Project Report: Leave Management System Using Django

Cover Page

Project Title: Leave Management System
Technology Used: Django Framework (Python)
Submitted by: Nadeem Alam Ansari
College: SRMCEM, LICKNOW
Guide: SELF
Department: Information Technology
Date: 22/03/2023

Acknowledgment

I express my sincere gratitude to Dr. Ashish Baiswar for his invaluable guidance, timely suggestions, and encouragement throughout this project. I would also like to thank Mr. Ajay Kumar Srivastava, Head of Department, and all faculty members of the IT Department for their consistent support.

Abstract

The Leave Management System is a web-based application developed using the Django framework. The project aims to simplify the leave request and approval process within an organization or institution. It allows employees to submit leave applications and managers to approve or reject them. Admin users can monitor and control the system. It reduces manual paperwork and enhances efficiency, accuracy, and transparency.

Table of Contents

14. References
15. Appendix

---

1. Introduction

The Leave Management System streamlines the process of managing leaves within an organization. Traditionally, leave processes involve paperwork, approvals, and delays. This project digitizes and automates these tasks using Django, a high-level Python web framework. The goal is to develop a platform where employees can apply for leave and supervisors can manage these requests effectively.

Objectives:

- Provide a centralized system for leave management
- Reduce paperwork and errors
- Enable faster approvals
- Maintain leave records and history

Scope:
The system can be used by educational institutions, small and medium enterprises, and corporate organizations. It is scalable and can be enhanced with additional modules.

---

2. Literature Survey

Manual leave systems are inefficient and prone to errors. Several HR software platforms like Keka, GreytHR, and Zoho People provide automated leave systems but are often complex or expensive. Django offers a flexible, open-source alternative. Research shows that digitization improves transparency, reduces workload, and provides valuable insights for management decisions.

---

3. System Requirements

Hardware Requirements:

- Processor: Intel i3 or higher
- RAM: Minimum 4 GB
- Storage: 100 GB HDD or SSD

Software Requirements:

- Operating System: Windows/Linux/Mac
- Backend: Python 3.x, Django 4.x
- Frontend: HTML, CSS, JavaScript, Bootstrap
- Database: SQLite / MySQL

- Tools: VS Code, Git, Browser

---

4. Technology Used

- **Python:** Programming language for logic and backend
- **Django:** Web framework for rapid development
- **HTML/CSS:** Structure and design
- **Bootstrap:** UI styling
- **SQLite:** Lightweight database
- **Git:** Version control

---

5. System Analysis

Problem Definition:
Manual systems are time-consuming, error-prone, and lack tracking.

Feasibility Study:

- **Technical:** Django is suitable for web applications
- **Economic:** Cost-effective with open-source tools
- **Operational:** Easy to use, efficient, and user-friendly

---

6. System Design

Use Case Diagram

- Users: Admin, Manager, Employee
- Actions: Apply Leave, Approve/Reject, View Status

ER Diagram

- Tables: Users, Leave Requests, Leave Types

Flowcharts and Class Diagrams

- Flow of data from UI to backend and database

---

7. Modules Description

1. **Admin Panel**
    - User Management
    - Leave Type Management

2. **Employee Panel**
    - Apply for Leave
    - View Leave Status
3. **Manager Panel**
    - View and Respond to Leave Requests
4. **Authentication System**
    - Login, Logout, Password Reset

---

8. Implementation

- Django Project created using `django-admin startproject lms`
- Apps: users, leaves, dashboard
- Models defined in `models.py`
- Forms built using Django Forms
- Templates use Bootstrap for UI
- URLs and Views manage request flow

---

9. Testing

Test Cases:

- TC001: Login with valid credentials - Pass
- TC002: Submit leave form - Pass
- TC003: Approve/Reject leave - Pass
- TC004: Unauthorized access - Fail (expected)

Testing Methods:

- Unit Testing
- Manual Interface Testing

---

10. Results and Discussion

The system performed well across different modules. Employees were able to apply for leave with ease, and managers could review them quickly. Admin controls helped maintain data integrity. The dashboard provided real-time updates.

---

11. Limitations and Challenges

- No mobile app version
- Requires stable internet access
- Basic notification system (no emails/SMS)

---

12. Future Scope

- Mobile App Integration
- Email/SMS Notifications
- Calendar Integration
- Advanced Analytics Dashboard

---

13. Conclusion

This project provided hands-on experience with Django, Python, and database systems. The Leave Management System meets the project objectives and lays a foundation for further development. It offers a practical, efficient solution for digitizing organizational processes.

---

14. References

1. https://docs.djangoproject.com/
2. https://getbootstrap.com/
3. Stack Overflow - community solutions
4. GitHub Django Projects

---

15. Appendix

- `models.py`, `views.py`, `forms.py`, `urls.py` excerpts
- Django commands: `makemigrations`, `migrate`, `runserver`
- Screenshot placeholders: login page, leave form, dashboard

End of Report