Leave Approval System Requirements and Specification

1. Introduction

1.1. Purpose

The purpose of this document is to present a detailed description of the Leave Approval System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate, and how the system will react to external stimuli.

1.2. Scope of Project

- This software system will serve as a portal for students and faculty to apply for leave and for the respective approvers to approve or reject leave applications.
- From an administrative perspective, this system will provide a comprehensive dashboard for monitoring leave applications.

Project Information:

- **Name:** Faseem Ahmed.G
- **Roll no:** 7376231EC801
- **Seat no:** 318
- **Project ID:** 22
- **Problem Statement:** Leave Approval System

---

2. System Overview

2.1. Users

1. **Students:**
   - Apply for leave.
   - Upload supporting documents.
   - Monitor the status of their application.
   - Review their leave application history.
2. **Faculty:**
   - Apply for leave.
   - Approve or reject student leave applications.
   - Monitor the status of their own leave applications.
   - Review their leave application history.

3. **HOD (Head of Department):**
    - Approve or reject faculty leave applications.
    - Monitor the status of leave applications within their department.
    - Review leave application history for faculty in their department.
4. **Admin:**
    - Manage user accounts.
    - Configure leave policies.
    - Generate reports and analytics on leave applications.

2.2. Features

1. **Login and Registration:**
    - Users can register for an account or login with their existing account.
2. **Leave Application Submission:**
    - Students and faculty can input relevant details regarding their leave application including dates, reason, type of leave, and any necessary attachments.
    - Applications are routed to the respective approvers (faculty for students, HOD for faculty).
3. **Application Status:**
    - Users can view the current status of their application and the history logs.
4. **Approval Process:**
    - Faculty can approve or reject student leave applications.
    - HODs can approve or reject faculty leave applications.
    - Approvers can add remarks during approval or rejection.
5. **Admin Access:**
    - Admins can view all submitted leave applications.
    - Admins can manage users, configure leave policies, and access analytical dashboards.
6. **Notifications:**
    - Email/SMS notifications for leave application submissions and status updates.

---

3. System Requirements Specification

3.1. Functional Requirements

- **User Management:**
    - Students, faculty, and HODs can register and login.
    - Admins have access control with an analytical dashboard and dedicated features.
- **Leave Application:**
    - Students and faculty can submit applications with appropriate details.
    - Application form contains:
        - Start Date
        - End Date
        - Reason for leave

        - Type of leave
        - Optional attachments
- **Application Status:**
    - Users can view the current status of their application.
    - If the application is rejected, the remarks are shown.
    - Users can also see the logs of their applications.
- **Approval Workflow:**
    - Student leave applications are routed to the responsible faculty for approval/rejection.
    - Faculty leave applications are routed to the respective HOD for approval/rejection.
- **Admin Dashboard:**
    - Admins can view a list of all submitted leave applications.
    - Applications can be filtered by type (student, faculty).
    - Admins can view details of each application.
    - Admins can approve or reject applications with suitable remarks.
- **Analytics Dashboard:**
    - Admins can view the number of applications by type.
    - Admins can view the number of approved/rejected applications.

3.2. Non-Functional Requirements

- **Performance:**
    - The system must respond to user actions within 2 seconds to ensure efficient usability.
    - Must handle a concurrent user load of at least 100 users without significant performance degradation.
- **Security:**
    - User data must be encrypted during transmission and storage.
    - Access to sensitive functionalities should be restricted to authorized admin users through secure authentication mechanisms.
- **Usability:**
    - The user interface should be intuitive and user-friendly, with clear and concise error messages provided to guide users in case of input errors or system failures.
- **Reliability:**
    - The system should be available 24/7 with minimal downtime.
    - Should have a backup and recovery mechanism in place to prevent data loss in case of system failures or crashes.
- **Scalability:**
    - The system should be designed to accommodate an increasing number of users and data volume over time.
    - It should be scalable to support additional features and functionalities as per future requirements.

4. System Design

Backend:

1. **User Entity:**
   - `user_id` (Primary Key)
   - `username` (String)
   - `password` (String, encrypted)
   - `role` (String: Student, Faculty, HOD, Admin)
   - `email` (String)
   - `department_id` (String, for Faculty and HOD)
2. **Leave Application Entity:**
   - `leave_id` (Primary Key)
   - `user_id` (Foreign Key)
   - `start_date` (Date)
   - `end_date` (Date)
   - `reason` (String)
   - `type` (String)
   - `status` (String: Pending, Approved, Rejected)
   - `approver_id` (Foreign Key)
   - `remarks` (String)
   - `created_at` (Date)
3. **Leave Type Entity:**
   - `type_id` (Primary Key)
   - `type_name` (String)
   - `max_days` (Integer)
4. **Department Entity:**
   - `department_id` (Primary Key)
   - `department_name` (String)

Frontend:

- **Login Form**
- **Register Form**
- **User Dashboard (Student/Faculty/HOD)**
- **Leave Application Form**
- **Leave Status and History Page**
- **Admin Dashboard**
- **Approval Page (Faculty/HOD)**
- **Analytics Dashboard (Admin)**

---

5. Technology Stack

- **Frontend:** React.js or Angular.js, Bootstrap or Tailwind CSS
- **Backend:** Spring Boot, Spring Security, Spring Data JPA
- **Database:** MySQL or PostgreSQL
- **Build Tool:** Maven or Gradle
- **Version Control:** Git
- **Deployment:** Docker, Kubernetes

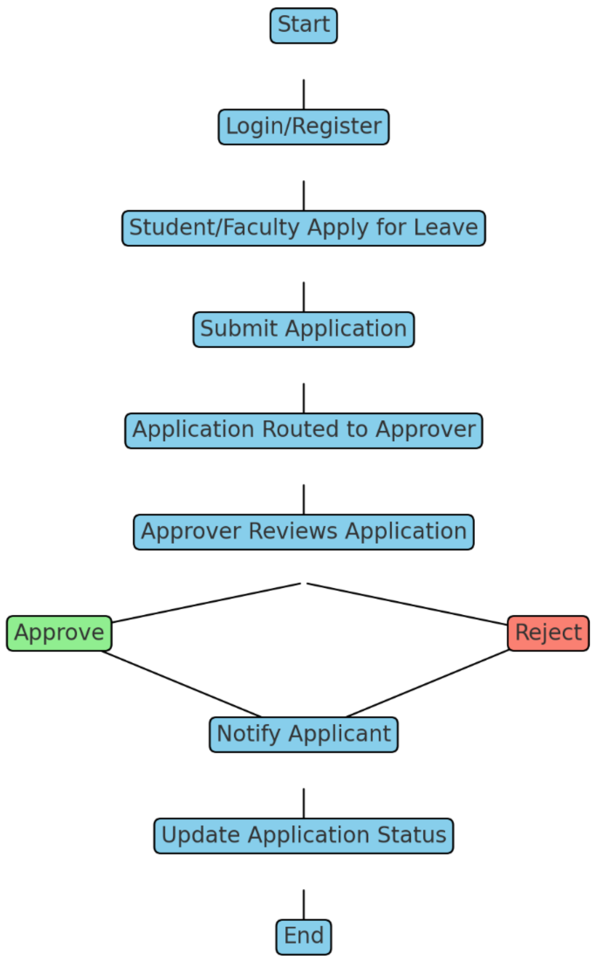
---

6. Prototype of the Project

1. **Login Form**
2. **Register Form**
3. **Student's View**
4. **Leave Application Form**
5. **Leave Status and History Page**
6. **Approval Page (Faculty/HOD)**
7. **Admin's View**
8. **Analytics Dashboard**

This approach ensures that all aspects of the leave approval project are well-documented and clear, providing a solid foundation for development and implementation.

WORK FLOW:


Start
Login/Register
Student/Faculty Apply for Leave
Submit Application
Application Routed to Approver
Approver Reviews Application
Approve
Reject
Notify Applicant
Update Application Status
End