

**SCHOOL OF COMPUTER SCIENCE AND
ENGINEERING**

**CSE1005 - Software Engineering
Laboratory**

Slot: L31+L32

FALL 2024-25

Faculty In charge:

SHAIK SHABANA, SCOPE, VIT-AP University

A Mini Project Report On

EASY LEAVE

Team Members:

22BCE9683: Shaik Soha

22BCE8980: Nukala Sadhvikha

22BCE9783: Farheen Naz Mohammad

22BCE9791: Baddela Varshitha Ambedkar

0. PROBLEM STATEMENT:

Many traditional leave management processes are largely dependent on outdated methods which means there are often issues with efficiency and operations with the organization. This includes late approvals of leave requests and leaves that can be difficult to track due to not having access to an individual or organization-wide leave balance leaving transparency issues in leave records and leaves being double booked and causing staffing shortages in the organization with little to no notice to the organization. Employees can be unhappy with lack of organizational clarity, organizational stability due to inconsistency in self-service recognition by the leave approvers, morale issues from lack of transparency/understanding of approvals within the organization and employee trust within the organization processes. Additionally, there is limited data and automation meaning management may have issues maintaining accurate leave records and data to determine when to approve leave and how scheduling effects resourcing within the organization. The vast majority of organizations are experiencing tremendous growth with very demanding work and labour environments as it is incredibly hard to keep pace with operational demands let alone immediately adapt to change operationally with conventional, basic leave processes. Therefore, as organizations prepare for additional growth they need to confirm they are using an automated and up to date leave management processes to improve efficiency, accuracy, employee satisfaction, and productivity.

1. PROBLEM ANALYSIS:

1.1 Overview:

The "Easy Leave" project aims to address these inefficiencies by transitioning from manual processes to an automated leave management system. Current systems create several issues:

- **Delayed Approvals:** Manual handling of leave requests can cause significant delays, affecting employee morale and operational efficiency.
- **Lack of Transparency:** Employees and managers often lack clear visibility into leave status and balances due to fragmented information.
- **Operational Disruptions:** Overlapping leaves and poor tracking disrupt operations and hinder effective workforce planning.
- **Employee Dissatisfaction:** Inefficient leave management processes can lead to lower employee satisfaction and engagement.

The "Easy Leave" project intends to resolve these issues by automating the leave application process, enhancing efficiency, accuracy, and transparency.

1.2 Scope:

The project includes:

1. **Automated Leave Application:** A user-friendly system for applying and tracking various leave types, with real-time balance updates.
2. **Approval Workflow:** Automated notifications and reminders for managers, with status updates for employees.
3. **Shared Calendar:** Integration to prevent overlapping leaves and provide managers with a clear view of team availability.
4. **Policy Enforcement:** Adherence to company-specific leave policies, including limits, blackout periods, and carry-forward rules.

5. **Reporting and Analytics:** Tools for analyzing leave trends, identifying patterns, and making informed resource planning decisions.

6. **Security Measures:** Role-based access control to protect sensitive employee data.

1.3 Objectives:

1. Increase efficiency by reducing manual steps in leave management processes. Automating approvals and notifications enables workflows to flow faster and with less delay, relieving HR teams and managers of administrative burdens.
2. Increase transparency through real-time access to leave data for employees and management with real-time visibility to leave requests and related information which supports employee tracking and managers are able to track team availability.
3. Increase employee satisfaction by ensuring leave management is simple and easy to complete. Reducing friction for applying, approving and tracking their leave increases trust and engagement.
4. Increase the efficiency of workforce planning through enhanced reporting tools that allow management to better understand patterns of leave (who may be taking leave at the same time) and whether the business will have enough staff when staff members are on leave.
5. Provide compliance and accountability with processes in place. Introducing standard leave processes and employing automated checks such as if the leave duration is within the existing policy reduces opportunities for mistakes and disagreements about leave entitlements and approvals.
6. Protect sensitive employee information by using robust data security measures. These measures can include secure login (two-factor authentication), encrypted data, and narrow access restrictions. Employee confidentiality builds confidence among the work group while supporting around data privacy policies and regulations.

1.4 Infrastructure:

1. Hosting: AWS, Google Cloud, or Azure for scalable hosting; on-premise servers if needed.
2. Database: RDBMS like MySQL, PostgreSQL, or SQL Server; optionally MongoDB.
3. Web Server: Apache or Nginx.
4. Backup: Automated backups with AWS Backup; multi-region disaster recovery.

Tools & Technologies:

1. Development: Visual Studio Code, Git, Docker.
2. Frontend: React.js, Bootstrap, JavaScript, HTML.
3. Backend: Python, Java, Node.js; RESTful APIs, Django.
4. Testing: JUnit, PyTest, Selenium.
5. Deployment: Render, Vercel.
6. Project Management: Jira, Slack.

2. SOFTWARE REQUIREMENT SPECIFICATION (SRS) DOCUMENT:

2.1 Description of Individual Module

2.1.1 User Characteristics

List of users and features users are given with.

2.1.2 General Constraints

Constraints of the system

2.1.3 Assumptions

Pre-requisites to use the system

Dependency

Completion of any process related to other.

2.1.4. Functional Requirements

Use Cases – description, input/output

2.1.5 Identify individual module deliverables

List of modules and working

2.2 References

2.1 Description of Individual Modules

2.1.1 User Characteristics

Types of Users:

- Employee:
Duties: Request leave, see status of leave request, and view leave history.
- Manager:
Duties: Approve or deny employee leave requests, view leave calendar for the team, and keep track of leave banks for the team.
- HR/Admin:
Duties: Administer the leave policies, have visibility into organization-wide leave data, and report on leave data.

User Functionality:

Employee Functionality:

- Request for various types of leave.
- View leave status and history.
- View current leave bank balance.

Manager Functionality:

- Approve or deny leave request from employees.
- View and manage team leave calendar.
- Track and manage team leave bank balances.

HR/Admin Functionality:

- Set up and edit leave policies.
- Manage user accounts and assign user types/roles.
- Make leave reports including exporting, and data analyzing.

2.1.2 General Constraints

System Constraints:

Leave management systems have specific requirements: implement different leave types (sick leave, vacation leave, emergency leave) to support the different needs of employees, be available in web and mobile applications to allow users some flexibility about how to access everything, organisation flexibility by providing integration for HR software systems; keep focused on what the workplace is doing keeping their approach aligned with their organisational software; allow for multi-languages to accommodate multiple time zones supported by international workforce; compliance with data protection and security laws (GDPR) to maintain trust with the organisation, ensure sensitive employee information remains confidential; and legal requirements will keep appropriate confidence with the employee's information. These requirements ensure a leave management system is flexible, easy to use, secure, privacy-complaint, and scalable to modern workplaces.

2.1.3 Assumptions

Pre-requisites to Use of the System:

To appropriately use the leave management system, the user must have an active employee account within the organisation's database. The system requires access to the internet; the system will be both web and mobile applications. Users should have basic skills in using web applications for leave applications, checking status of leave applications and other related processes. These conditions will ensure that users have direct experience of using the system with little training, ensuring a better user experience.

Dependencies:

This system will depend on being able to have accurate employee information such as leave balances, employee role and department requirements in relation to access permissions, potential leave requests and management of leave applications. It also needs a robust backend infrastructure including some form of stable data store, such as a relational database (i.e. MySQL) to maintain user data, leave cases, and any system logs. It requires some form of web server (e.g. apache web server) to host the application and serve user requests. In terms of necessary workflow manager, the system is also dependent on managerial level actions. Must have some approval (managerial) to finalize leave applications and make visible to the rest of the organisation (management responsibility) within the system, therefore, process management and a responsible step will be required to remain in control of the application and workforce management. These dependencies should be considered carefully as they also have implications around data integrity and security within the organisational environment.

2.1.4 Functional Requirements

Use Cases:

Apply for Leave:

Employees start leave requests by selecting the leave type (sick leave, vacation leave, or emergency leave) and the start & end date for their leave along with the reason for it. Once an employee submits a leave request, that request is in a pending state until the manager approves

it. This application simplifies the leave process from start to finish for the employee by eliminating manual paperwork.

Approve / Reject Leave:

Managers receive all pending leave requests from their team members. Managers will accept or reject requests based on their workload and host company policies. The system will notify the employee of the approval and change their leave balances automatically. Managers will also be notified of any changes to their team or personal leave balances as well with real-time updates.

Generate Leave Report:

HR or administrative users can generate leave reports for any date range or leave type across any department. The report gives a pattern of the leave days requested and which leave types were used among the workforce. The application generates reports for exporting and downloading this could be in PDF or EXCEL formats for easy sharing and collaboration.

2.1.5 Identify Individual Module Deliverables

Catalog of Modules:

Employee Module:

This module allows employees to request time off through a simple application form which includes useful information for the employee. It includes a personal dashboard which enables employees to see the status of their current requests, their leave history, and current leave balance.

Manager Module:

Managers access the module to manage leave requests by approving/disapproving leave requests. It includes a team leave calendar that shows a visual of team members' time off requests together, which will help managers plan and manage personnel resource availability.

HR/Admin module:

This module is for HR and administrative personnel, providing them with detailed information and administration tools to manage and setup organizational leave styles. This includes user account management functions to enable HR to add/edit/end user accounts or modify policies. Furthermore, the HR/Admin module includes options for report and analytics that will allow HR to create precise leave reports and get the intelligence they require for organizational workforce planning.

Working:

The Employee Module allows employees to quickly and easily submit leave requests, check the status of their requests in real-time, and check their entire leave history for transparency. This allows employees to have more control and more awareness of their leave records.

The Manager Module allows managers to quickly review and accept or reject leave requests submitted by their team members. The Manager Module also provides an overall view of the

teams leave balances and an integrated calendar, allowing for total team member availability with ease of monitoring and reporting for effective workforce management.

The HR/Admin Module allows administrators to configure and amend leave policies in accordance with the rules of the organization. It also manages the roles and permissions of users to maintain appropriate access controls. This module also generates detailed leave reports, which helps facilitate data-driven decision-making and good workforce planning.

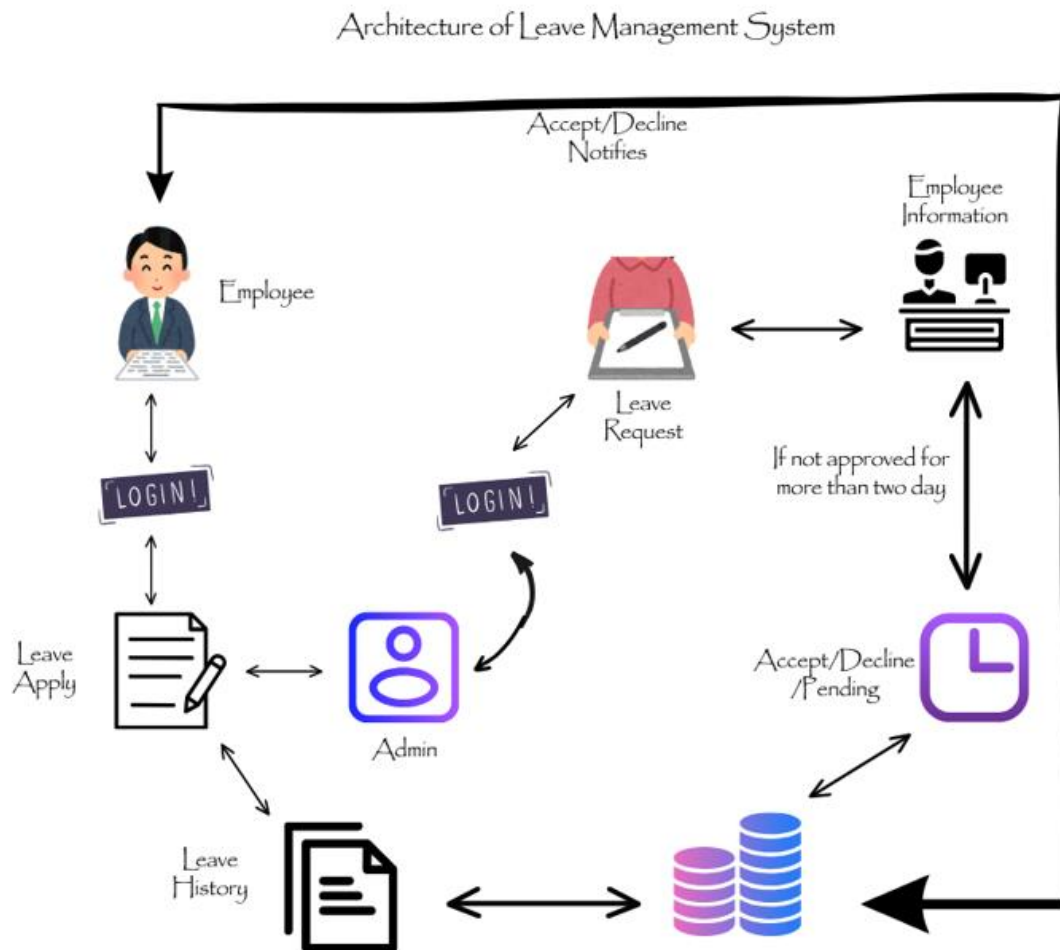
The modular design allows complete functionality of the "Easy Leave Management" system to take advantage of the best use case, while allowing rework with a development and implementation methodology.

2.2 References

1. <https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document>
2. <https://docs.samarth.ac.in/docs/employee-services/leave-management-system/>
3. <https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document>
4. <https://www.swiftelearningservices.com/leave-management-system-a-complete-guide/>

3. DATA MODELING

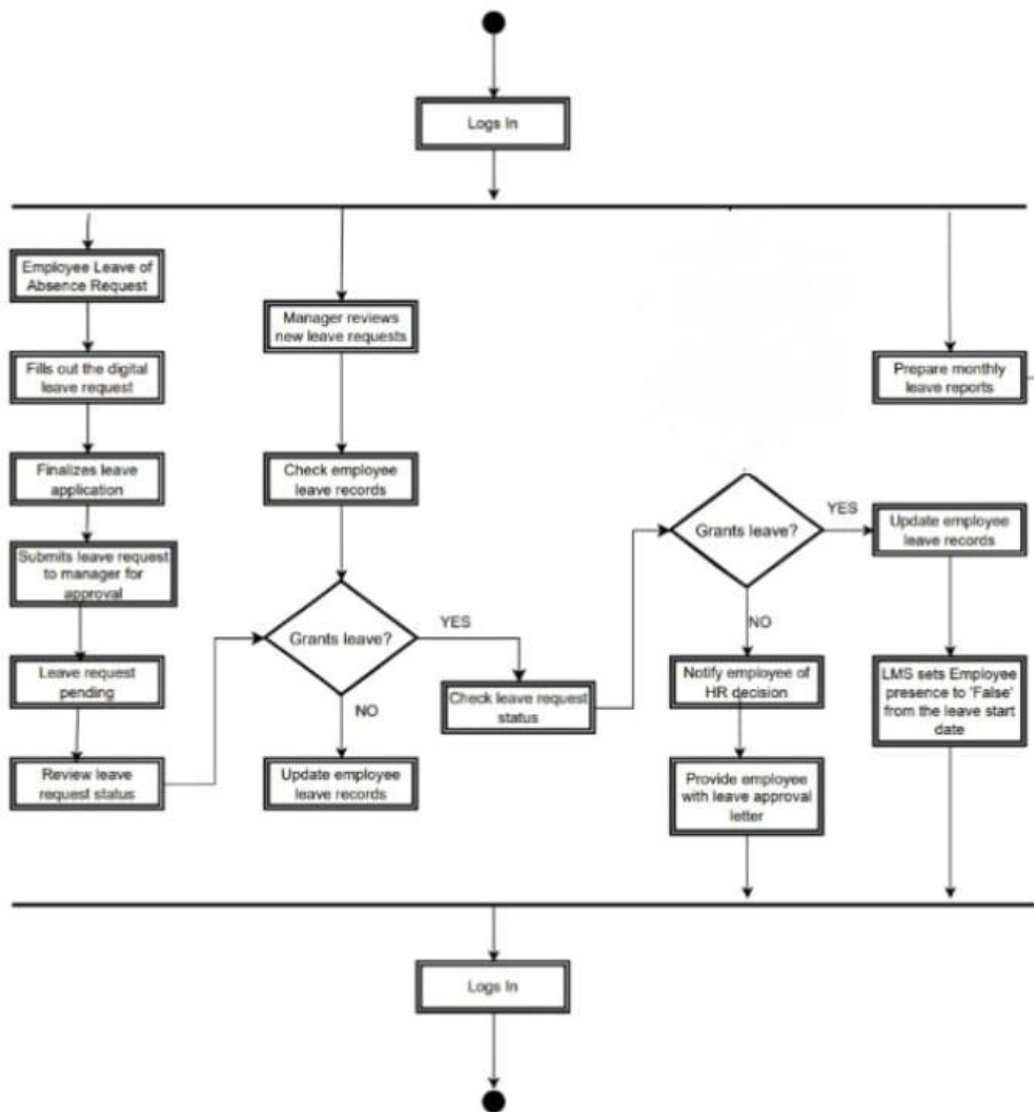
3.1 Architecture diagram



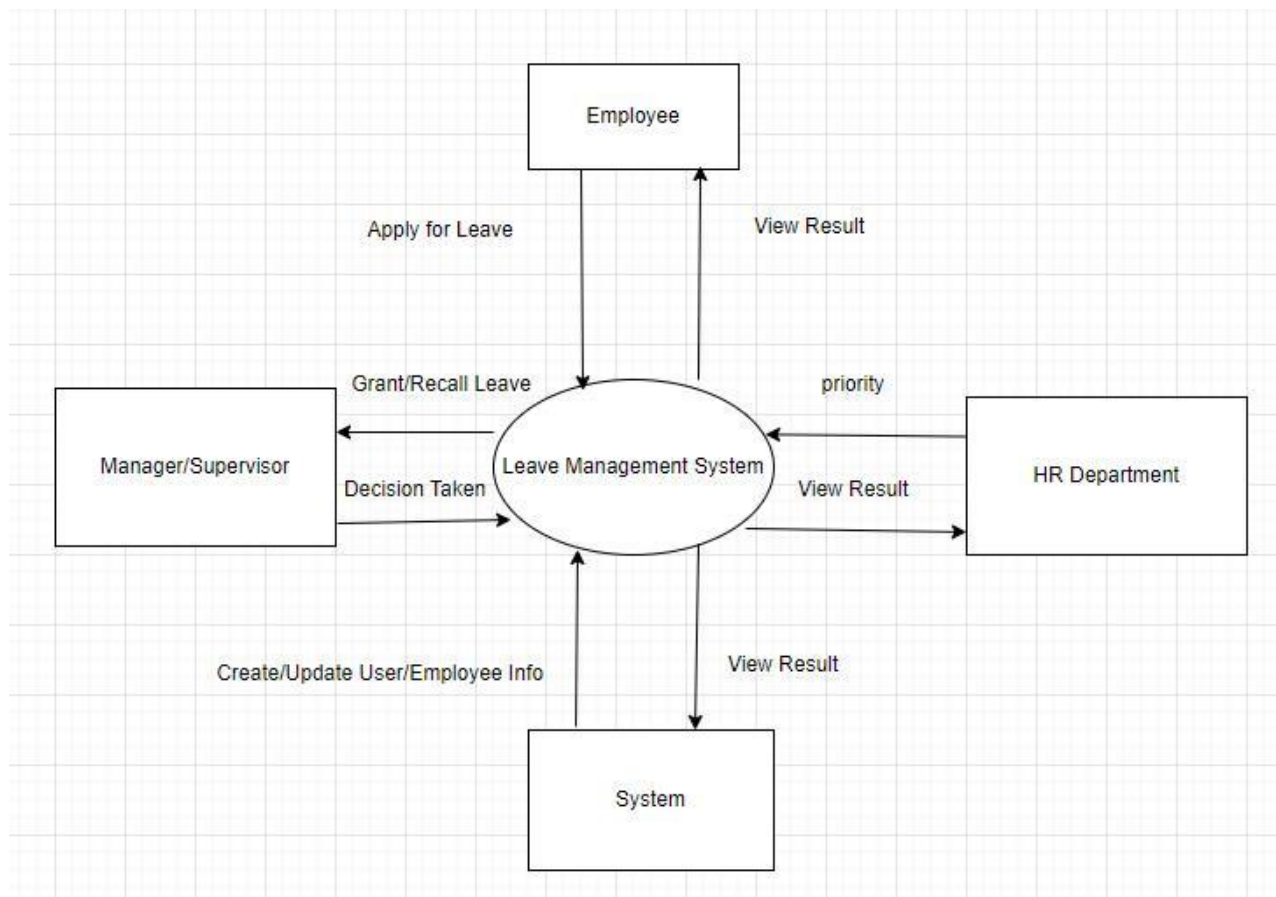
3.2 Use Case diagram



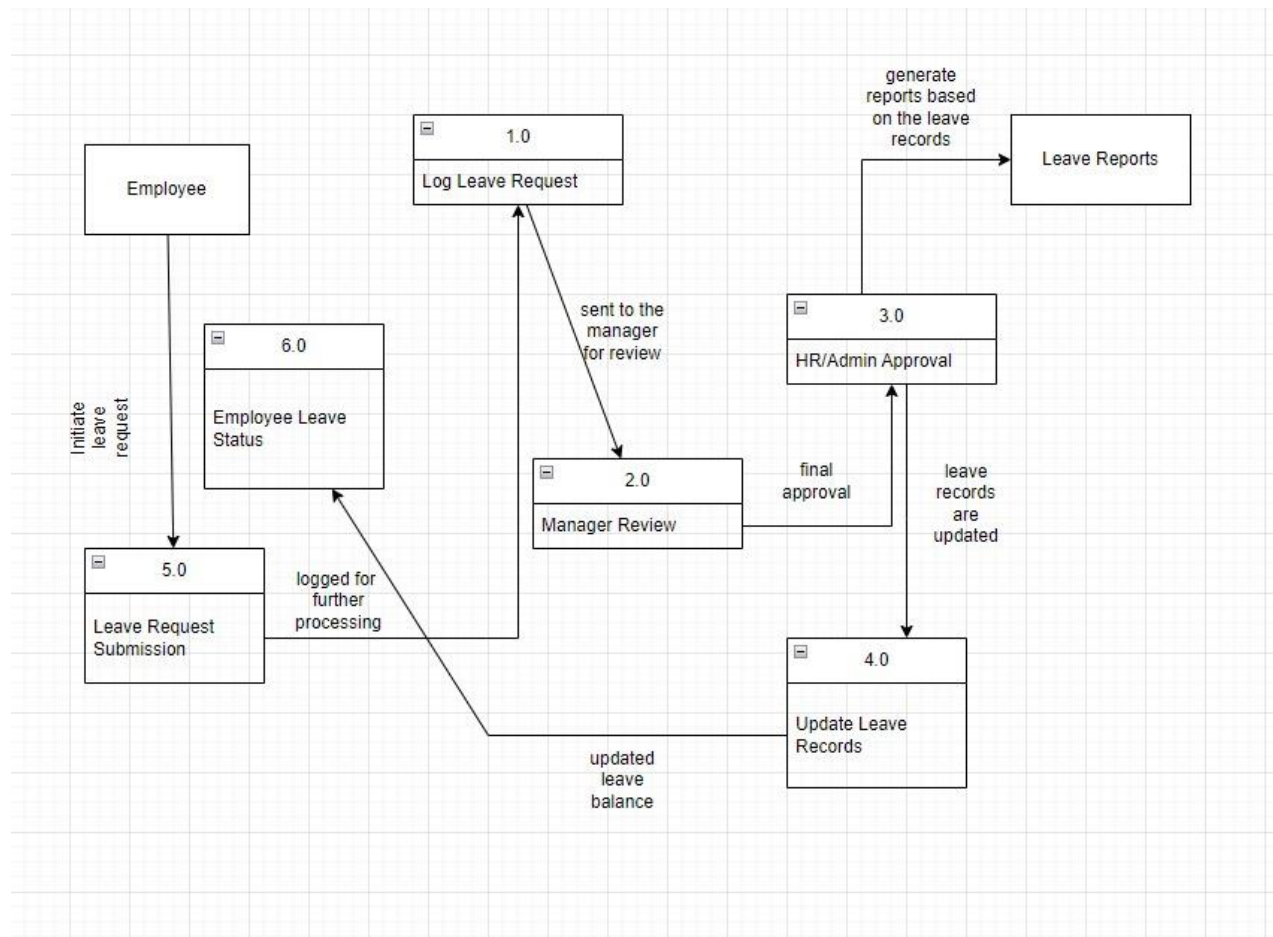
3.3 Activity Diagram (Single / Module-wise)



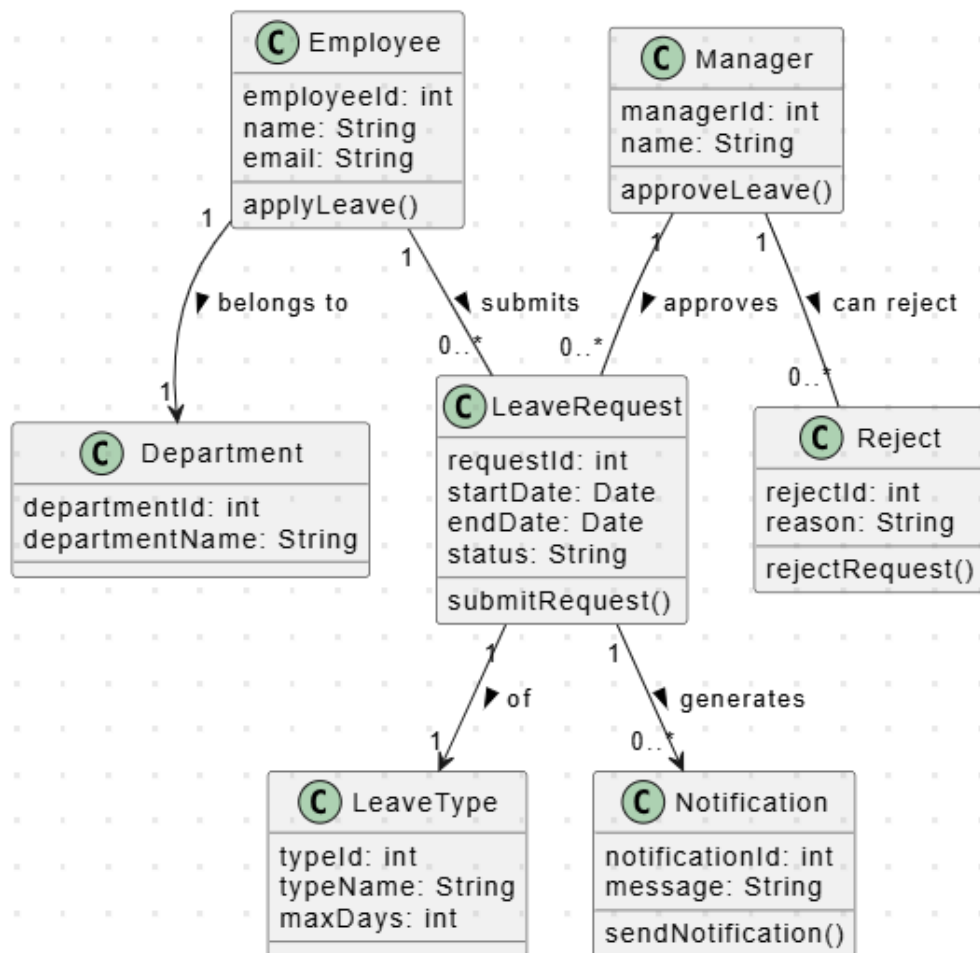
3.4.1 Context Diagram (Logical)



3.4.2 Data Flow Diagram (Physical)

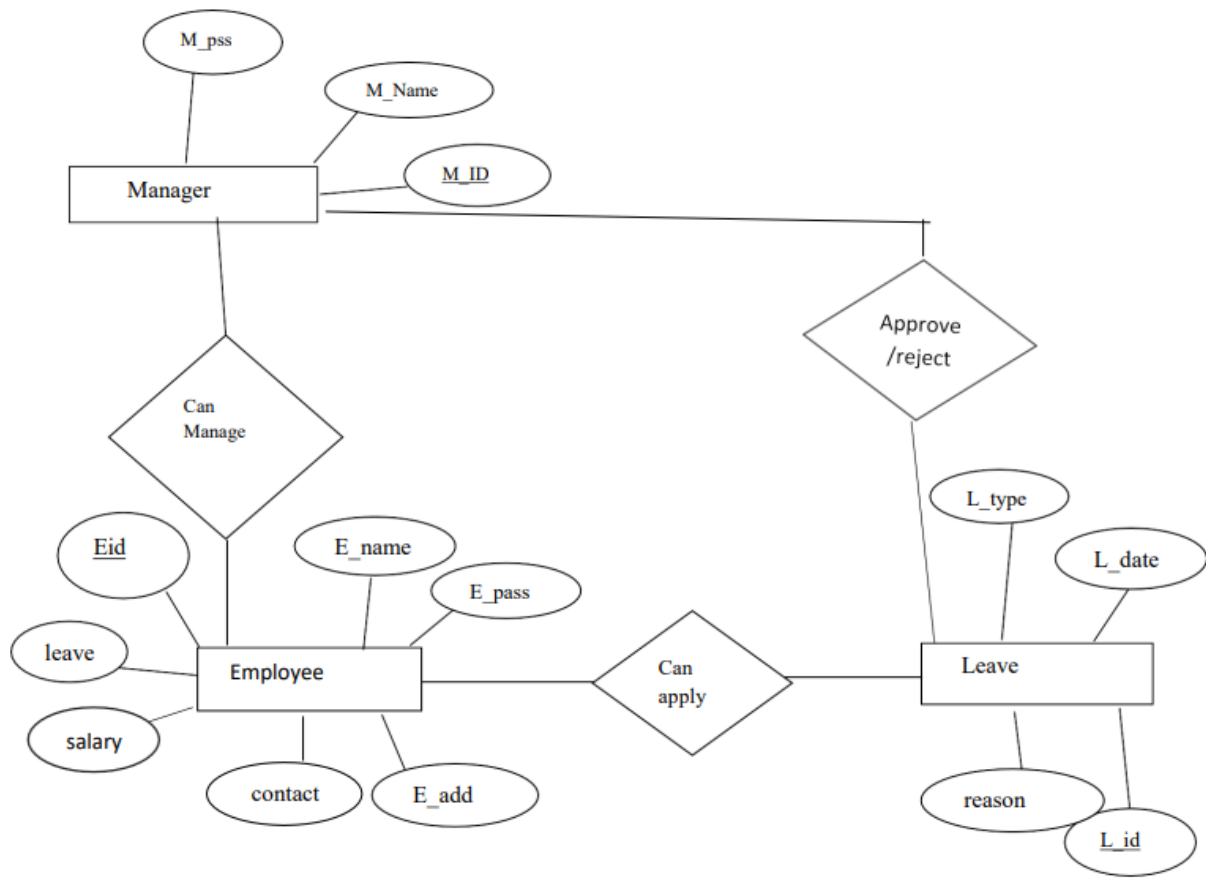


3.5 Class Diagram

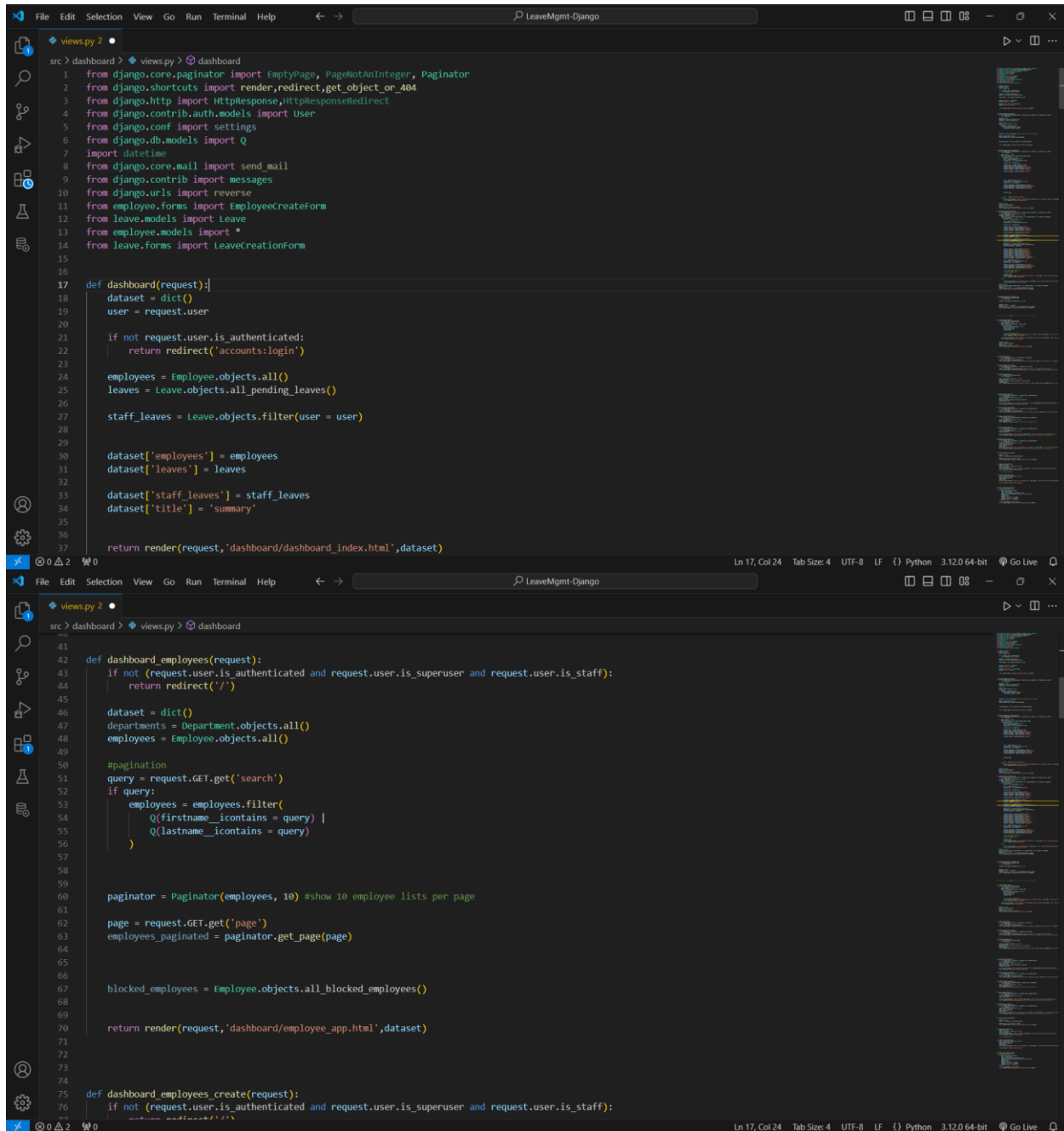


4. Development-Database Structure

4.1 E-R Diagram



4.2 Code

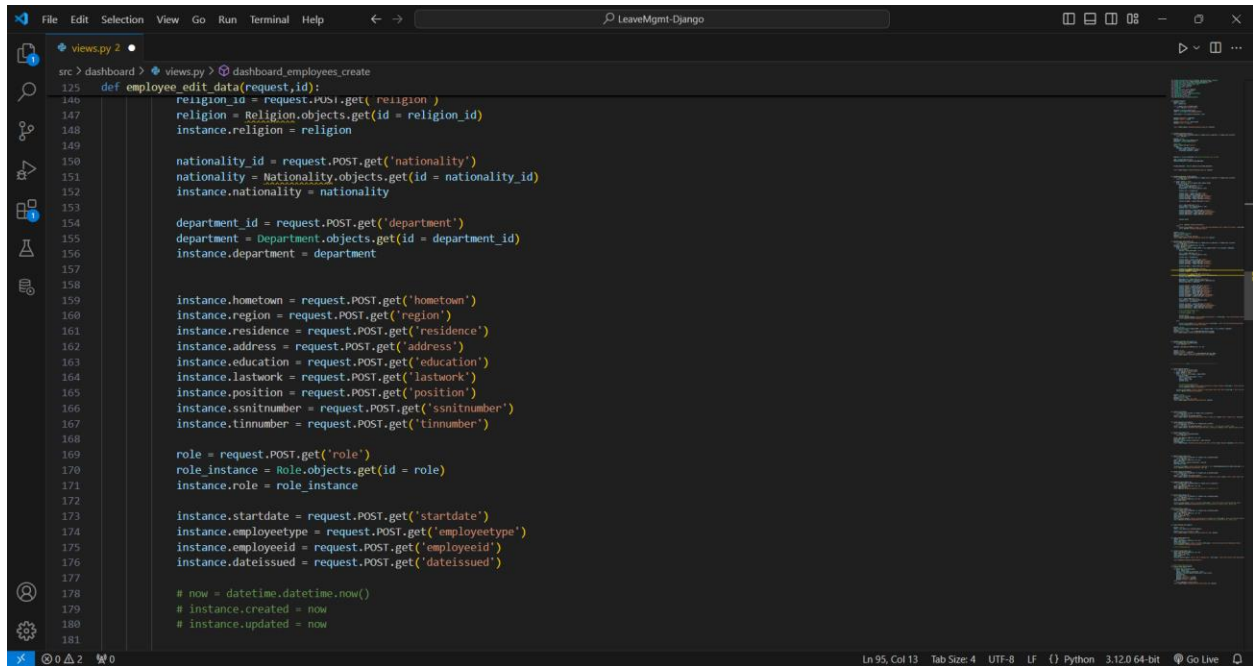


```
src > dashboard > views.py > dashboard
1 from django.core.paginator import EmptyPage, PageNotAnInteger, Paginator
2 from django.shortcuts import render, redirect, get_object_or_404
3 from django.http import HttpResponseRedirect
4 from django.contrib.auth.models import User
5 from django.conf import settings
6 from django.db.models import Q
7 import datetime
8 from django.core.mail import send_mail
9 from django.contrib import messages
10 from django.urls import reverse
11 from employee.forms import EmployeeCreateForm
12 from leave.models import Leave
13 from employee.models import *
14 from leave.forms import LeaveCreationForm
15
16
17 def dashboard(request):
18     dataset = dict()
19     user = request.user
20
21     if not request.user.is_authenticated:
22         return redirect('accounts:login')
23
24     employees = Employee.objects.all()
25     leaves = Leave.objects.all_pending_leaves()
26
27     staff_leaves = Leave.objects.filter(user = user)
28
29
30     dataset['employees'] = employees
31     dataset['leaves'] = leaves
32
33     dataset['staff_leaves'] = staff_leaves
34     dataset['title'] = 'summary'
35
36
37     return render(request, 'dashboard/dashboard_index.html', dataset)
```

```
src > dashboard > views.py > dashboard
41
42 def dashboard_employees(request):
43     if not (request.user.is_authenticated and request.user.is_superuser and request.user.is_staff):
44         return redirect('/')
45
46     dataset = dict()
47     departments = Department.objects.all()
48     employees = Employee.objects.all()
49
50     #pagination
51     query = request.GET.get('search')
52     if query:
53         employees = employees.filter(
54             Q(firstname__icontains = query) |
55             Q(lastname__icontains = query)
56         )
57
58
59
60     paginator = Paginator(employees, 10) #show 10 employee lists per page
61
62     page = request.GET.get('page')
63     employees_paginated = paginator.get_page(page)
64
65
66     blocked_employees = Employee.objects.all_blocked_employees()
67
68
69
70     return render(request, 'dashboard/employee_app.html', dataset)
71
72
73
74
75 def dashboard_employees_create(request):
76     if not (request.user.is_authenticated and request.user.is_superuser and request.user.is_staff):
77         return redirect('/')
78
79     form = EmployeeCreateForm()
80     if request.method == 'POST':
81         form = EmployeeCreateForm(request.POST)
82         if form.is_valid():
83             employee = form.save()
84             messages.success(request, 'Employee created successfully')
85             return redirect('dashboard_employees')
```

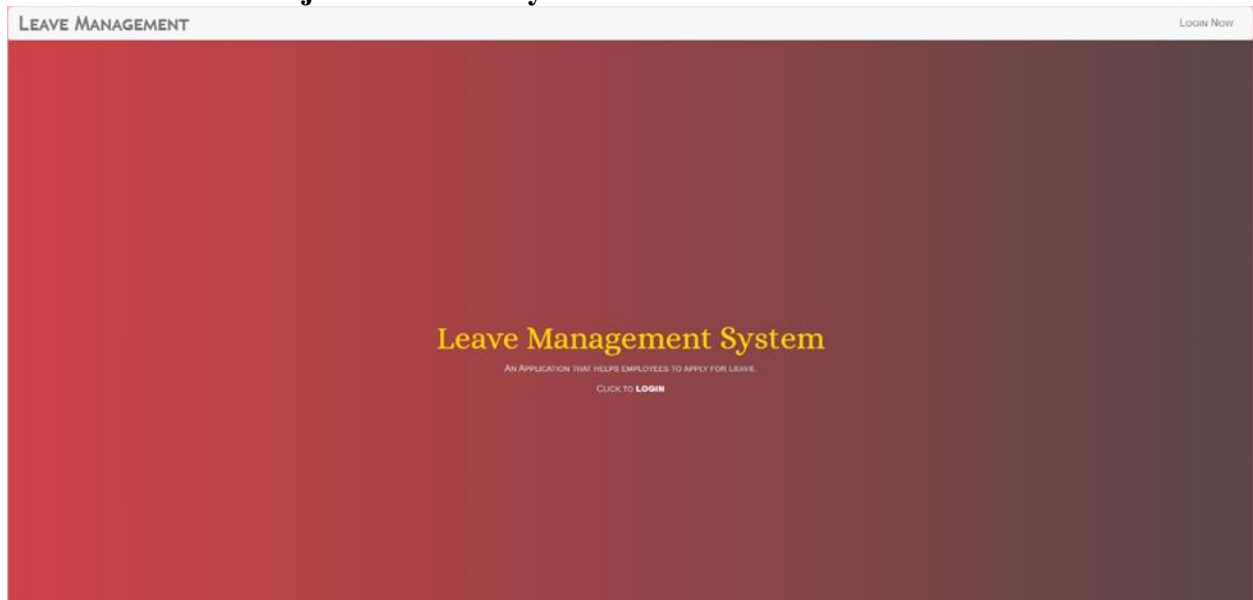


```
File Edit Selection View Go Run Terminal Help ← → LeaveMgmt-Django
views.py 2
src > dashboard > views.py > dashboard_employees_create
75 def dashboard_employees_create(request):
76     if not (request.user.is_authenticated and request.user.is_superuser and request.user.is_staff):
77         return redirect('/')
78
79     if request.method == 'POST':
80         form = EmployeeCreateForm(request.POST,request.FILES)
81         if form.is_valid():
82             instance = form.save(commit = False)
83             user = request.POST.get('user')
84             assigned_user = User.objects.get(id = user)
85
86             instance.user = assigned_user
87
88             instance.title = request.POST.get('title')
89             instance.image = request.FILES.get('image')
90             instance.firstname = request.POST.get('firstname')
91             instance.lastname = request.POST.get('lastname')
92             instance.othername = request.POST.get('othername')
93
94             instance.birthdate = request.POST.get('birthdate')
95
96
97             role = request.POST.get('role')
98             role_instance = Role.objects.get(id = role)
99             instance.role = role_instance
100
101             instance.startdate = request.POST.get('startdate')
102             instance.employeetype = request.POST.get('employeetype')
103             instance.employeeid = request.POST.get('employeeid')
104             instance.dateissued = request.POST.get('dateissued')
105
106
107             instance.save()
108
109
110
Ln 95, Col 13 Tab Size: 4 UTF-8 LF ( ) Python 3.12.0 64-bit Go Live
File Edit Selection View Go Run Terminal Help ← → LeaveMgmt-Django
views.py 2
src > dashboard > views.py > dashboard_employees_create
75 def dashboard_employees_create(request):
111         return redirect('dashboard:employees')
112     else:
113         messages.error(request,'Trying to create duplicate employees with a single user account ',extra_tags = 'alert alert-warning alert-dismissible show')
114         return redirect('dashboard:employeecreate')
115
116
117
118     dataset = dict()
119     form = EmployeeCreateForm()
120     dataset['form'] = form
121     dataset['title'] = 'register employee'
122     return render(request,'dashboard/employee_create.html',dataset)
123
124
125 def employee_edit_data(request,id):
126     if not (request.user.is_authenticated and request.user.is_superuser and request.user.is_staff):
127         return redirect('/')
128     employee = get_object_or_404(Employee, id = id)
129     if request.method == 'POST':
130         form = EmployeeCreateForm(request.POST or None,request.FILES or None,instance = employee)
131         if form.is_valid():
132             instance = form.save(commit = False)
133
134             user = request.POST.get('user')
135             assigned_user = User.objects.get(id = user)
136
137             instance.user = assigned_user
138
139             instance.image = request.FILES.get('image')
140             instance.firstname = request.POST.get('firstname')
141             instance.lastname = request.POST.get('lastname')
142             instance.othername = request.POST.get('othername')
143
144             instance.birthdate = request.POST.get('birthdate')
145
146             religion_id = request.POST.get('religion')
```



```
src > dashboard > views.py > dashboard_employees_create
125 def employee_edit_data(request, id):
140     religion_id = request.POST.get('religion')
147     religion = Religion.objects.get(id = religion_id)
148     instance.religion = religion
149
150     nationality_id = request.POST.get('nationality')
151     nationality = Nationality.objects.get(id = nationality_id)
152     instance.nationality = nationality
153
154     department_id = request.POST.get('department')
155     department = Department.objects.get(id = department_id)
156     instance.department = department
157
158     instance.hometown = request.POST.get('hometown')
159     instance.region = request.POST.get('region')
160     instance.residence = request.POST.get('residence')
161     instance.address = request.POST.get('address')
162     instance.education = request.POST.get('education')
163     instance.lastwork = request.POST.get('lastwork')
164     instance.position = request.POST.get('position')
165     instance.ssnitnumber = request.POST.get('ssnitnumber')
166     instance.tinnumber = request.POST.get('tinnumber')
167
168     role = request.POST.get('role')
169     role_instance = Role.objects.get(id = role)
170     instance.role = role_instance
171
172     instance.startdate = request.POST.get('startdate')
173     instance.employeeetype = request.POST.get('employeeetype')
174     instance.employeeid = request.POST.get('employeeid')
175     instance.dateissued = request.POST.get('dateissued')
176
177     # now = datetime.datetime.now()
178     # instance.created = now
179     # instance.updated = now
180
181
```

Demonstration of Project Functionality:



User Login

Username*

username

Password*

password

Login

Dashboard - LMS | approved leav

127.0.0.1:8000/dashboard/leaves/approved/all/

admin1

LEAVE MANAGEMENT

Dashboard

Employee Section

Leave Section

Leave Users Employee

APPROVED LEAVES

Total Approved Leaves - 6

USER	TYPE	DAY(S)	STATUS	ACTIONS
CHARLES	QUARANTINE	24	APPROVED	VIEW UNAPPROVE
CHRIS	SABBATICAL	4	APPROVED	VIEW UNAPPROVE
LIAMMOORE	SICK	3	APPROVED	VIEW UNAPPROVE
KATHRYN	EMERGENCY	1	APPROVED	VIEW UNAPPROVE
KATHRYN	SICK	2	APPROVED	VIEW UNAPPROVE
KATHRYN	STUDY	3	APPROVED	VIEW UNAPPROVE

LEAVE MANAGEMENT

DASHBOARD

EMPLOYEE SECTION

LEAVE SECTION

Leave • Users • Employee •

127.0.0.1:8000/dashboard/leaves/pending/all/

admin1 •

PENDING LEAVES

USER	TYPE	DAY(S)	STATUS	ACTIONS
SOHA	EMERGENCY	7	PENDING	View
ONDY	BEREAVEMENT	4	PENDING	View
CHRS	SICK	2	PENDING	View

LEAVE MANAGEMENT

DASHBOARD

LEAVE SECTION

Leave •

127.0.0.1:8000/dashboard/leave/apply/

soha •

APPLY FOR LEAVE

Start Date*

leave start date is on ...

End Date*

coming back on ...

Leavetype*

Sick Leave

Reason

Request Leave

