PROJECT:- EMPLOYEE LEAVE MANAGEMENT SYSTEM

NAME:- RAJESH KUMAR PANDA

EMAIL:- rajeshkumarpanda857@gmail.com

Mobile:- +91 8658182272

ABSTRACT:-

The Employee Leave Management System is an enterprise-oriented web application that centralizes leave operations by providing a structured, real-time workflow for request submission, departmental approval, and balance management. Developed using Flask with a MySQL-backed data layer and real-time event handling via Socket.IO, the system enforces role-based permissions to ensure secure and accountable interactions between employees and managers. It enables seamless leave visibility through dynamic dashboards and calendar views while delivering instant status updates to eliminate manual follow-ups and communication gaps. By prioritizing reliability, performance, and ease of use, the solution supports scalable HR process digitization for organizations seeking efficient workforce availability management without operational overhead.

INTRODUCTION:-

In traditional workplace environments, managing employee leave requests through manual processes, email chains, or paper forms creates inefficiencies, delays, and potential errors. HR departments and managers often struggle with tracking leave balances, maintaining accurate records, and ensuring timely approvals. Employees face uncertainty about their request status and lack visibility into team availability, which can impact project planning and collaboration. The digital transformation of HR processes has become essential for modern organizations. An automated leave management system reduces administrative overhead, ensures data accuracy, provides instant status updates, and creates a transparent audit trail for compliance purposes.

PROBLEM STATEMENT:

Organizations with distributed teams and departmental structures encounter recurring issues in leave administration that stem from the absence of a unified system. Approval processes are slowed by manual handoffs, leave balances are prone to human error, and employees lack real-time awareness of request status or team schedules. Managers, on the other hand, require scoped visibility limited to their departments while maintaining accountability and auditability. The absence of instant communication further amplifies inefficiencies, leading to operational delays and avoidable conflicts in project planning.

SCOPES:

- User authentication and authorization (employees and managers)
- Leave request submission with multiple leave types (vacation, sick, other)
- Department-based approval workflows
- Real-time Socket.IO notifications
- Leave balance tracking and automatic deduction
- Interactive calendar with approved leave visualization
- Request history and status tracking
- Manager dashboard with pending requests queue
- Responsive design for desktop and mobile access

SYSTEM OVERVIEW

System Architecture

The Employee Leave Management System follows a three-tier architecture:

1. **Presentation Layer (Frontend)**
    - HTML5, CSS3, JavaScript
    - Responsive design using Flexbox/Grid
    - Interactive UI components
    - Real-time updates via Socket.IO client
2. **Application Layer (Backend)**
    - Flask web framework (Python)
    - RESTful API endpoints
    - Session-based authentication
    - Business logic implementation
    - Socket.IO server for real-time features
    - Werkzeug security for password hashing
3. **Data Layer (Database)**
    - MySQL relational database
    - Three main tables: users, leaves, notifications
    - Foreign key constraints for referential integrity
    - Indexes for query optimization

Database Design:

1) **Users Table**
    a. Stores employee and manager information
    b. Fields: id, name, email, password, role, department, vacation_balance, sick_balance, created_at
    c. Role: ENUM('employee', 'manager')
    d. Department-based organization
2) **Leaves Table**
    a. Tracks all leave requests
    b. Fields: id, user_id, user_name, department, leave_type, start_date, end_date, reason, status, submitted_at, manager_comment, approved_by, approved_at, rejected_by, rejected_at
    c. Status: ENUM('pending', 'approved', 'rejected')
    d. Foreign key reference to users table
3) **Notifications Table**
    a. Manages real-time notification history
    b. Fields: id, type, message, leave_id, department, created_at, read
    c. Foreign key reference to leaves table
    d. Department-specific filtering

Key Features and Functionality

1) **User Authentication**
    a. Secure login with hashed passwords using Werkzeug
    b. Registration system for new employees
    c. Session management for persistent login
    d. Role-based access control (RBAC)
    e. Automatic redirection based on user role
2) **Employee Features**
    a. Dashboard Overview: View personal leave balances (vacation and sick days)
    b. Leave Request Submission:
        i. Select leave type (vacation, sick, other)
        ii. Choose start and end dates
        iii. Provide reason for leave
        iv. Submit to department manager

    c. Request History: View all past requests with status
    d. Team Calendar: Interactive monthly calendar showing approved team leaves
    e. Real-time Notifications: Instant alerts when requests are approved/rejected

3) **Manager Features**
    a. Pending Requests Queue: View all pending requests from department employees
    b. Approval Workflow:
        i. Review request details
        ii. Approve or reject with comments
        iii. Automatic balance deduction on approval
    c. Department View: See only requests from managed department
    d. Team Calendar: Full visibility of department leave schedule
    e. All Requests History: Filter and review all historical requests
    f. Real-time Notifications: Instant alerts when employees submit new requests

4) **Calendar Visualization**
    a. Monthly grid view with navigation
    b. Color-coded leave type indicators
    c. Employee name display on leave dates
    d. Legend for leave type identification
    e. Detailed leave list below calendar

Technology Stack

1) **Backend Technologies:**
    a. Python 3.8+
    b. Flask 2.3.0 (Web framework)
    c. MySQL Connector Python 8.0.33 (Database driver)
    d. Werkzeug 2.3.0 (Password hashing and security)

2) **Frontend Technologies:**
    a. HTML5 (Semantic markup)
    b. CSS3 (Modern styling with Flexbox/Grid)
    c. JavaScript

3) **Database:**
    a. MySQL 8.0+ (Relational database)

User Workflow

1) **Employee Workflow:**
    a. Login → View Dashboard → Check Leave Balances
    b. Navigate to "New Request" → Fill Form (dates, type, reason)
    c. Submit Request → Receive Confirmation
    d. Receive Real-time Notification on Approval/Rejection
    e. View Updated Balance → Check Team Calendar
2) **Manager Workflow:**
    a. Login → View Dashboard → See Pending Count Badge
    b. Navigate to "Pending Requests" → Review Details
    c. Click "Review" → Read Request Information
    d. Approve/Reject with Comment → Submit Decision
    e. System Updates Balance (if approved)
    f. Receive Real-time Notification on New Submissions

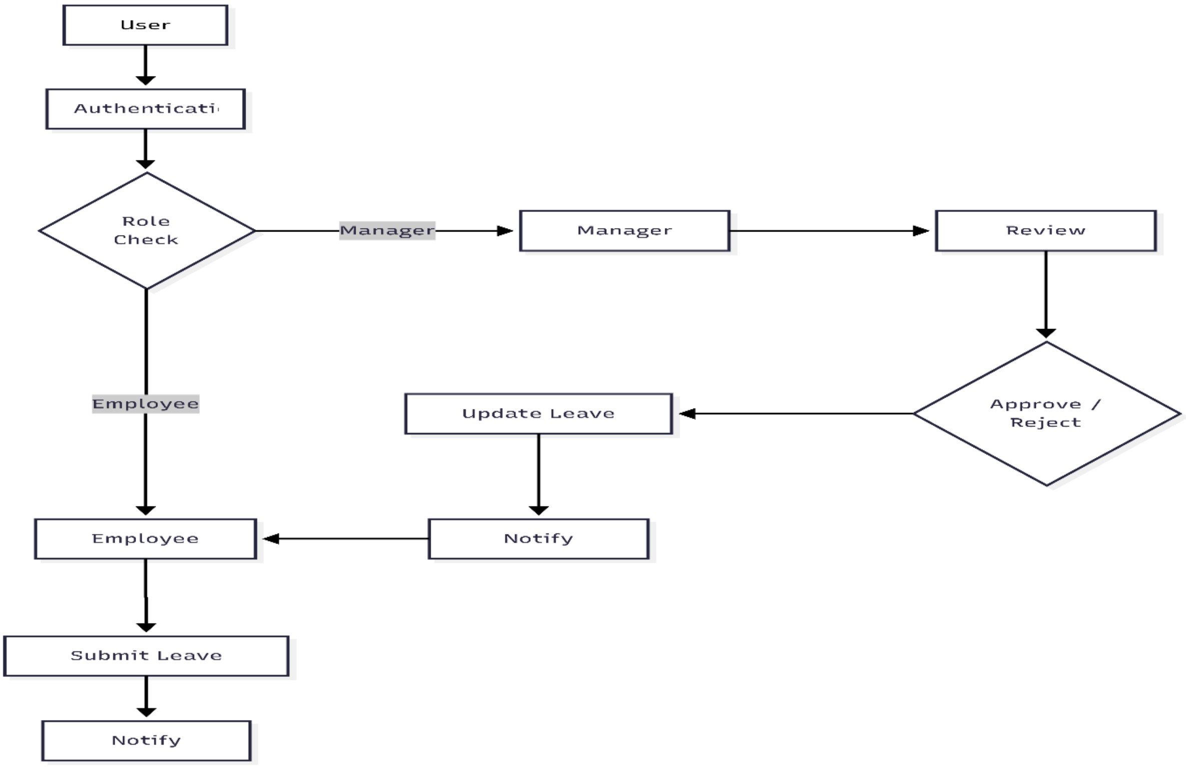


Figure 1 Workflow of the Employee Leave Management System

Figure 1: workflow diagram of the Employee Leave Management System demonstrating role-based access, leave request submission, managerial review, status update, and notification flow between employees and managers.

TECHNICAL DIAGRAMS

Entity-Relationship (ER) Diagram

The ER diagram shows the actual MySQL database structure with three tables: users, leaves, and notifications, including all attributes, data types, and foreign key relationships.

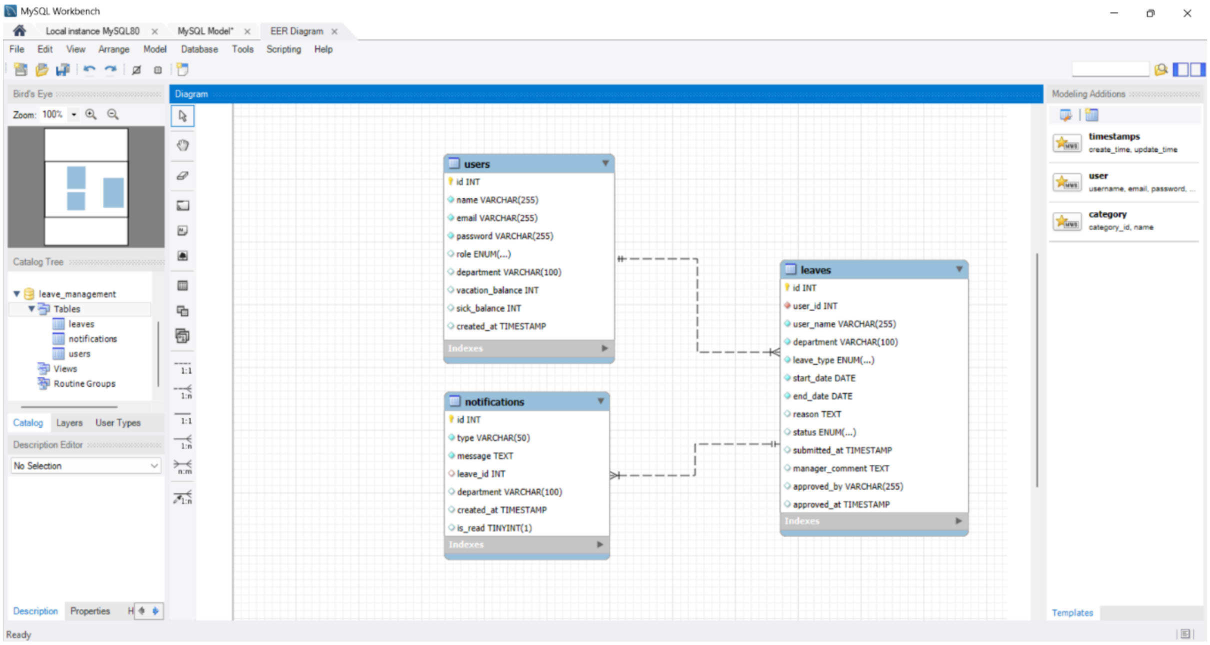


Figure 2 Entity–Relationship (ER) Diagram of the Employee Leave Management System

Data Flow Diagram (DFD) - Level 0

The Context Diagram shows the system boundary with external entities (Employee and Manager) and their interactions with the Leave Management System.

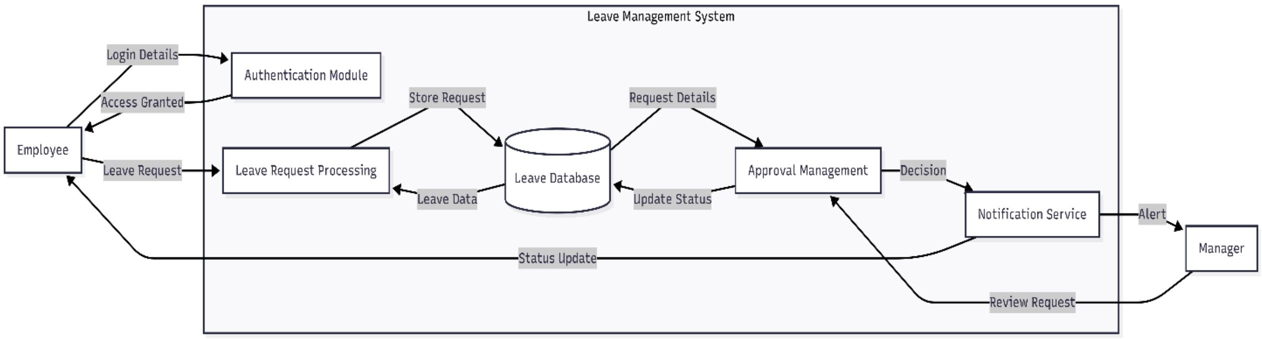


Figure 3 Data Flow Diagram (DFD) of the Employee Leave Management System

Sequence Diagram

This sequence diagram illustrates the complete interaction flow when an employee submits a leave request and a manager approves it, including real-time notifications.

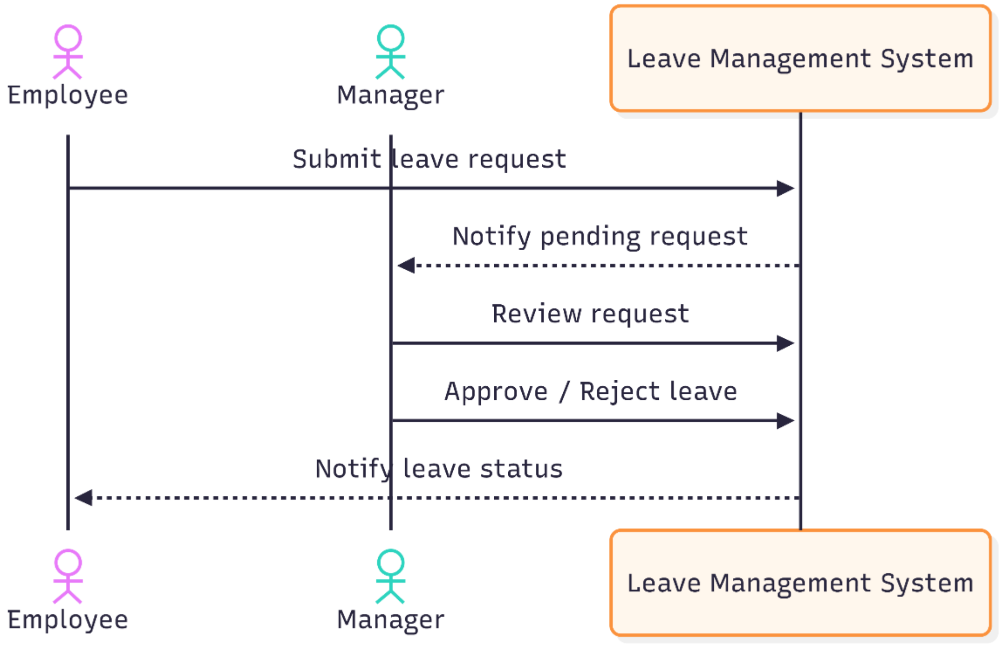
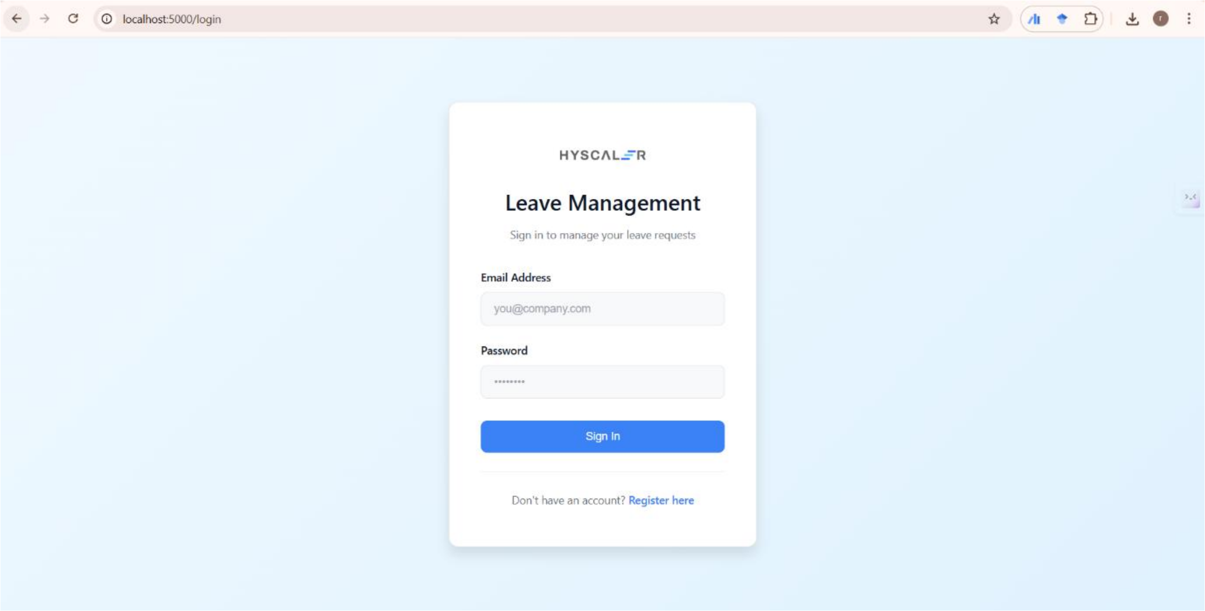


Figure 4 Sequence Diagram of Leave Request and Approval Process

SYSTEM SCREENSHOTS



Leave Management

Sign in to manage your leave requests

Email Address

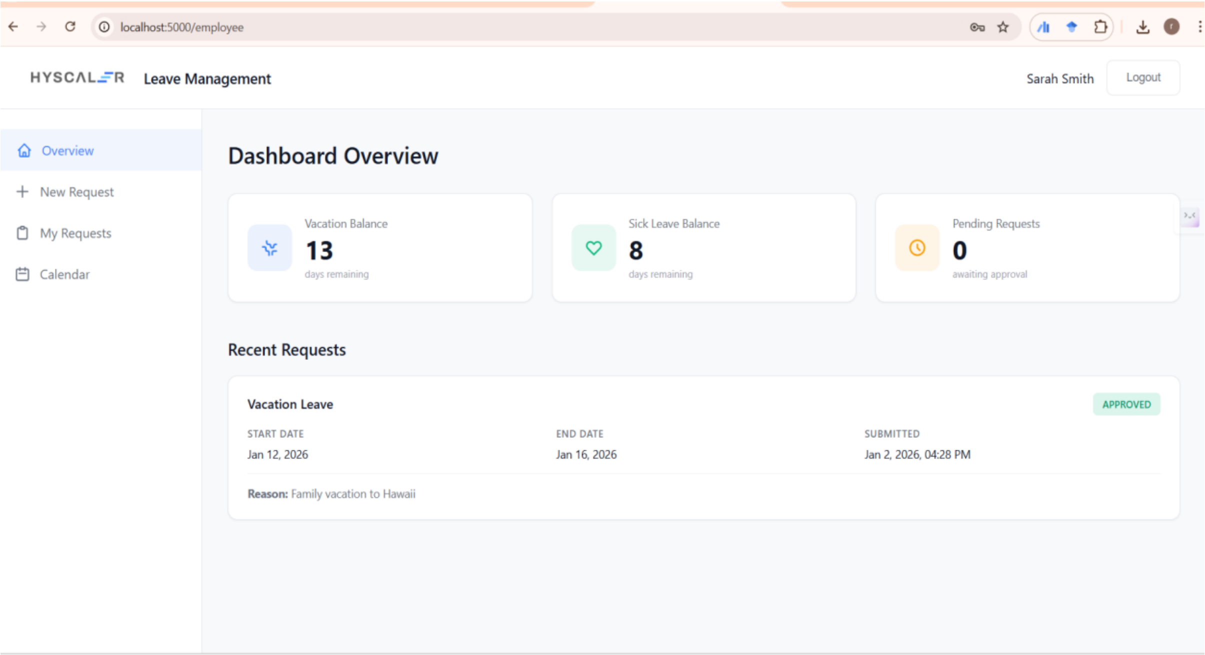
you@company.com

Password

••••••••

Don't have an account? Register here

Figure 5 Login Page - User authentication interface



localhost:5000/employee

HYSCALER Leave Management

Sarah Smith Logout

Overview

New Request

My Requests

Calendar

Dashboard Overview

Vacation Balance
13
days remaining

Sick Leave Balance
8
days remaining

Pending Requests
0
awaiting approval

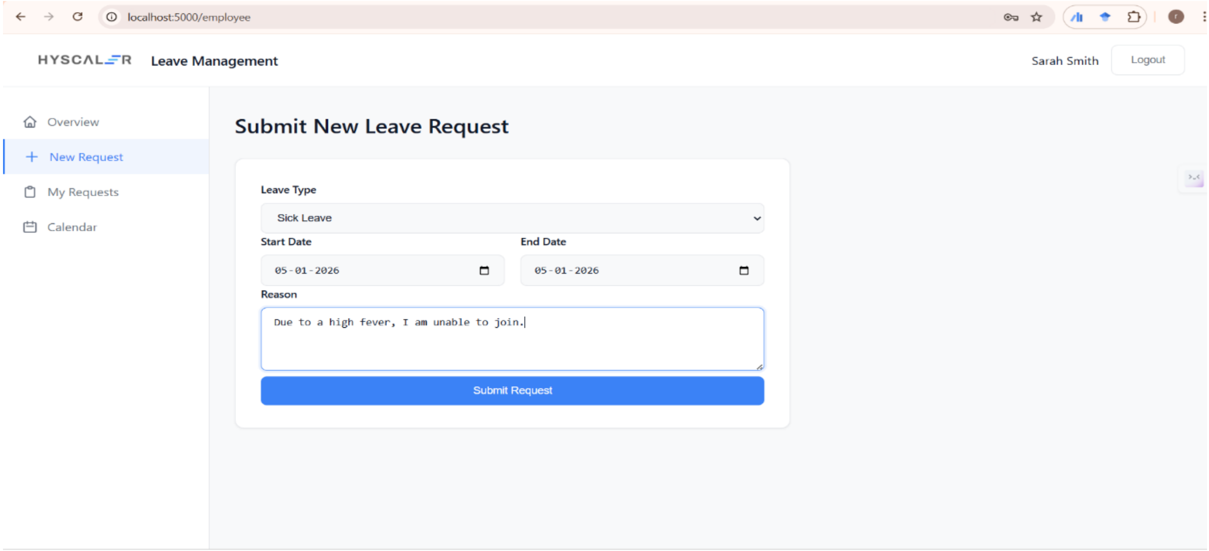
Recent Requests

Vacation Leave APPROVED

START DATE	END DATE	SUBMITTED
Jan 12, 2026	Jan 16, 2026	Jan 2, 2026, 04:28 PM

Reason: Family vacation to Hawaii

Figure 6 Employee Dashboard - Overview with leave balances



localhost:5000/employee

HYSCALER Leave Management

Sarah Smith Logout

Overview

New Request

My Requests

Calendar

Submit New Leave Request

Leave Type: Sick Leave

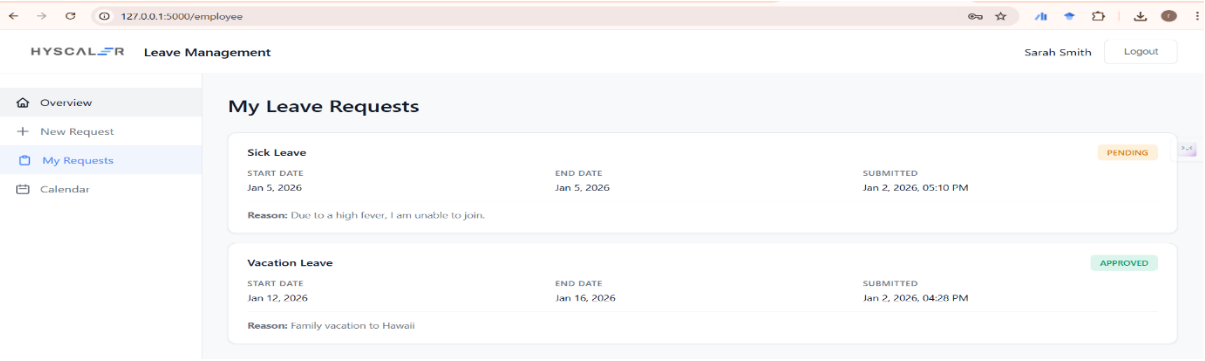
Start Date: 05-01-2026

End Date: 05-01-2026

Reason: Due to a high fever, I am unable to join.

Submit Request

Figure 7 New Leave Request Form - Leave submission interface



127.0.0.1:5000/employee

HYSCALER Leave Management

Sarah Smith Logout

Overview

New Request

My Requests

Calendar

My Leave Requests

Sick Leave PENDING

START DATE	END DATE	SUBMITTED
Jan 5, 2026	Jan 5, 2026	Jan 2, 2026, 05:10 PM

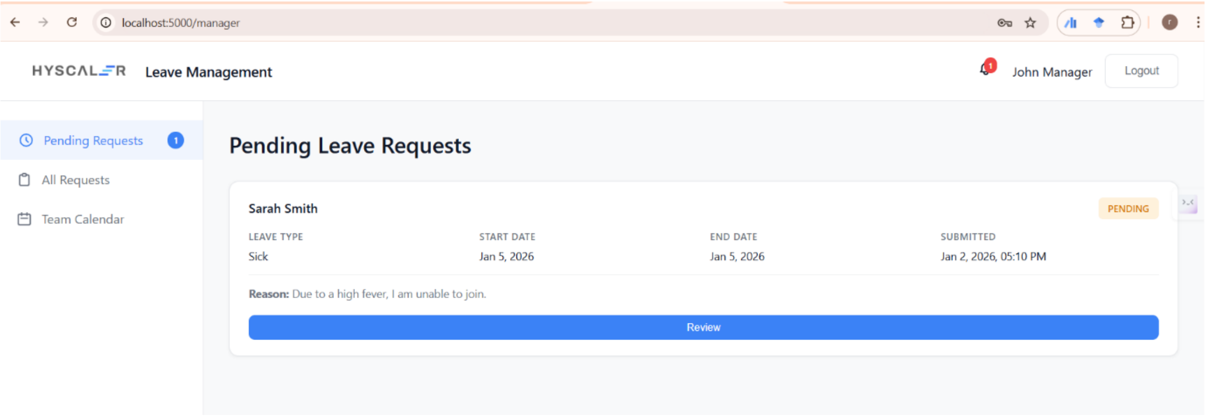
Reason: Due to a high fever, I am unable to join.

Vacation Leave APPROVED

START DATE	END DATE	SUBMITTED
Jan 12, 2026	Jan 16, 2026	Jan 2, 2026, 04:28 PM

Reason: Family vacation to Hawaii

Figure 8 My Requests Page - Request history with status



localhost:5000/manager

HYSCALER Leave Management

John Manager Logout

Pending Requests 1

All Requests

Team Calendar

Pending Leave Requests

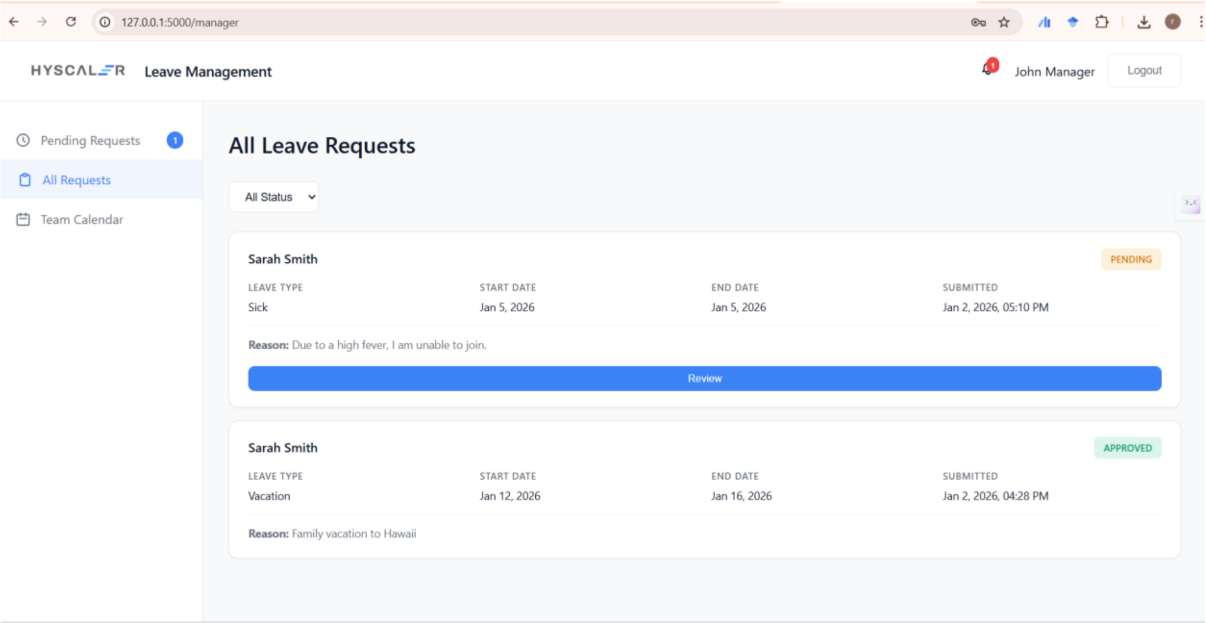
Sarah Smith PENDING

LEAVE TYPE	START DATE	END DATE	SUBMITTED
Sick	Jan 5, 2026	Jan 5, 2026	Jan 2, 2026, 05:10 PM

Reason: Due to a high fever, I am unable to join.

Review

Figure 9 Manager Dashboard - Overview with pending count



127.0.0.1:5000/manager

HYSCALER Leave Management

John Manager Logout

Pending Requests 1

All Requests

Team Calendar

All Leave Requests

All Status

Sarah Smith PENDING

LEAVE TYPE	START DATE	END DATE	SUBMITTED
Sick	Jan 5, 2026	Jan 5, 2026	Jan 2, 2026, 05:10 PM

Reason: Due to a high fever, I am unable to join.

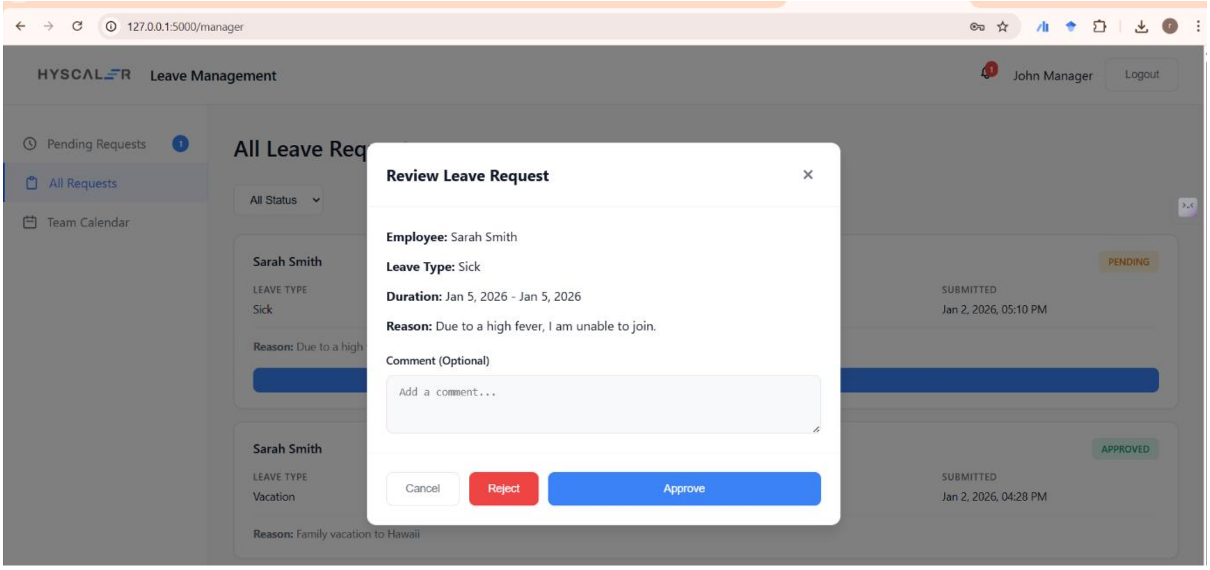
Review

Sarah Smith APPROVED

LEAVE TYPE	START DATE	END DATE	SUBMITTED
Vacation	Jan 12, 2026	Jan 16, 2026	Jan 2, 2026, 04:28 PM

Reason: Family vacation to Hawaii

Figure 10 Pending Requests - Manager approval queue



127.0.0.1:5000/manager

HYSCALER Leave Management

John Manager Logout

Pending Requests 1

All Requests

Team Calendar

All Leave Req

All Status

Sarah Smith PENDING

LEAVE TYPE SUBMITTED

Sick Jan 2, 2026, 05:10 PM

Reason: Due to a high

Sarah Smith APPROVED

LEAVE TYPE SUBMITTED

Vacation Jan 2, 2026, 04:28 PM

Reason: Family vacation to Hawaii

Review Leave Request ×

Employee: Sarah Smith

Leave Type: Sick

Duration: Jan 5, 2026 - Jan 5, 2026

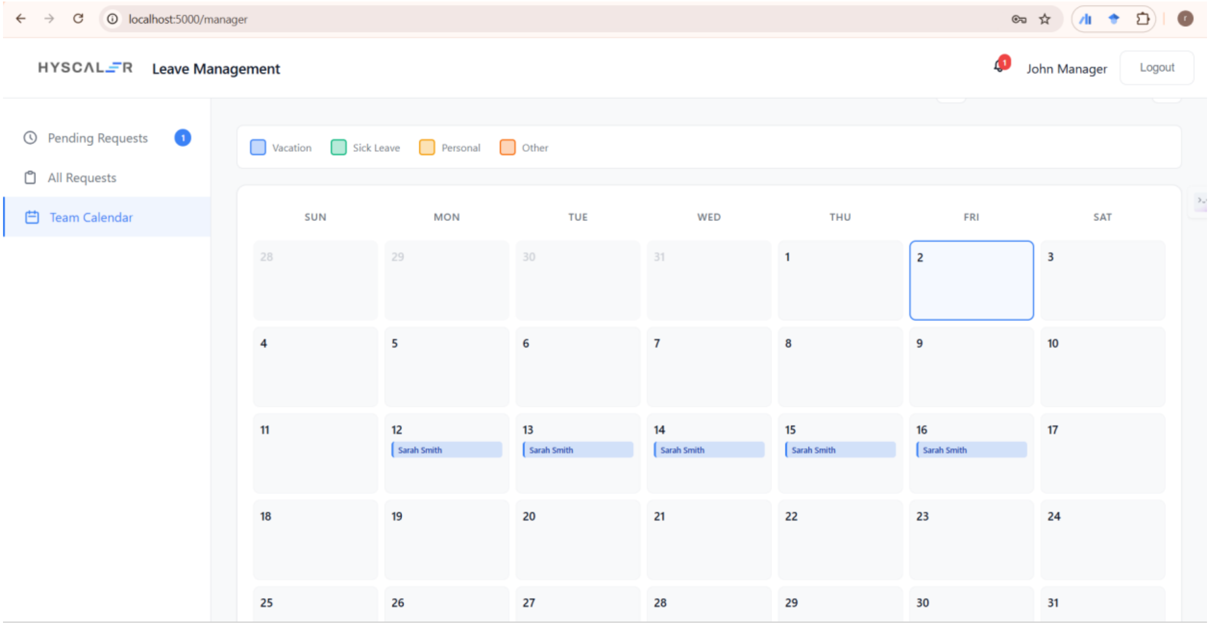
Reason: Due to a high fever, I am unable to join.

Comment (Optional)

Add a comment...

Cancel Reject Approve

Figure 11 Approval Modal - Leave review and decision interface



SUN	MON	TUE	WED	THU	FRI	SAT
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12 Sarah Smith	13 Sarah Smith	14 Sarah Smith	15 Sarah Smith	16 Sarah Smith	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Figure 12 All Department Leaves - Complete leave history

CONCLUSION

The Employee Leave Management System effectively replaces manual leave handling with a centralized, secure, and real-time digital platform. By leveraging Flask, MySQL, and Socket.IO, the system automates leave submission and approval workflows, ensures accurate balance tracking, and provides clear visibility through dashboards and calendar views, resulting in improved efficiency, transparency, and coordination across departments.

This project offered hands-on experience in full-stack development, real-time web architecture, relational database design, and role-based security implementation. It strengthened practical understanding of building scalable, user-centric systems while applying software engineering best practices suitable for real-world organizational environments.
]