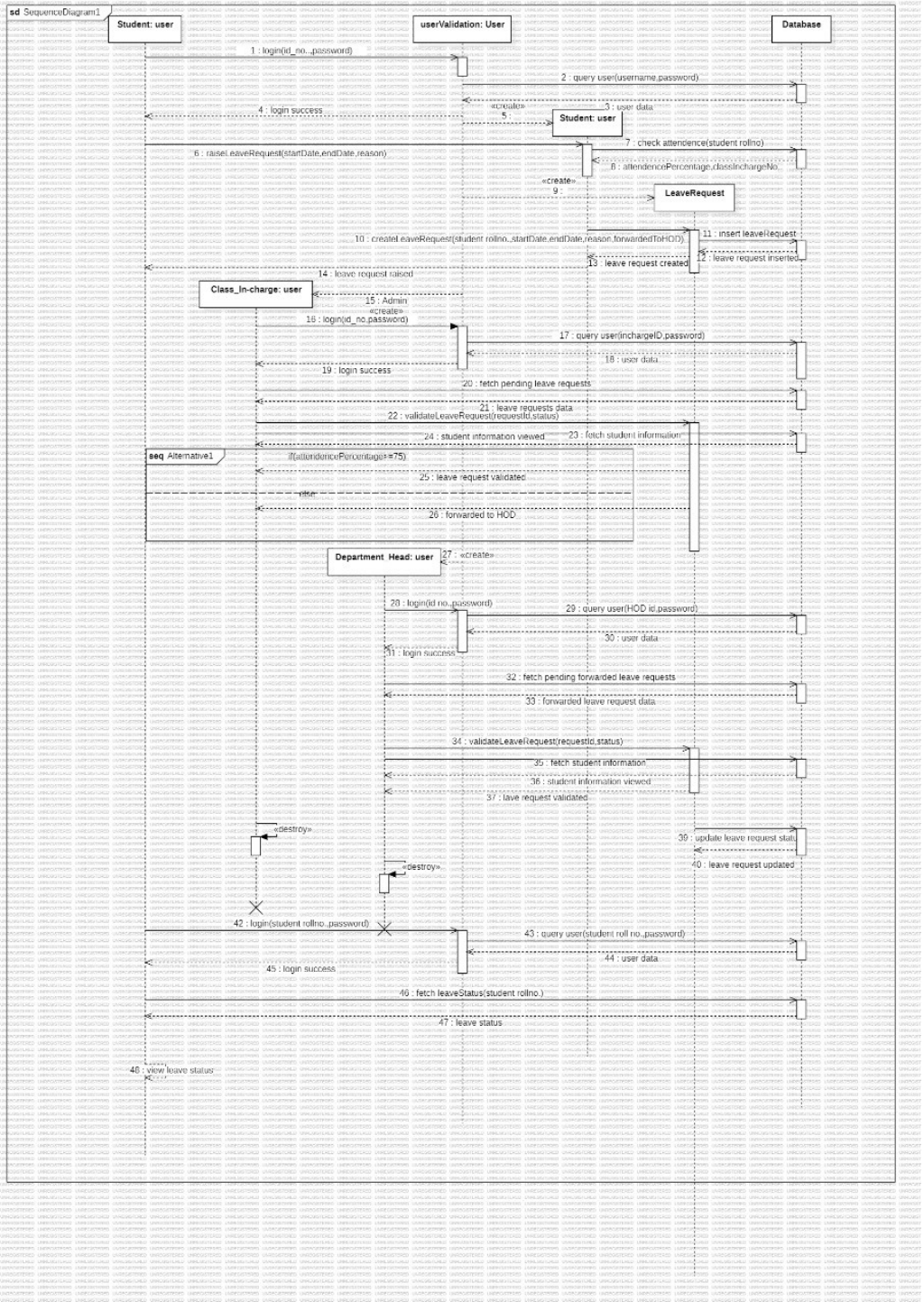
- **Model:** These classes (User, Student, ClassIncharge, HOD, LeaveRequest) represent the model layer. They encapsulate data and business logic related to users and leave management.

- **View:** This layer would involve the user interface components where users interact with the system, such as forms for submitting leave requests and pages for viewing user profiles.
- **Controller:** Controllers handle the flow of data and interactions between the model and view layers. For example, a controller might process a leave request submitted by a student, validate it, and update the corresponding models (e.g., LeaveRequest status, approval status).

SEQUENCE DIAGRAM

EXP NO:9 DATE:10-05-24

SEQUENCE DIAGRAM



Lifelines

- **Student**: Represents the student interacting with the system to submit a leave request.
- **Class in Charge**: Represents the faculty member reviewing and approving/rejecting leave requests.
- **Department Head**: Represents the higher authority overseeing leave requests.
- **Student Portal**: Interface through which the student and Class in Charge interact with the system.
- **LeaveService**: Backend service handling the creation and management of leave requests.
- **ApprovalService**: Backend service managing the approval or rejection of leave requests.
- **SQLDatabase**: The database where leave request data is stored and retrieved.

Messages

- **Submit Leave Request**: A synchronous message where the student submits a leave request through the Student Portal.
- **Create Leave Request**: A synchronous call from the Student Portal to the LeaveService to create a leave request.
- **Save Leave Request**: A synchronous call from LeaveService to the SQLDatabase to save the new leave request.
- **Leave Request Confirmation**: The response sent back to the student confirming the leave request creation.
- **View Leave Requests**: A synchronous call from the Class in Charge to the Student Portal to view pending leave requests.
- **Fetch Leave Requests**: A synchronous call from the Student Portal to the SQLDatabase to retrieve leave request data.
- **Approve/Reject Leave Request**: A synchronous call from the Class in Charge through the Student Portal to the ApprovalService to process the leave request.
- **Update Leave Status**: A synchronous call from the ApprovalService to the SQLDatabase to update the status of the leave request.
- **Confirmation of Approval/Rejection**: The response sent back to the Class in Charge confirming the action taken on the leave request.
- **Notify Student**: An asynchronous call from the ApprovalService to the Student Portal to notify the student of the leave request's approval/rejection status.
- **Leave Request Approved/Rejected**: The asynchronous notification sent to the student informing them of the decision.

Activations

- **StudentPortal, LeaveService, SQLDatabase, ApprovalService**: These are activated when they are processing a specific request. For example, when the StudentPortal is creating a leave request, its lifeline is activated, showing it is actively performing an action.

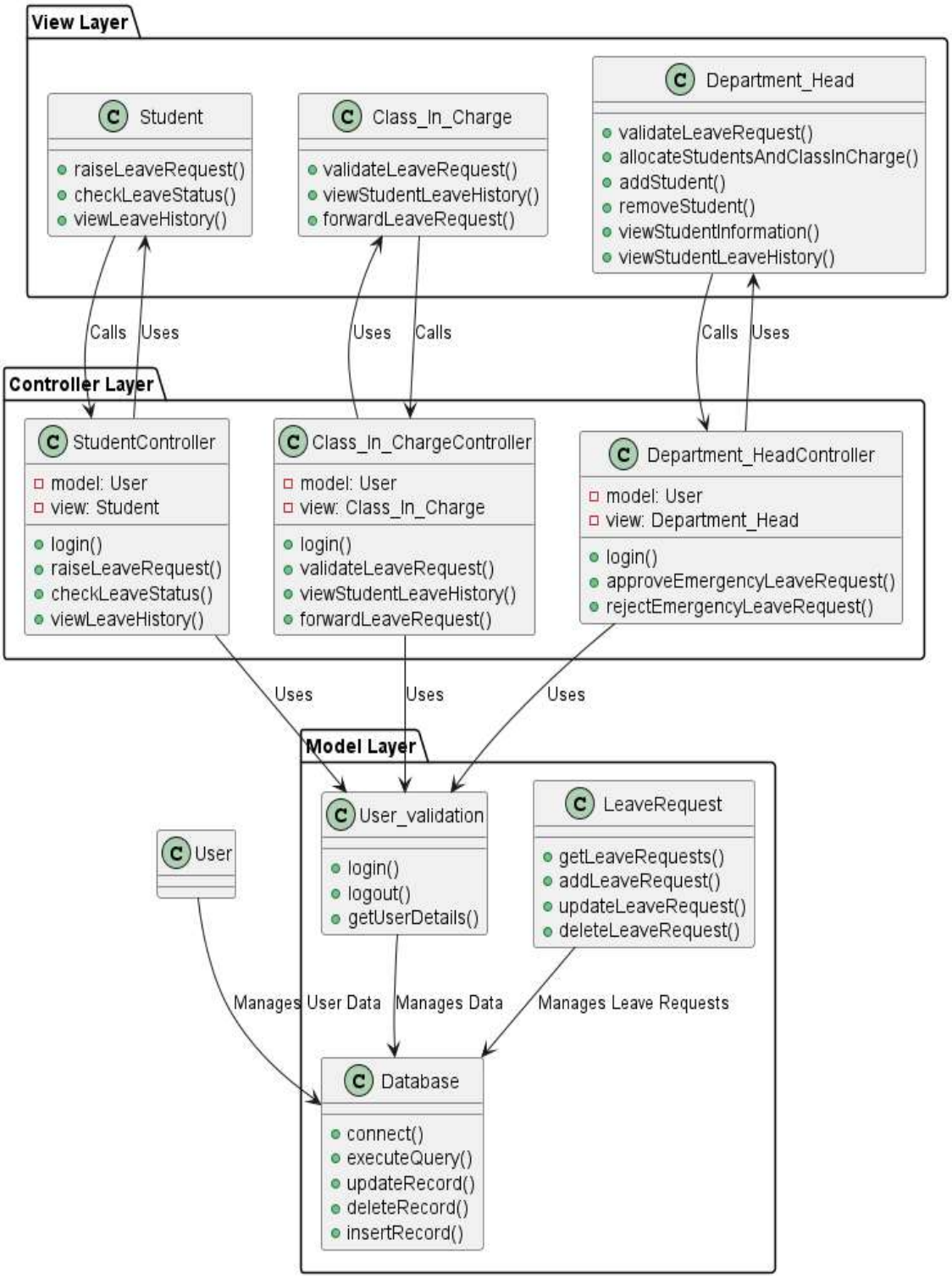
Synchronous and Asynchronous Calls

- **Synchronous Calls**: The system waits for a response before proceeding. For example, when **LeaveService** calls **SQLDatabase** to save a leave request, it waits for a confirmation before sending a response back to the **StudentPortal**.
- **Asynchronous Calls**: The system does not wait for a response and can continue processing other tasks. For instance, the **ApprovalService** sends an asynchronous notification to the **StudentPortal** to inform the student of the approval/rejection, and the **ApprovalService** can continue processing other tasks without waiting.

ARCHITECTURAL PATTERN

EXP NO:10 DATE:17-05-24

ARCHITECTURE



PLANTUML CODE

```
@startuml

' Define the components of the MVC architecture
package "Model Layer" {
    class Database {
        + connect()
        + executeQuery()
        + updateRecord()
        + deleteRecord()
        + insertRecord()
    }

    class LeaveRequest {
        + getLeaveRequests()
        + addLeaveRequest()
        + updateLeaveRequest()
        + deleteLeaveRequest()
    }

    class User_validation {
        + login()
        + logout()
        + getUserDetails()
    }
}

package "View Layer" {
    class Student {
        + raiseLeaveRequest()
        + checkLeaveStatus()
        + viewLeaveHistory()
    }

    class Class_In_Charge {
        + validateLeaveRequest()
        + viewStudentLeaveHistory()
        + forwardLeaveRequest()
    }

    class Department_Head {
        + validateLeaveRequest()
        + allocateStudentsAndClassInCharge()
        + addStudent()
        + removeStudent()
        + viewStudentInformation()
        + viewStudentLeaveHistory()
    }
```

```
}

package "Controller Layer" {
    class StudentController {
        - model: User
        - view: Student
        + login()
        + raiseLeaveRequest()
        + checkLeaveStatus()
        + viewLeaveHistory()
    }

    class Class_In_ChargeController {
        - model: User
        - view: Class_In_Charge
        + login()
        + validateLeaveRequest()
        + viewStudentLeaveHistory()
        + forwardLeaveRequest()
    }

    class Department_HeadController {
        - model: User
        - view: Department_Head
        + login()
        + approveEmergencyLeaveRequest()
        + rejectEmergencyLeaveRequest()
    }
}

' Connect the components within MVC
StudentController --> User_validation: Uses
StudentController --> Student: Uses
Class_In_ChargeController --> User_validation: Uses
Class_In_ChargeController --> Class_In_Charge: Uses
Department_HeadController --> User_validation: Uses
Department_HeadController --> Department_Head: Uses

Student --> StudentController: Calls
Class_In_Charge --> Class_In_ChargeController: Calls
Department_Head --> Department_HeadController: Calls

User_validation --> Database: Manages Data
LeaveRequest --> Database: Manages Leave Requests
User --> Database: Manages User Data

@enduml
```

1. **Model Layer:**

   - **Database:** This component represents the database layer where all data operations are handled. It includes methods like **connect()** to establish a connection to the database, **executeQuery()** for executing SQL queries, and methods like **updateRecord()**, **deleteRecord()**, and **insertRecord()** for managing data in the database tables related to users, leave requests, and other system data.
   - **LeaveRequest:** This part of the model layer manages operations specific to leave requests. Methods like **getLeaveRequests()** fetch leave requests from the database, **addLeaveRequest()** adds new leave requests, **updateLeaveRequest()** updates existing requests, and **deleteLeaveRequest()** removes leave requests from the system.

2. **View Layer:**

   - **Student:** The student view handles interactions and displays information related to leave management for students. Functions like **raiseLeaveRequest()** allow students to submit leave requests, **checkLeaveStatus()** enables them to check the status of their requests, and **viewLeaveHistory()** shows their past leave history.
   - **Class_In_Charge:** This view is for class in charge personnel who are responsible for managing leave requests for their respective classes. Methods like **validateLeaveRequest()** allow them to approve or reject leave requests, **viewStudentLeaveHistory()** lets them see leave history for students in their class, and **forwardLeaveRequest()** helps them forward requests to higher authorities if needed.
   - **Department_Head:** The department head view provides functionalities for department heads to manage leave requests at a broader level. Functions such as **validateLeaveRequest()** handle the approval process for leave requests, **allocateStudentsAndClassInCharge()** assigns students and class in charge personnel to specific departments or classes, and other functions like **addStudent()**, **removeStudent()**, **viewStudentInformation()**, and **viewStudentLeaveHistory()** aid in managing students and their leave-related data.

3. **Controller Layer:**

   - **StudentController:** This controller bridges the gap between the student view and the model layer. It handles actions like **login()**, **raiseLeaveRequest()**, **checkLeaveStatus()**, and **viewLeaveHistory()** by interacting with the appropriate parts of the model layer and updating the student view accordingly.
   - **Class_In_ChargeController:** Similarly, this controller manages interactions between the class in charge view and the model layer. It oversees actions such as **login()**, **validateLeaveRequest()**, **viewStudentLeaveHistory()**, and **forwardLeaveRequest()**, ensuring smooth communication and data flow between components.
   - **Department_HeadController:** This controller serves the department head view, handling tasks such as **login()**, **approveEmergencyLeaveRequest()**, **rejectEmergencyLeaveRequest()**, and other administrative functions related to leave management and student data at the department level.

Interactions:

- Controllers interact with the model layer to perform data operations, such as validating users, managing leave requests, and accessing database functions.
- Views communicate with their respective controllers to handle user interactions and display data based on the actions performed.
- The model layer handles data processing and storage, ensuring that the system operates efficiently and securely.
- The MVC architecture promotes separation of concerns, making it easier to maintain and scale the system as each component has a specific role and responsibility.