Human Resource Management System (HRMS)

# Table of Contents

1. Abstract

2. Introduction

3. Technologies Used

4. Database Design

5. Backend API Structure

6. Challenges Faced

7. Conclusion

# 1. Abstract

This document describes the backend development of a Human Resource Management System (HRMS). The system handles employee management, attendance tracking, payroll processing, job applications, and user authentication. It uses FastAPI for API creation and PostgreSQL for database management, with a clean separation between API logic and front-end rendering.

# 2. Introduction

The HRMS is designed to simplify and automate HR tasks in an organization. It supports the management of employees, departments, positions, leave, payroll, projects, and job postings. The backend was developed using Python and FastAPI, offering RESTful APIs that communicate with a PostgreSQL database.

# 3. Technologies Used

- Backend Framework: FastAPI

- Frontend Rendering (Optional): Flask with Jinja2

- Database: PostgreSQL

- ORM: psycopg2 (Raw SQL queries used)

- Authentication: Session-based login

- Language: Python

- Others: Passlib for password hashing, HTML templates for UI

# 4. Database Design

Key Tables:

- employees: Stores employee details

- departments: Department-wise segregation

- positions: Designations or job roles

- attendance: Daily attendance log

- leave\_requests: Employee leave applications

- payroll: Payroll generation and salary details

- salary\_structures: Salary breakdowns based on roles

- projects: Project assignments and tracking

- employee\_projects: Links employees to projects

- job\_postings: Recruitment-related job advertisements

- candidates: External candidate details

- applications: Candidate applications for jobs

Each table includes necessary foreign key relationships, constraints, and status fields for proper data integrity.

# 5. Backend API Structure

Authentication APIs:

- /api/signup – Register a new user

- /api/login – Authenticate user credentials

- /api/logout – Logout and end session

Employee Management APIs:

- /api/employees – List all employees

- /api/employees/{id} – View, update, or delete employee details

Department & Position APIs:

- /api/departments – CRUD operations for departments

- /api/positions – CRUD for job roles

Attendance APIs:

- /api/attendance – Add/view daily attendance

- /api/attendance/{id} – Edit or delete attendance record

Payroll APIs:

- /api/payroll – Generate payroll

- /api/payroll/{id} – View/edit payroll data

Job Posting & Applications:

- /api/job\_postings – Create and manage job listings

- /api/applications – Handle job applications from candidates

Project Assignment:

- /api/employee\_projects – Assign employees to projects

All APIs follow REST conventions and return JSON responses.

# 6. Challenges Faced

- Integrating Flask frontend with FastAPI backend in a hybrid project

- Maintaining relational integrity in PostgreSQL while handling complex joins

- Session-based authentication alongside API-based communication

- Designing flexible but normalized database tables to accommodate future HRMS expansion

- Managing file structure cleanly while separating front-end and back-end logic

# 7. Conclusion

The HRMS project provided a strong foundation in full-stack backend development using FastAPI and PostgreSQL. With modular APIs and a cleanly structured database, the system is scalable and maintainable. It effectively supports the core HR operations and can be expanded in the future to include analytics, notifications, or mobile integration.