CHAPTER I: INTRODUCTION

1.1 Background

Effective workforce management is becoming increasingly difficult for organizations in today's rapidly evolving business environment. Human resource (HR) departments are in charge of hiring, retaining, and maintaining general employee satisfaction in addition to keeping track of employee records. Traditional HR systems often handle only administrative tasks, missing the opportunity to use data to predict crucial workforce trends.

Employee churn, or the rate at which workers depart an organization, is a major worry for many businesses. High churn can result in lost experienced talent, lower productivity, and higher recruitment expenses. Organizations can enhance retention and maintain team stability by recognizing trends in employee departures and anticipating potential departures.

Systems that combine role-based access, predictive analytics, real-time insights, and employee data management are becoming more and more necessary to satisfy contemporary HR requirements. While still in its early stages, advanced analytics in HR has demonstrated great promise in enhancing attrition forecasts and pinpointing important retention elements (Dutta et al., 2024). This method improves decision-making, lowers employee attrition, and increases productivity.

1.2 Problem Statement

Adapting to changes in the workforce and managing employee data are challenges for many organizations. The inability to forecast employee turnover results in higher expenses and disruptions, and HR teams frequently lack advanced tools for rapid data analysis (Alabi et al., 2024). Delays and mistakes are also caused by manual leave and record management systems (Gaur, 2020).

Three major issues are addressed by this project:

1. Analyzing and visualizing employee data is a challenge for HR teams.
2. Leave and employee records are managed manually.
3. There is no system to predict which employees may leave.

1.3 Objectives

This project aims to address key challenges faced by HR departments through clear and focused objectives. Specifically, the objectives are:

1. To develop an integrated HRM system for managing employee information.
2. To create a machine learning system to predict employee churn.
3. To automate leave management through a web-based application.

1.4 Literature Review

The Literature Review section of this report examines existing research, tools, and systems related to HR management and employee attrition prediction. Its objective is to understand current methodologies, identify gaps in traditional systems, and justify the use of machine learning for proactive churn management. This background helps position the developed system within the context of ongoing technological advancements in HR analytics.

1.4.1 Overview of Existing Systems

Employee Analytics and Churn Prediction Systems are important tools for modern human resource management. Popular software like OrangeHRM and BambooHR help organizations track employee data and show basic trends in staff turnover. However, these systems usually do not provide advanced prediction of which employees might leave.

Larger platforms such as Workday and SAP SuccessFactors use more advanced methods, including some machine learning, to estimate the risk of employees leaving. Still, these features often need extra setup and are not always easy to use for every organization.

Most of these systems mainly use basic HR data, such as employee age or years of service, and do not include information from emails or other unstructured sources. They also do not give real-time predictions and sometimes make it hard for managers to understand why an employee is at risk of leaving.

In summary, while current systems help organizations watch employee turnover, their ability to accurately predict and explain employee churn is still limited. More advanced methods from academic research are not yet widely used in these commercial tools.

1.4.2 Technologies used in Similar Systems

Similar HR systems use a mix of cloud-based architectures, REST APIs, and technologies like PHP, Java, JavaScript, and proprietary platforms to enable modular, scalable, and mobile-friendly HR solutions.

Table 1.1: Technologies Used in Similar Systems

Product	Main Technologies Used	Notes
OrangeHRM	PHP, MySQL, Apache, Vue.js	Open-source, web-based
BambooHR	JavaScript (React), Python, AWS	Cloud-based, modern web stack
Workday	Proprietary cloud tech, APIs, AI tools	Enterprise, cloud-only
SAP SuccessFactors	SAP Cloud, SAP HANA, APIs, AI/ML	Enterprise, SAP ecosystem

1.4.3 Theoretical Background

In academic and technical writing, the theoretical framework is the set of ideas, models, and tools that support a project. It gives you the big picture that helps you understand why certain tools or methods were chosen and how they work together to reach the project's goals. The ideas behind the Employee Analytics and Churn Prediction System come from both machine learning and web application development.

The web app is built on Flask, a lightweight web framework for Python. Flask is well-known for being simple to use and flexible, which makes it great for quickly building web-based systems. Because it has a simple design, developers can only add the parts they need, which makes the codebase efficient and easy to keep up with. Flask handles routing and processing requests, and it works well with databases and other backend services.

The system uses MySQL, a relational database management system, to store and manage data. MySQL puts data into structured tables so that it is easier to search, update, and get employee records, leave requests, and analytics data. SQL is great for manipulating data because it supports complex queries needed for HR analytics and makes sure that the data is always correct.

JavaScript makes things happen dynamically, Tailwind CSS makes the styling responsive and utility-first, and HTML gives the user interface structure. Tailwind CSS has a lot of utility classes, which cuts down on the need for custom CSS and makes it easy to make a user interface quickly. JavaScript is used to make real-time updates on the client side better, control form submissions, and make things more interactive.

Chart.js is used by the system to show data in a way that makes sense. The JavaScript library Chart.js lets you make interactive and eye-catching charts right in the browser, like bar graphs and pie charts. HR users can now quickly look at departmental distributions, workforce trends, and attrition rates thanks to clear graphical representations.

The system's predictive part is powered by a machine learning model made with scikit-learn, a powerful library for data mining and analysis in Python. Scikit-learn has tools for preparing data, training models, and checking how well they work. We use a RandomForestClassifier in this project to predict employee turnover based on a number of factors. We use a standard 80/20 train-test split to train and test the model, which makes sure that it will work correctly before we put it into the live system.

These technologies work together to give the Employee Analytics and Churn Prediction System a modern web application framework that makes it possible to safely manage data, do smart analytics, and make smart predictions.

1.4.4 Comparative Study

When compared to popular HR platforms like Workday, SAP, BambooHR, and OrangeHRM, the Employee Analytics and Churn Prediction System has a clear edge in predictive analytics and customization. OrangeHRM and BambooHR are great for core

HR tasks and easy-to-use workflows, but their analytics are mostly descriptive and don't include machine learning for predicting churn. SAP and Workday are strong and have a lot of features, but they are made for big businesses and can be costly and hard to set up and customize.

On the other hand, this project stands out because it directly adds a machine learning model that has been trained to predict employee turnover to the HR workflow. The system gives very accurate and useful insights based on real organizational data by using a Random Forest Classifier and scikit-learn. Because it is built on Flask and modern web technologies and has an open-source, modular architecture, it is great for businesses that want data-driven HR solutions without the hassle of enterprise platforms.

The Employee Analytics and Churn Prediction System is a little better than traditional HR systems at coming up with targeted, early retention strategies using modern predictive analytics. However, traditional HR systems are better at handling a lot of data and growing.

1.4.5 Research Papers and Related Work

Employee analytics and churn prediction systems are now essential parts of modern human resource management. Both commercial platforms and academic models have made progress in the field in recent years. These systems are meant to help businesses understand how their employees work together, keep an eye on how engaged they are, and deal with turnover risks before they happen.

Many small and medium-sized businesses use open-source platforms like OrangeHRM for basic HR management and analytics. These systems usually give descriptive statistics and historical trend analysis about employee turnover, but they don't have advanced predictive features or the ability to connect to data from outside sources (Ekawati, 2019).

Cloud-based tools like BambooHR go beyond traditional HR tasks by giving you dashboards that show turnover rates and point out possible risk factors. BambooHR mostly looks at past data and simple regression models to find employees who are at risk. But it still doesn't have much predictive power; it focusses more on showing trends than on giving each person a real-time churn risk score (Fallucchi et al., 2020).

Kannimuthu and Premalatha (2024) looked at a number of models and found that deep learning algorithms were better at predicting turnover than traditional methods. This shows how important it is to find the main factors that affect attrition. Another enterprise-level solution, SAP SuccessFactors, comes with pre-set metrics and dashboards for analysing turnover. However, its ability to predict things isn't real-time and often depends on people interpreting data by hand (Gabrani & Kwatra, 2018).

Combining different types of data, such as internal communications and external labour market trends, is very important for making HR strategies more sustainable. Unstructured data is being analysed with advanced technologies like

machine learning and natural language processing. This makes things more efficient and encourages eco-friendly HR practices. HR practices still need to find a balance between data-driven strategies and the human element (Asfahani, 2024). Maan and Maan (2023) used Shapley values to explain feature importance and stressed how important it is for models to be easy to understand when predicting customer churn. These methods are meant to make things more clear and help create flexible risk thresholds. This will allow HR professionals to use the models to figure out the best ways to keep employees based on the specific factors they find (Mohiuddin et al., 2023; Makanga et al., 2024).

There have been a number of studies that looked into using machine learning models to predict when employees will leave. In many studies, Random Forest was always the best algorithm, with an accuracy rate of 97.5% (Alamsyah & Salma, 2018; Musanga et al., 2022). Bandyopadhyay and Jadhav (2021) also found that Random Forest was better than SVM and Naive Bayes. Other good models were Logistic Regression, Decision Trees, and Gradient Boosting Machines (Musanga et al., 2022). BI-driven HR practices have been shown to help keep employees and improve the performance of the organisation (Ara, 2025).

To sum up, Employee Analytics and Churn Prediction Systems are helping companies go from just keeping track of turnover to actually stopping it. HR teams are better able to figure out why people leave and take action before problems happen as both research and business tools get better. Using these insights to make the workplace more supportive and interesting for everyone is where the real benefit lies.

1.5 Development Methodology

The Development Methodology explains the general way the project will go from idea to completion. The Research and Development Process was part of it. This process involved looking into existing solutions and planning system features. The Project Framework made sure that the workflow was organised and repeated. Data Collection Methods are the ways that useful information was gathered to help with system functionality and analysis.

1.5.1 Research and Development Process

The project started by looking closely at how HR works now and what problems it faces, especially in companies with 50 to 1,000 employees. This first step in the research was to look at current employee management systems and studies that predicted employee turnover to find things they all had in common, things they were missing, and ways to make them better. The goal was to create a system that is useful and based on data. Recent studies have focused on using data to predict and lower employee turnover. Machine learning methods like deep neural networks, random forests, and gradient boosting have shown promising results in predicting employee turnover (Pachghare et al., 2024; Srivastava & Eachempati, 2021). To help with this, the right datasets were carefully chosen and prepared so that the machine learning model could be trained correctly to predict churn. With these new ideas and tools, the

project moved on to the development phase, which focused on making the interfaces easy to use, the backend strong, and the predictive model work smoothly with the rest of the system.

1.5.2 Project Framework

The project used the Spiral Model, which is a methodical way of improving the system over time by going through the same steps over and over again. Planning, checking for risks, developing, and testing are all parts of each cycle. The project used this framework in the following way:

1. **Planning Phase**: The first step in the project was to figure out what the system needed to do, like keeping track of employee information, handling leave requests, and predicting how many employees would leave. There was a list of all the features that were needed. Then, the project's scope was set based on the time and resources that were available.
2. **Risk Analysis Phase**: In this phase, we looked for possible problems. These included wrong predictions about churn, problems logging in, and database crashes. There were plans for early solutions, such as using hashed passwords for extra security and testing the model's performance very carefully.
3. **Engineering Phase**: The system was built in steps. It used MySQL for the database, Flask for the backend, and machine learning to make predictions. First, the login page was created. Then, one by one, employee forms, a leave system, and a churn prediction feature were added.
4. **Evaluation Phase**: Each part of the system was tested to make sure it worked right. When problems were found, changes were made. For instance, the prediction model was made more accurate, and problems with user access were fixed.

This process happened over and over again in cycles (the spiral model). Every cycle added new features and made old ones better. The spiral method helped find problems early on and make the system better over time.

1.5.3 Data Collection Methods

To build the churn prediction model, a Kaggle dataset called ‘IBM HR Analytics Employee Attrition and Performance’ was used. The original dataset has 1,470 records and 35 columns with details like age, job role, and attrition status. Some columns are removed during preprocessing to keep only the most important features for predicting employee churn.

To improve the model's performance, additional synthetic data was added, increasing the dataset to around 4,680 records. Since the data had fewer cases of employees who left, a balancing method was used to make both classes equal. This helped the model learn more fairly and avoid bias. Then, 80% was used for training and 20% for testing. For real-world simulation within the system, a subset of 20 employee records was extracted from the dataset.

1.6 Scope and Limitations

The Scope and Limitations section defines what the system is designed to achieve and the boundaries within which it operates. It outlines the key features and functions included in the project while also acknowledging any constraints, such as features that were excluded, technical limitations, or areas for future improvement. This helps set clear expectations for users and stakeholders.

1.6.1 Project Scope

The following key features and functionalities are included in this project:

1. It allows HR to perform create, read, update and delete (CRUD) operations on employee information.
2. A job application feature lets applicants apply, and HR can hire them.
3. Employees can submit leave requests, and HR can approve or reject them.
4. HR users can see visual dashboards showing employee trends and important data.
5. The system gives different access to HR and employees based on their roles.
6. To predict which employees are at risk of leaving, it uses a machine learning model.

1.6.2 Project Limitations

The project excludes certain features and functionalities that fall outside its current scope, such as:

1. It does not track employee attendance, as this is usually done with special IoT devices.
2. It does not handle salary calculations or payroll management, which are managed by financial tools.
3. It does not include detailed employee performance evaluations, which require more advanced tools.
4. It does not support recruitment processes beyond basic job applications, such as full applicant tracking systems (ATS).

1.7 Report Organization

The Report Structure and Organisation section gives an overview of how the document is set up. It lists the order in which the topics are covered, such as the project overview, methodology, system features, implementation details, testing, and evaluation.

Table 1.2: Report Structure

Chapter No.	Chapter Title	Description
CHAPTER I	**INTRODUCTON**	Provides an overview of the project's background, objectives, related works, development approach, scope, limitations, and report structure.
CHAPTER II	**SYSTEM DEVELOPMENT PROCESS**	
	Analysis	Covers requirement analysis, feasibility, and system modeling.
	Design	Presents the system's user interface and architectural design.
	Implementation	Describes the tools, modules, and coding processes used.
	Testing	Discusses the testing strategies and results.
CHAPTER III	**CONCLUSION AND RECOMMENDATIONS**	This section summarizes the project results, presents key conclusions drawn from the findings, and suggests possible improvements or future enhancements.
REFERENCES		Lists all sources and materials referenced in the report.
APPENDICES		Contains supplementary materials such as code snippets, data samples, and additional documentation.

CHAPTER II: SYSTEM DEVELOPMENT PROCESS

2.1 Analysis

In this phase, the system requirements and user needs are very clear. Looking at current processes helps us learn important things. This helps with planning the growth and makes sure the system will work as it should.

2.1.1 Requirement Analysis

The Requirement Analysis section lists the system's most important needs and features. We get input from potential users and stakeholders to find out what features are needed, how well the system should work, and any limits. This analysis helps to direct the design and development process so that the final product meets the needs of users well.

2.1.1.1 Functional Requirements

Functional requirements (FR) tell you what the system can do and how it will act in certain situations. These requirements are based on the needs of the end users—HR personnel and employees—and cover features that ensure the system performs key tasks like managing employee data, handling leave requests, and predicting employee churn. The development process is guided and the end result is guaranteed to match user expectations when these functionalities are well defined.

The role-based functional requirements of Employee Analytics and Churn Prediction System includes:

Table 2.1: HR and Employee Functions

Requirement ID	Requirement Description	Input	Output
FR-01	User Login with Role-Based Access	Username and password	System dashboard access according to user role (HR / Employee)

Table 2.2: Employee Functions

Requirement ID	Requirement Description	Input	Output
FR-02	View and Edit Employee Profile	Profile update form	Updated profile details
FR-03	Submit Leave Request	Leave request form	Leave request entry under employee profile

Table 2.3: HR Functions

Requirement ID	Requirement Description	Input	Output
FR-04	View Analytical Dashboards	User login (HR role)	Charts showing attrition trends, department stats
FR-05	Review and Manage Job Applicants	List of submitted applications	Applicant status updated to Hired or Rejected; user account created if hired
FR-06	HR adds or edits employee data	Employee details form (name, role, salary, etc.)	New or updated employee record in the database
FR-07	Approve or Reject Leave Request	Leave request with decision	Updated leave request status and email notification
FR-08	Predict Employee Churn	Employee data (internal from DB)	Churn prediction label (Yes/No)

Table 2.4: Applicant Functions

Requirement ID	Requirement Description	Input	Output
FR-09	Submit Job Application	Job application form	New applicant record created
FR-10	Receive Hiring Notification Email	Hiring decision trigger	Email sent to applicant if hired

2.1.1.2 Use Case Diagram

A use case diagram shows the functional needs of a system by showing how users interact with it.

Figure 2.1: Use-Case Diagram

2.1.1.3 Use Case Scenario

Use Case (UC) scenarios help in the clarification of functional requirements and user interactions by describing certain action sequences that users and the system take to achieve specified goals.

Table 2.5: UC-01 User Login with Role-Based Access

Field	Description
Use Case Identifier	UC-01
Primary Actor	User (HR or Employee)
Secondary Actor	System
Description	Based on the user's role, the system authenticates them and allows them access to the relevant dashboard.
Precondition	The user should be registered with valid credentials and a valid role must be assigned.
Post-condition	The user is logged in and a dashboard is displayed that matches their role.
Success Scenario	User enters valid username and password → system verifies credentials → role is checked → session starts → user is redirected to HR or Employee dashboard.
Failure Scenario	Invalid credentials or role → access denied → appropriate error message is displayed.

Table 2.6: UC-02 View and Edit Employee Profile

Field	Description
Use Case Identifier	UC-02
Primary Actor	Employee
Secondary Actor	System
Description	The employee views and updates their personal profile details such as name, contact info, etc.
Precondition	Employee must be successfully logged in.
Post-condition	Profile is updated and reflects the new information.
Success Scenario	Employee logs in → navigates to profile → edits information → submits → system updates record.
Failure Scenario	Invalid input or system error during submission → update fails → error message is shown.

Table 2.7: UC-03 Submit Leave Request

Field	Description

Use Case Identifier	UC-03
Primary Actor	Employee
Secondary Actor	System
Description	Employee fills out a leave form and submits a request for approval.
Precondition	Employee must be logged in and have access to the leave request page.
Post-condition	Leave request is saved in the system and awaits HR approval.
Success Scenario	Employee logs in → navigates to leave request form → fills and submits → request saved in DB.
Failure Scenario	Form incomplete or invalid data → system shows error → leave request not submitted.

Table 2.8: UC-04 View Analytical Dashboards

Field	Description
Use Case Identifier	UC-04
Primary Actor	HR
Secondary Actor	System
Description	HR views interactive dashboards showing employee analytics, attrition trends, and department stats.
Precondition	HR user must be logged in and have dashboard access rights.
Post-condition	Relevant charts and reports are displayed based on current employee data.
Success Scenario	HR logs in → navigates to dashboard → system retrieves and displays updated analytics charts.
Failure Scenario	Data retrieval failure or system error → charts fail to load → error message shown.

Table 2.9: UC-05 Review and Manage Job Applicants

Field	Description
Use Case Identifier	UC-05
Primary Actor	HR User
Secondary Actor	System
Description	The HR views job applications and decides whether to hire or reject each applicant. If hired, the system creates a user

	account and transfers applicant data to the employee database.
Precondition	The applicant has submitted a complete job application and the HR user is logged in.
Post-condition	The applicant's status is updated; if hired, their data is added to the employee records and a system account is generated.
Success Scenario	HR logs in → views list of applicants → selects an applicant → chooses "Hire" → system adds data to employee table and creates a user account.
Failure Scenario	HR attempts to hire without valid application data → error shown; or hiring action fails → status not updated.

Table 2.10: UC-06 HR Adds or Edits Employee Data

Field	Description
Use Case Identifier	UC-06
Primary Actor	HR
Secondary Actor	System
Description	HR staff adds new employee records or updates existing employee information in the system.
Precondition	HR user must be logged in with appropriate permissions.
Post-condition	Employee data is saved or updated successfully in the database.
Success Scenario	HR logs in → navigates to employee management → fills or edits form → submits → data saved.
Failure Scenario	Invalid input or database error → data not saved → error message displayed.

Table 2.11: UC-07 Approve or Reject Leave Requests

Field	Description
Use Case Identifier	UC-07
Primary Actor	HR
Secondary Actor	System, Employee
Description	HR reviews pending leave requests and approves or rejects them, triggering notifications to employees.
Precondition	HR must be logged in and have access to leave request management.

Post-condition	Leave request status is updated; if approved, employee receives email notification.
Success Scenario	HR logs in → reviews leave requests → approves or rejects → status updated → notification sent.
Failure Scenario	System error or invalid operation → status not updated → error message shown.

Table 2.12: UC-08 Predict Employee Churn

Field	Description
Use Case Identifier	UC-08
Primary Actor	HR
Secondary Actor	System
Description	The system predicts the likelihood of employee attrition using the trained machine learning model.
Precondition	HR must be logged in; employee data must be up-to-date and available for prediction.
Post-condition	Churn prediction results are generated and displayed on the dashboard.
Success Scenario	HR requests prediction → system processes data → displays prediction (Yes/No) for each employee.
Failure Scenario	Data missing or model error → prediction fails → error message displayed.

Table 2.13: UC-09 Submit Job Application

Field	Description
Use Case Identifier	UC-09
Primary Actor	Applicant
Secondary Actor	System
Description	The applicant fills out and submits a job application form through the system.
Precondition	The job application form is accessible, and the applicant provides required details.
Post-condition	A new applicant record is stored in the database.
Success Scenario	Applicant opens form → fills in details → submits form → system saves application.
Failure Scenario	Required fields are missing → system shows error → form not submitted.

Table 2.14: UC-10 Receive Hiring Notification Email

Field	Description
Use Case Identifier	UC-10
Primary Actor	System
Secondary Actor	Applicant
Description	The system sends an email to the applicant after the HR marks them as hired.
Precondition	HR must update the applicant status to "Hired."
Post-condition	The applicant receives a notification email with hiring details.
Success Scenario	HR hires applicant → system triggers email → applicant receives confirmation.
Failure Scenario	Email service is down or incorrect email provided → applicant does not receive notification.

2.1.1.4 Non-functional Requirements

Non-functional requirements defines the expected performance of the system, focusing on quality, security, and usability. Some non-functional requirements of Employee Analytics and Churn Prediction System include:

Table 2.15: Non-functional Requirements

Requirement ID	Category	Description
NFR-01	Performance	The system should load key pages (dashboard, profile) within 3 seconds under normal conditions.
NFR-02	Usability	The user interface must be simple, intuitive, and consistent for both HR and Employees.
NFR-03	Security	Passwords must be hashed using bcrypt; sessions and user roles must be securely managed.
NFR-04	Compatibility	The application should work across major web browsers.
NFR-05	Accuracy	The churn prediction model should maintain at least 90% accuracy on test data.

2.1.2 Feasibility Study

An evaluation covering Technical, Economic, Legal, Operational, and Schedule aspects was performed to determine the project's viability. This helped ensure the

system could be developed successfully while meeting all necessary requirements and constraints.

2.1.2.1 Technical Feasibility

The project is technically feasible if it uses well-supported technologies like Flask (Python), MySQL, HTML, Tailwind CSS, JavaScript, and scikit-learn for machine learning. These tools are open-source, have strong community support, and are suitable for building both the analytical and predictive features of the system. Having experience with these tools and all resources being freely available, technical implementation was achievable without external dependencies.

2.1.2.2 Economic Feasibility

The system is cost-effective since it primarily relies on free and open-source technologies. No paid tools or licenses are required for development or local testing. Since the project is not intended for deployment in this phase, server or hosting costs are not applicable. Long-term economic benefits include reduced HR workload, better retention planning, and improved decision-making through predictive analytics.

2.1.2.3 Legal Feasibility

The system uses publicly available datasets (IBM HR Dataset from Kaggle) that are cleared for academic and non-commercial use. It handles sensitive employee data with proper login authentication and hashed passwords, following good data handling practices. If deployed commercially, additional data protection policies may need to be considered.

2.1.2.4 Operational Feasibility

The system meets key operational needs of HR and employees, such as managing employee data, processing leave requests, and predicting churn. The interfaces are role-specific, user-friendly, and meet the day-to-day requirements of the intended users. The system has been tested to ensure functionality and ease of use.

2.1.2.5 Schedule Feasibility

The project was planned and developed using the Spiral methodology, with clearly defined tasks and iterative updates. The scope was well managed by focusing only on core features. The work was completed within the expected academic timeline, making it feasible in terms of scheduling.

2.1.3 Structured Modelling

The Employee Analytics and Churn Prediction System is designed using a structured modeling approach. This method focuses on breaking the whole system into clearly defined processes and flow of data, making it ideal for applications with role-based features and CRUD operations.

Structured modeling helps visualize how data moves through different modules such as login, employee management, leave handling, and churn prediction. It typically

involves tools like data flow diagrams (DFDs) and entity-relationship diagrams (ERDs) to represent how inputs are processed and stored.

By following a top-down design approach, the system remains organized, easy to develop, and straightforward to test. Given the predictable workflows and well-defined user roles, structured modeling is a suitable and effective choice for this project.

The following diagrams were central to this process:

2.1.3.1 Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) explains how data moves within the system, describing the interaction between users, system processes, and the database. In this project, the DFD includes key processes like user login, employee data management, leave requests, churn prediction, and the job application and hiring process.

External entities such as HR, Employees, and Applicants interact with processes that handle authentication, CRUD operations, machine learning predictions, and job applications. These processes exchange data with internal data stores like the users, employees, applicants, and leave requests tables in the database.

The DFD helps to clearly show how all modules are connected and how data flows through the system step-by-step, making the design easier to understand and manage.

Level 0 DFD (Context Diagram)

The Level 0 DFD, also known as the context diagram, provides a high-level overview of the entire system as a single process. It shows the system's interactions with external entities such as users or other systems, emphasizing what data enters and exits the system without detailing internal processes.

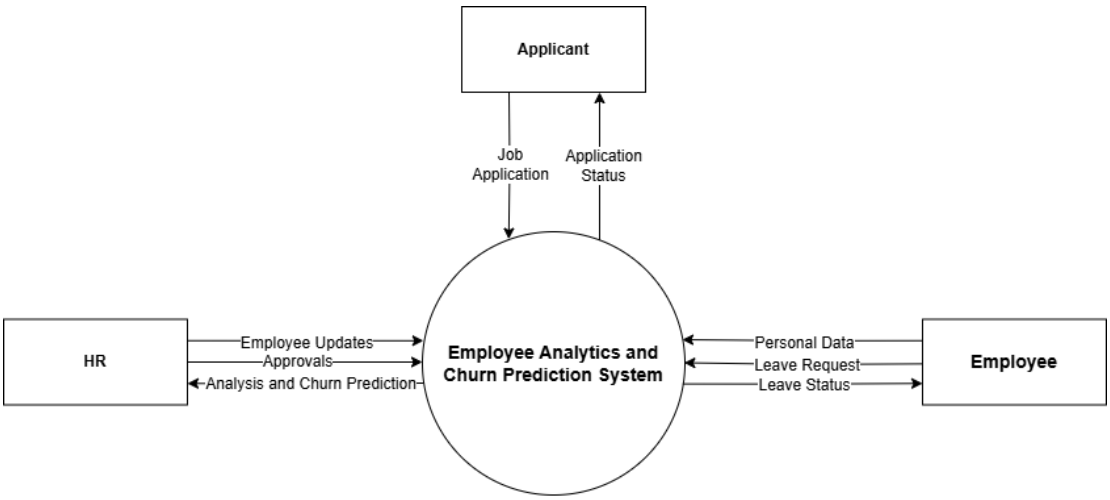

Figure 2.2: DFD Level 0 (Context Diagram)

Level 1 DFD

A Level 1 DFD breaks the one process from the Level 0 diagram down into several smaller processes. It shows the main functional areas of the system and how data moves between them, external entities, and data stores. This level gives more information about how the system works and why it does things the way it does.

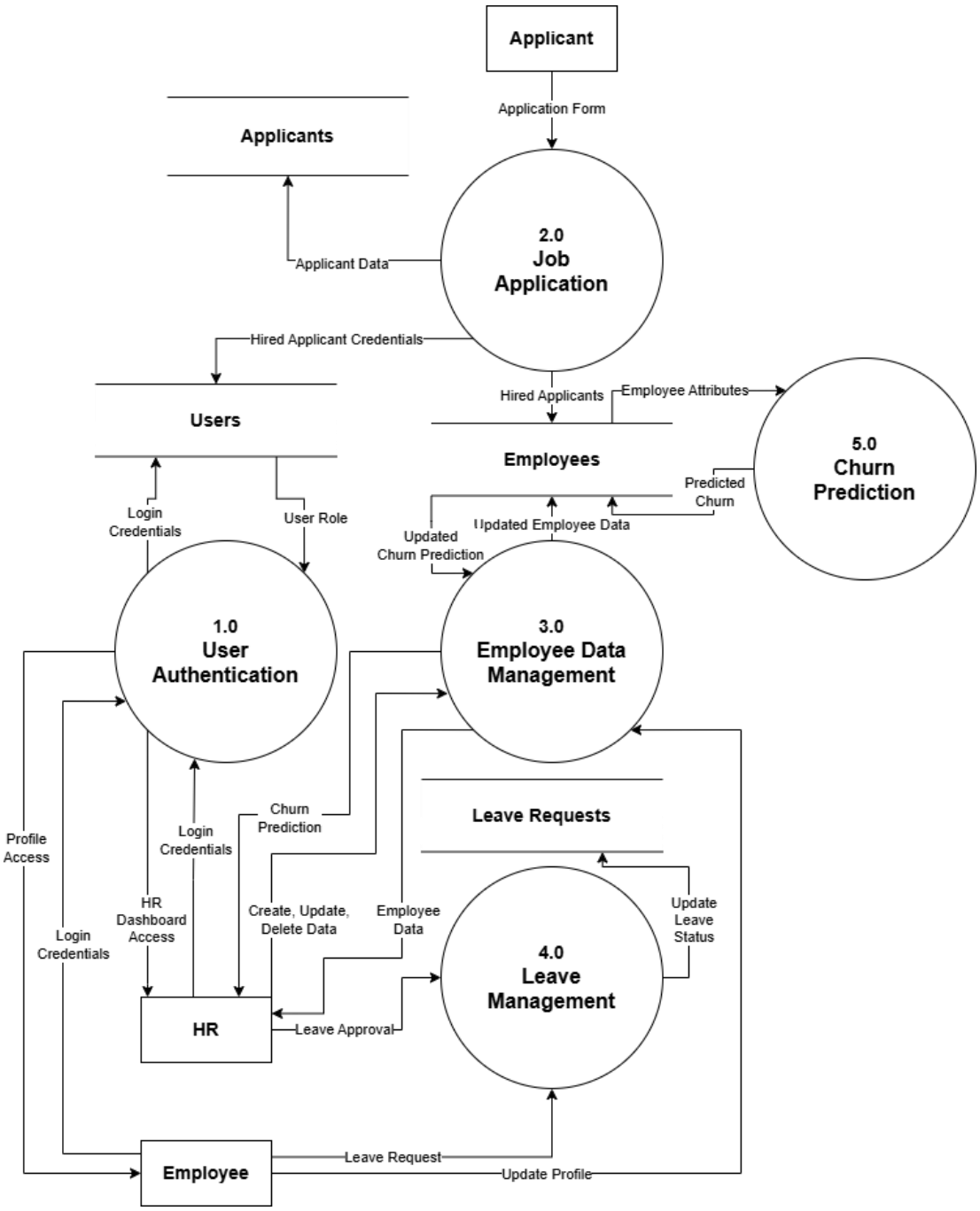

Figure 2.3: DFD Level 1

Level 2 DFD

The Level 2 DFD takes the Level 1 diagram and breaks down each process into smaller, more specific sub-processes. It goes into great detail about each functional component, showing exact data inputs, processing steps, and outputs. Developers and system analysts often use this level to figure out complicated workflows and come up with ways to make system parts work better.

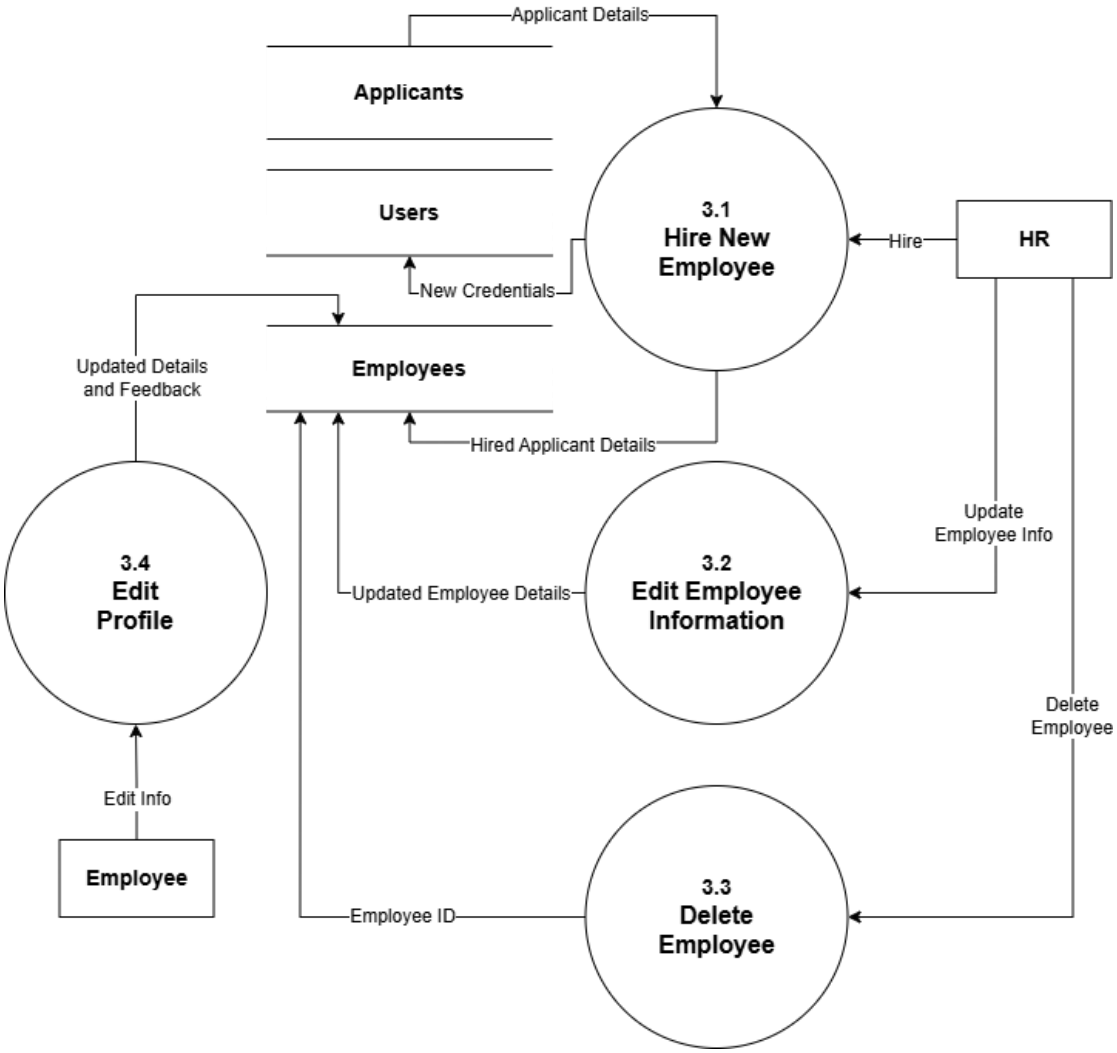

Figure 2.4: DFD Level 2

2.1.3.2 Entity-Relationship Diagram (ERD)

The Entity-Relationship Diagram (ERD) shows how the database is set up by showing the entities, their properties, and how they are related to each other. Users, Employees, and Leave Requests are the most important things in this system. There is an employee record for each user, and they are either an HR person or an employee. HR keeps track of and manages all of the leave requests that employees make. The ERD makes sure that the database is well-organized, supports role-based access, and keeps the data safe. The ERD gives you a clear base for making efficient queries and keeping the system consistent by showing how these relationships work.

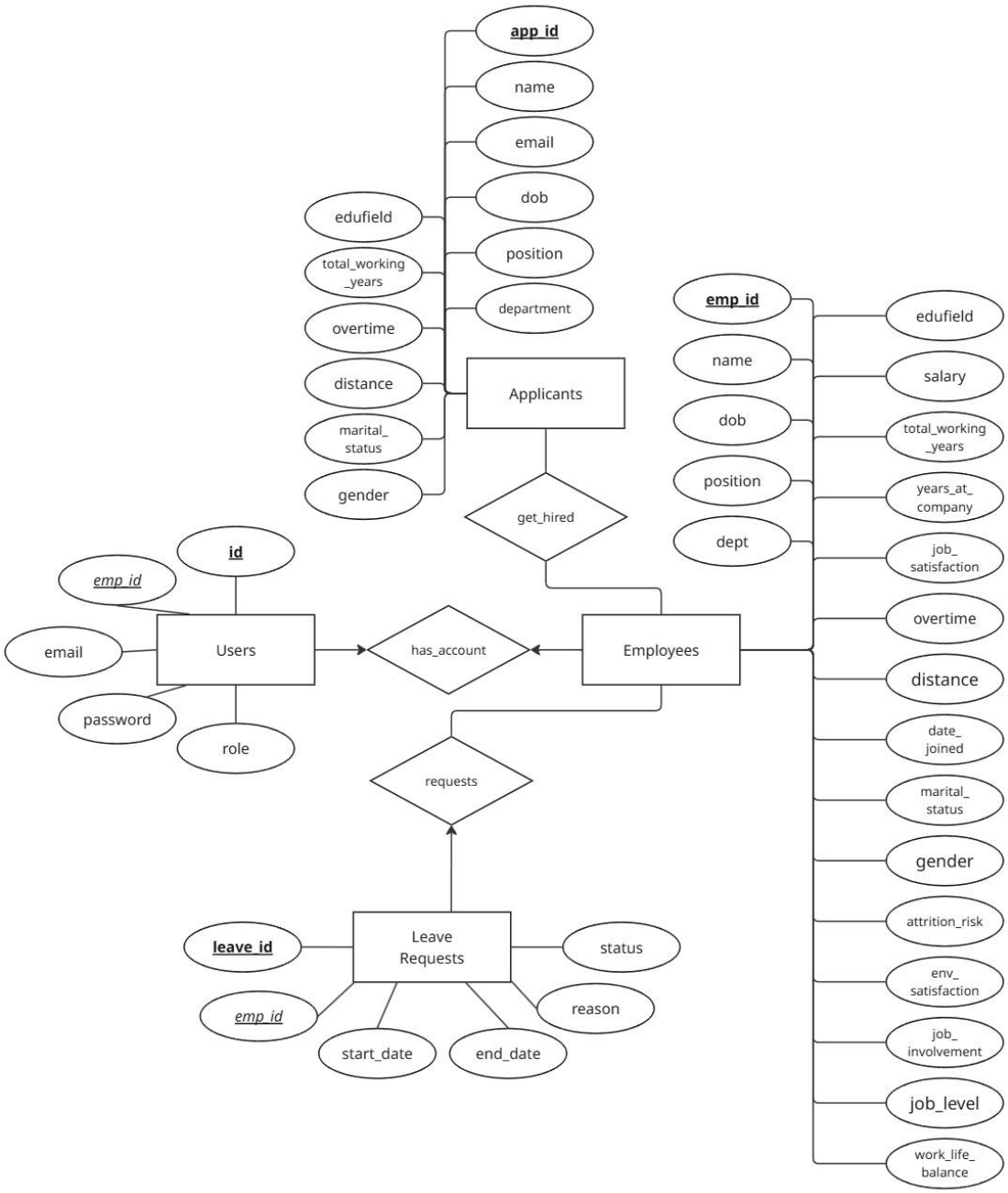

Figure 2.5: ER Diagram

2.1.4 System Requirements Specification (SRS)

The System Requirements Specification (SRS) tells you what hardware and software you need for the system to work.

Table 2.16: Hardware Requirements

Category	Requirements (Minimum / Recommended)	Purpose
Processor	Minimum: Dual-core 2.0 GHz / Recommended: Quad-core 2.5 GHz+	To run the application and ML models smoothly
RAM	Minimum: 4 GB / Recommended: 8 GB or more	To support multitasking and data processing
Storage	Minimum: 100 GB HDD/SSD / Recommended: 256 GB SSD+	To store project files, database, and software
Display	Minimum: 13-inch HD / Recommended: 14-inch Full HD+	To provide clear visualization of UI and dashboards

Table 2.17: Software Requirements

Category	Requirements (Minimum / Recommended)	Purpose
Operating System	Minimum: Windows 10 64-bit or Linux / Recommended: Latest versions	To provide a stable environment for development and testing
Programming Language and Framework	Python 3.8+ / Flask Framework latest stable	To develop backend, APIs, and machine learning components
Database	MySQL Server latest stable	To store and manage employee and leave data securely
Web Browser	Modern browsers like Chrome, Firefox, Edge latest versions	To access and test the web application interface
Development Tools	Text Editor (minimum) / IDE like VS Code or PyCharm (recommended)	To write, debug, and maintain code efficiently

2.2 Design

Planning the structure and user interface of the system is important for making a solution that is clear and well-organized. This means figuring out how parts will connect and how users will use the system to meet all of the requirements.

2.2.1 System Architecture

The Employee Analytics and Churn Prediction System is built on a three-tier architecture. This means that the system is divided into three logical layers: the Presentation Tier, the Application Tier, and the Data Tier. This design makes it easier to add new parts, grow, and keep things running.

1. **Presentation Tier (Frontend)**: The presentation layer is the part of the system that the user sees and interacts with through a web browser. It was made with HTML, Tailwind CSS, JavaScript, and Chart.js. HR and employees can use the system to log in, manage records, ask for time off, and see analytics dashboards, among other things. This tier takes care of all user input and shows visual feedback from backend responses.
2. **Application Tier (Backend/Logic Layer)**: The Flask framework takes care of the application's logic. It handles user requests, authenticates users, manages sessions, does CRUD operations on employee and leave data, and acts as a bridge between the front end and back end.
3. **Data Tier (Database)**: The data storage layer uses a MySQL database. It holds all persistent data including user credentials, employee records, leave applications, and churn prediction results. This tier ensures data integrity, supports queries for data retrieval, and serves as the foundation for analytics and model inputs.

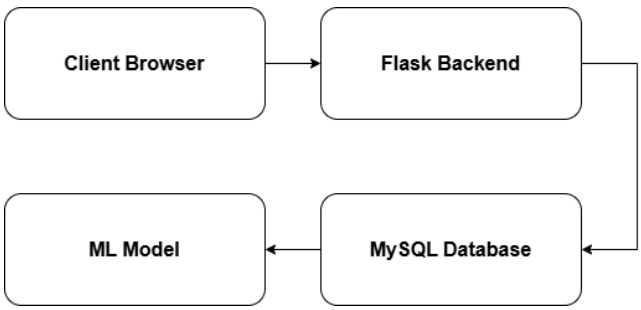

Figure 2.6: System Architecture Diagram

2.2.2 Database Design

The structure of a database schema shows how data is stored in a database, such as tables, fields, relationships, and rules. It serves as a guide for how the system stores, connects, and accesses data.

The Employee Analytics and Churn Prediction System has a database schema with four main tables: employees, users, leave requests, and applicants. The employees table has a lot of information about each employee, like their department, salary, experience, and other fields that are used for analytics and predicting churn. The users

table handles system login and access by storing credentials and giving roles (HR or employee). It is connected to each employee through a foreign key. The leave requests table keeps track of employees' requests for leave, including the type of leave, how long it will last, and whether or not it has been approved. The applicants table keeps track of the job applications that users send in when they apply for jobs. When an applicant is hired, their information is moved to the employees table and a user account is made.

These tables work together to keep the system running smoothly and the data consistent. They support important features like authentication, leave handling, analytics, and the hiring process.

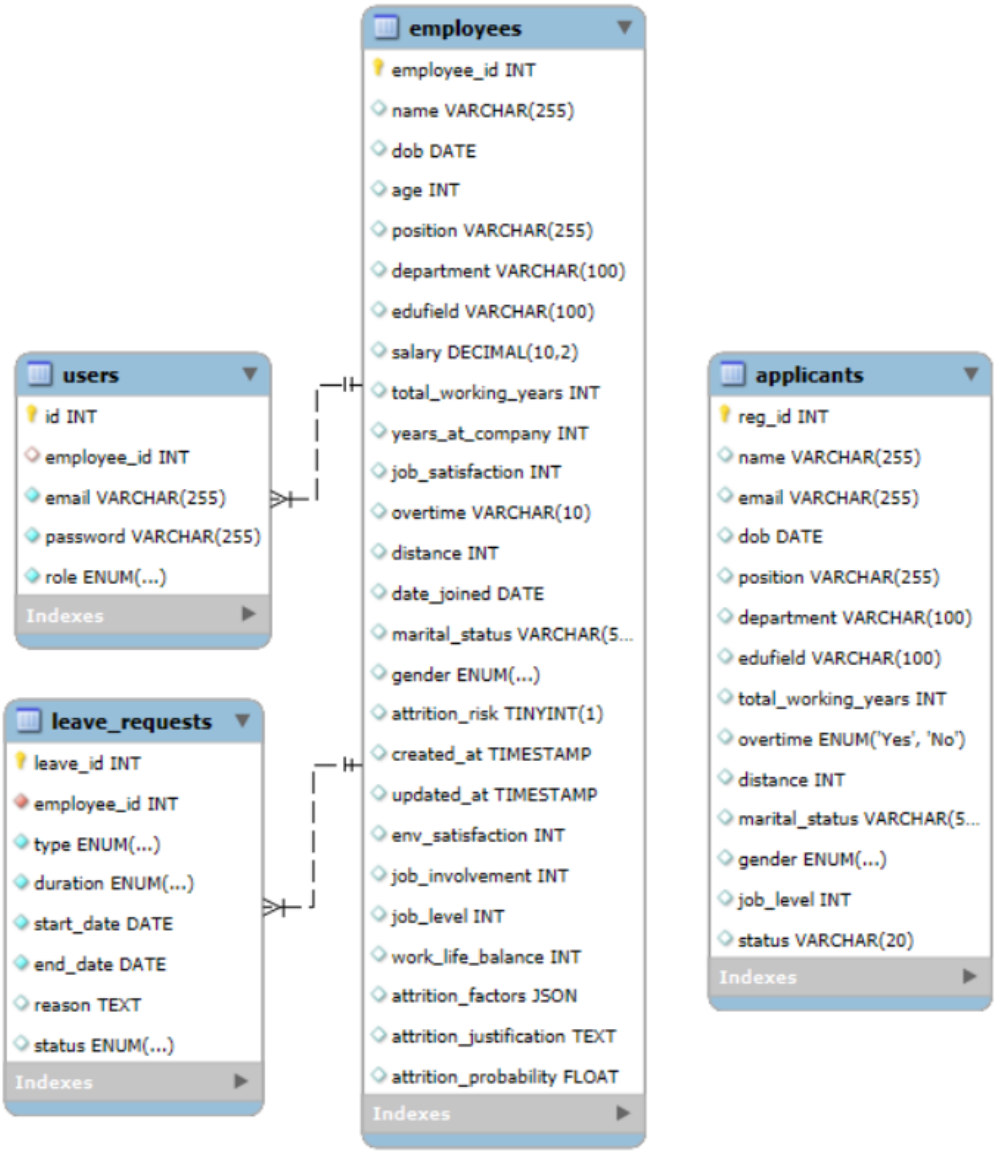

Figure 2.7: Database Schema Diagram

2.2.3 User Interface Design

A user interface (UI) diagram shows how the screens or pages in a system are set up and how users get from one to the other. It helps you see how the navigation works and where important things like forms, buttons, and dashboards are located.

The UI diagram for the Employee Analytics and Churn Prediction System shows three main roles: HR, Employee, and Applicant. The HR interface has analytics dashboards, pages for managing employees (add, edit, view, delete), sections for approving leave, churn prediction results, and views for managing job applications. Employees can see their own profile page, request time off, see their leave history, and give feedback. There is a special view where applicants can fill out and send in job application forms. This layout makes the system easier to use and keeps it simple. It also makes it easier for all users to find the features they need for their job.

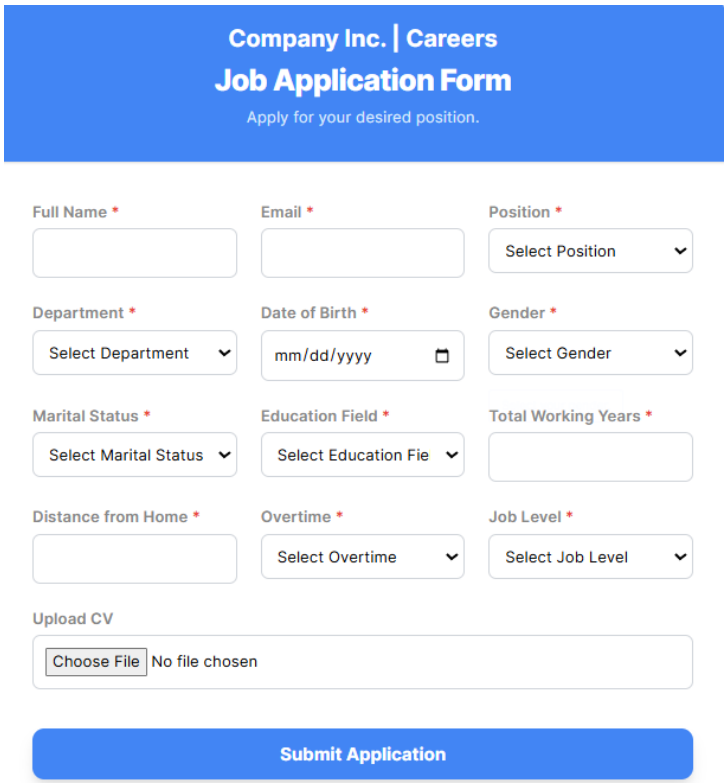

Figure 2.8: Job Application UI

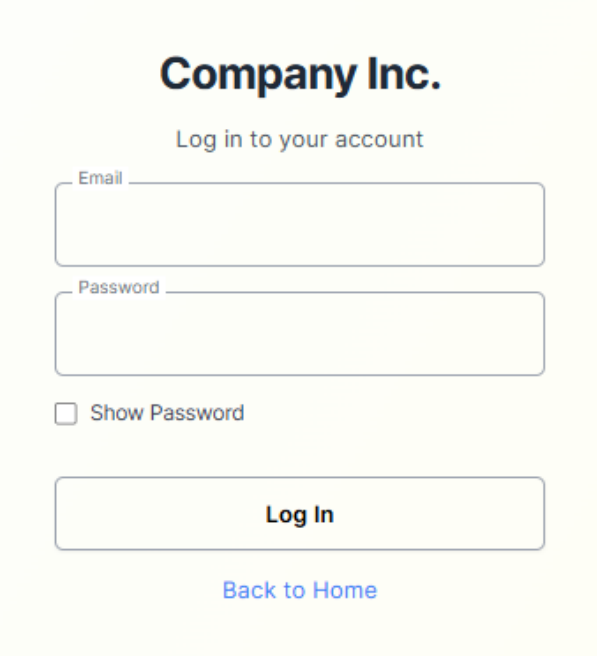

Figure 2.9: Login UI

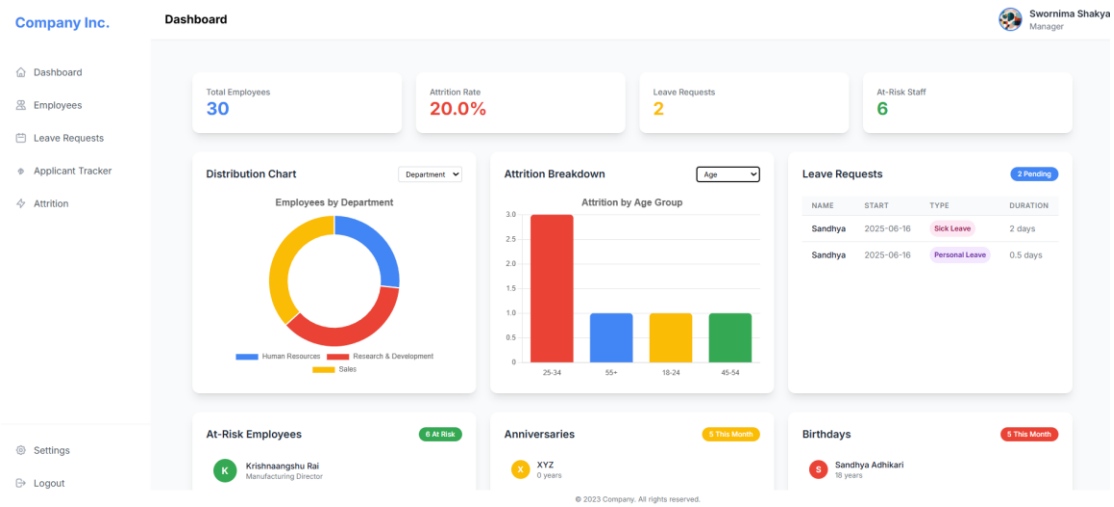

Figure 2.10: Dashboard View (HR)

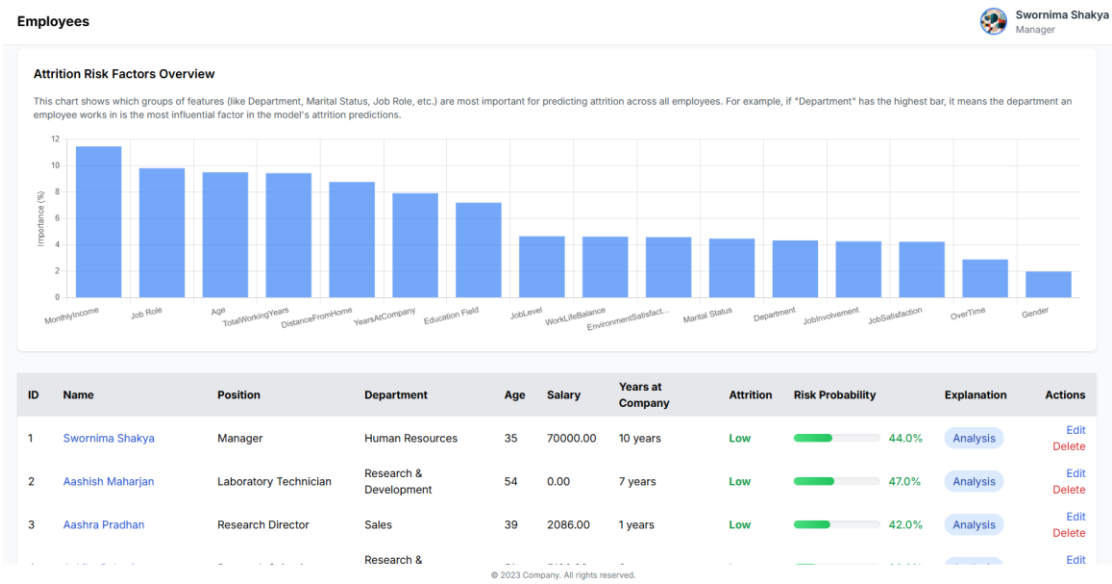

Figure 2.11: Employee Details and Churn Risk View (HR)

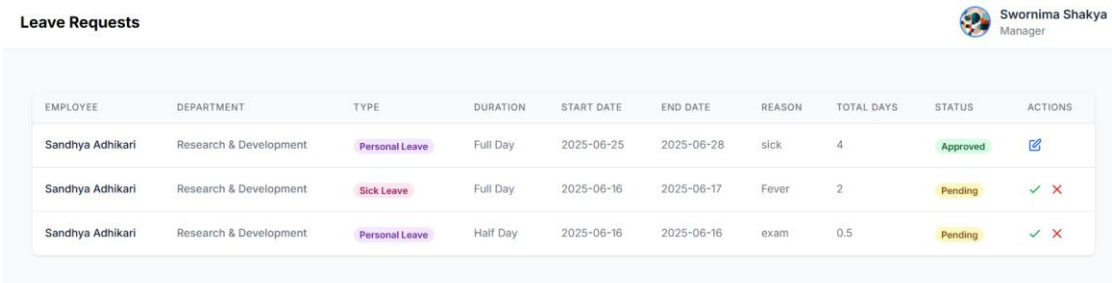

Figure 2.12: Leave Requests View (HR)

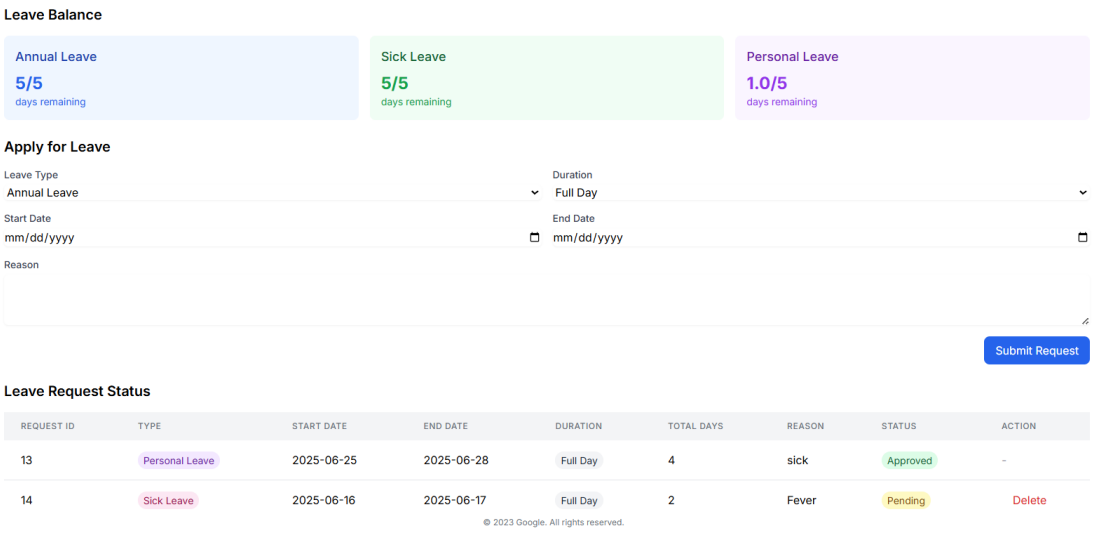

Figure 2.13: Leave Request View (Employee)

2.2.4 Algorithm Design

Algorithm design is the process of making a clear and logical set of steps for a system to follow in order to solve problems or finish tasks. It helps figure out how the system should handle inputs to get the right output.

In the Employee Analytics and Churn Prediction System, algorithm design is mainly used in the machine learning part. A Random Forest Classifier was used to predict employee attrition. The model was trained on the IBM HR dataset after cleaning the data by removing unnecessary fields, converting categories to numbers, and normalizing values. It uses factors like job satisfaction, years at the company, and salary to predict if an employee might leave. Additionally, simple logic is used in the hiring process to move applicant data to the employee table once hired.

This logical structure helps make accurate predictions and improves HR decision-making to lower churn and manage employee data more effectively.

2.2.4.1 User Login and Authentication Module

The User Login and Authentication module is responsible for securely verifying user identities before granting access to the system.

Pseudocode

```
FUNCTION login:
    IF form is submitted:
        Get email and password from form

        Connect to database
        Look for user with given email

        IF user not found:
            Show error and go back to login

        Check if stored password is hashed
        Compare entered password with stored one

        IF password is incorrect:
            Show error and go back to login

        Save user info in session

        Get employee name and position
        Save in session

        IF role is HR:
            Go to HR dashboard
        ELSE IF role is Employee:
            Go to Employee view
        ELSE:
            Show error for invalid role
            Go back to login
    ELSE:
        Show login form
```

2.2.4.2 Job Application Submission Module

The Job Application Submission module allows applicants to fill in and submit their job applications. The system stores the application details in the database for HR review.

Pseudocode

```
FUNCTION job_apply:
    IF form is submitted:
        Read all form fields like name, email, position, etc.

        Connect to the database

        TRY:
            Open database cursor
            Insert the form data into applicants table
            Save the changes
            Close the cursor

        FINALLY:
            Close the database connection

        Show message: "Application submitted successfully"
        Go back to application page

    ELSE:
        Show the job application form
```

2.2.4.3 Employee Data Management Module

This module allows HR users to view, add, edit, and delete employee records. It also supports automatic user account creation and churn prediction for each employee.

Pseudocode

```
FUNCTION show_employees:
    Check if user is logged in and is HR
    Get all employee data from database
    Show the list on the HR page

FUNCTION add_employee:
    IF form is submitted:
        Get employee details from the form
        Calculate how many years they've worked

        Add employee to the database
        Create login account for the employee
        Check if the employee is likely to leave

        Show success message and go back to list
    ELSE:
        Show the form to add a new employee

FUNCTION edit_employee(employee_id):
    IF form is submitted:
        Get updated details from form
        Recalculate years at company

        Update employee info in database
        Run prediction again

        Show success message and go back
    ELSE:
        Load employee data and show in form

FUNCTION delete_employee(employee_id):
    Remove user account for employee
```

```
        Delete employee from database
        Show success message and go back to list
```

2.2.4.4 Leave Management Module

The Leave Management module allows employees to submit leave requests, view their leave history, and enables HR to approve or reject leave applications while sending email notifications upon approval.

Pseudocode

```
    FUNCTION leave_requests:
        Get all leave requests for HR
        Show the leave requests page

    FUNCTION emp_leave:
        IF form is submitted:
            Read leave details from form
            Calculate total leave days
            Save new leave request with 'Pending' status
            Redirect to leave page

        ELSE:
            Get leave records for current employee
            Calculate total leave days for each
            Show the leave history page

    FUNCTION update_leave_status(request_id, status):
        IF user not logged in OR not HR:
            Redirect to login

        IF status is invalid:
            Redirect to leave requests page

        Update status of the leave request

        IF status is 'Approved':
            Get employee info
            Send approval email

        Redirect to leave requests page
```

2.2.4.5 Churn Prediction Module

The Churn Prediction module evaluates the likelihood of employee attrition using machine learning and updates the database, helping HR proactively identify at-risk employees.

Pseudocode

```
    FUNCTION preprocess_employee_data(raw_data):
        Convert data to DataFrame
        Standardize key names to lowercase
        Encode binary fields like gender and overtime
        One-hot encode categorical fields like department,
education, etc.
        Ensure all expected columns exist
        Reorder columns to match model input
        Apply saved scaler to normalize data
        RETURN processed data
```

```
FUNCTION predict_attrition_for_all:
    Get all employee records from database
    FOR each employee:
        Preprocess data
        Predict attrition risk with model
        Update prediction result in database
    Save and close database connection

ROUTE /predict-attrition:
    Check user is logged in and is HR
    Run predictions for all employees
    Get employees marked high-risk
    Show results on prediction page
```

2.2.5 System Flowcharts

A system flowchart is a visual diagram that shows the flow of data and processes within a system, illustrating how different components interact. It helps simplify complex systems by making it easier to understand, analyze, and communicate the workflow. System flowcharts are closely related to algorithm design because they both outline step-by-step procedures, with flowcharts providing a clear, visual representation of the algorithm's logic. This connection aids in planning and debugging system processes effectively.

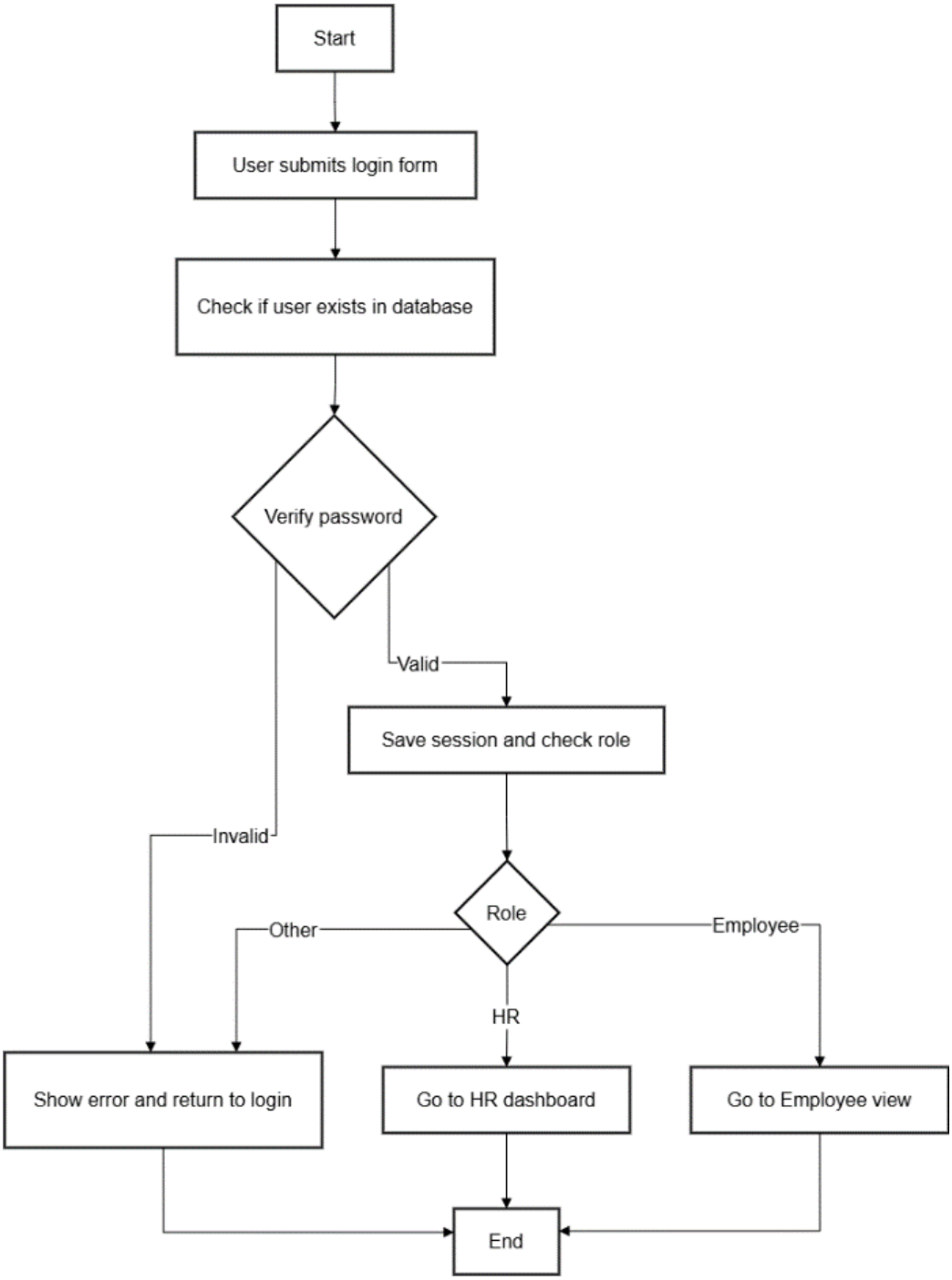

Figure 2.14: User Login and Authentication Algorithm Flowchart

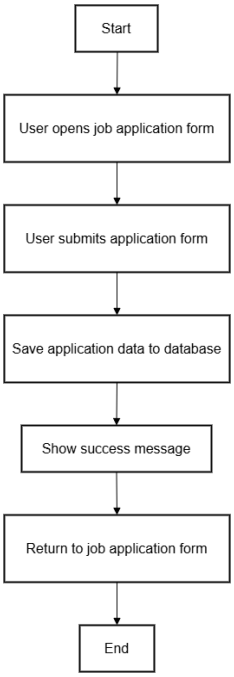

Figure 2.15: Job Application Module Flowchart

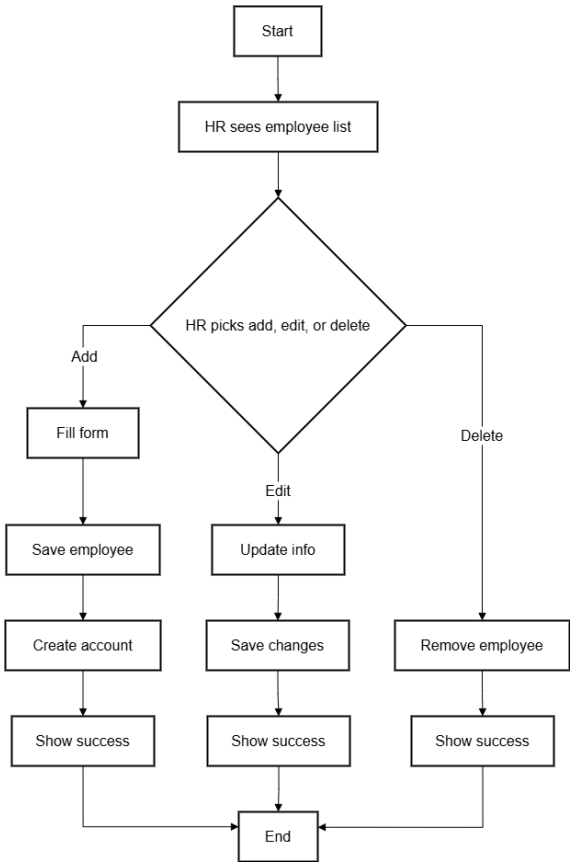

Figure 2.16: Employee Data Management Module Flowchart

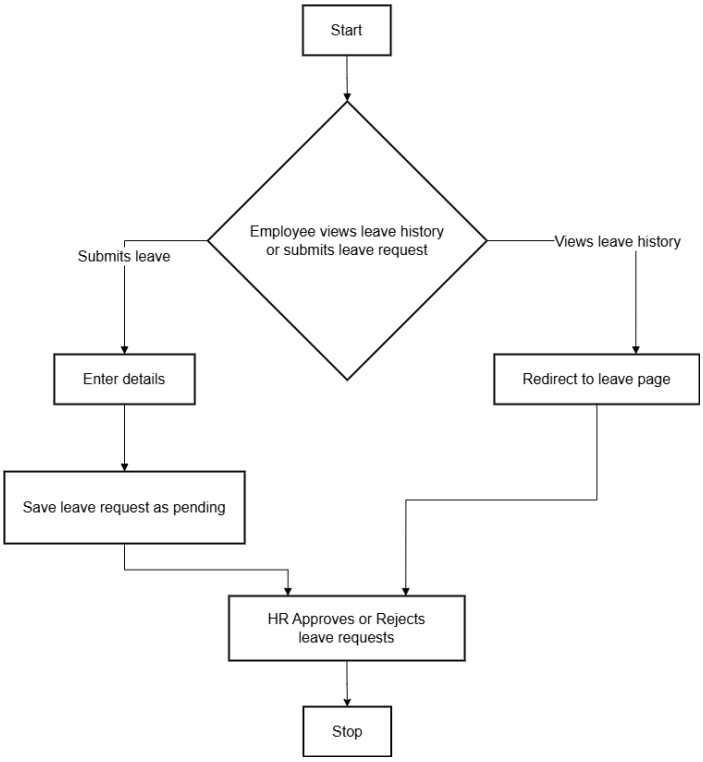

Figure 2.17: Leave Management Algorithm Flowchart

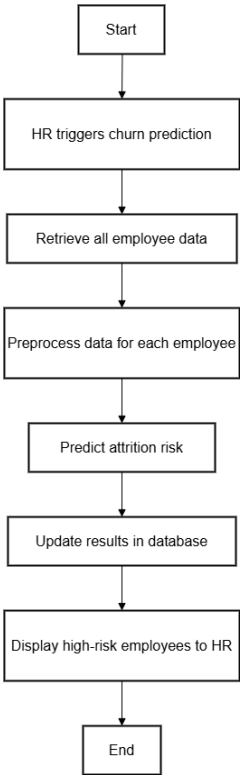

Figure 2.18: Churn Prediction Module Algorithm Flowchart

2.3 Implementation

Putting the planned designs into action by building the system's features and integrating all components is a key step. This process turns concepts into a working application that performs the intended tasks and functions.

2.3.1 Tools Used

The Employee Analytics and Churn Prediction System uses a modern set of tools to provide a smooth and efficient user experience. Below is a summary of the main technologies used:

2.3.1.1 Frontend Development

The frontend was made using HTML to build the main parts of the web pages, like forms and tables. Tailwind CSS helped create designs that work well on different devices and look good. JavaScript was used to add interactive parts and make the pages more dynamic. For showing data in charts, Chart.js was used because it makes easy-to-understand graphs.

2.3.1.2 Backend Development

The backend used Flask to handle website routes, user sessions, and APIs. The frontend and backend talk to each other using RESTful APIs to share data smoothly. For security, user login is protected with flask-login and passwords are safely stored using bcrypt hashing.

2.3.1.3 Machine Learning

We used scikit-learn, a Python tool for machine learning, to build the prediction model. We used the Random Forest Classifier model to guess which workers might quit their jobs.

2.3.1.4 Database

MySQL was used as a database to keep track of important information like employee information, user accounts, and leave requests. This keeps everything safe and easy to find.

2.3.1.5 Development Tools

The project used a number of important tools to make the workflow easier. Visual Studio Code (VS Code) was the main integrated development environment (IDE) for writing code, organizing project files, and fixing bugs. It also had useful extensions for Python and Flask. We used Postman to test API endpoints to make sure that the frontend and backend components could talk to each other correctly. Git was the version control tool that made it easy to keep track of changes and work together on development. The Command Line Interface (CLI) was also used a lot to run Flask servers, control virtual environments, and run Python scripts.

2.3.2 Module Description

Code modules are separate parts that put related functions, routes, or logic into neat sections. They help you divide a big application into smaller parts that are easier to understand, test, and keep up with. Using modules makes code easier to reuse and lets different team members work on different features without stepping on each other's toes. This structure makes it easier to scale and helps keep the development process clean.

2.3.2.1 User Login and Authentication Module

This module handles secure access control for both HR and employee users. Users log in through a form where their credentials are verified against stored data in the database. Passwords are hashed using bcrypt to ensure security. After authentication, session data is initialized, including user role and employee ID. Depending on their role, users are redirected to different dashboards. HR users access administrative tools, while employees access their personal dashboard. Invalid credentials or roles trigger appropriate error messages.

2.3.2.2 Job Application Submission Module

This module allows external users (job applicants) to submit applications for open roles. The form collects essential fields such as name, email, date of birth, position applied for, department, education field, work experience, marital status, and other demographics. On submission, the data is stored in the applicants table in the database for HR review. A success message confirms the submission, and applicants are redirected to the application page. This module ensures all applications are organized and ready for future processing or selection.

2.3.2.3 Employee Data Management Module

Designed for HR users, this module supports complete CRUD operations (Create, Read, Update, Delete) on employee records. HR can add new employees by entering details such as name, department, salary, position, and date of joining. When a new employee is added, a login account is automatically generated with a unique email and default password. The system also calculates years at company based on the joining date. Updates to existing employee records trigger a re-evaluation of attrition risk using the churn prediction model. Deleting an employee also removes their user account to maintain referential integrity.

2.3.2.4 Leave Management Module

This module handles leave requests and approvals. Employees can apply for leave through a form, providing start and end dates along with the reason. Leave requests are marked as 'Pending' by default and saved in the database. HR users can view all leave requests and change their status to 'Approved' or 'Rejected'. Upon approval, the system sends an automatic email notification to the employee. Employees can also view their own leave history, while HR maintains oversight of all applications for decision-making and record-keeping.

2.3.2.5 Churn Prediction Module

The Churn Prediction Module helps HR find employees who are likely to leave the company so that they can take action quickly. We do this by using machine learning models that were trained on the IBM HR Analytics dataset. The following classification algorithms were tested: Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Naive Bayes, and Random Forest.

We used four main performance metrics to judge the models:

1. **Accuracy**: The percentage of correct predictions out of the total number of predictions.
2. **Precision**: The percentage of predicted positives that turned out to be true positives.
3. **Recall**: The percentage of true positives that were correctly identified.
4. **F1-Score**: The harmonic mean of precision and recall; this is helpful when the classes are not equal.

Model Performance Comparison:

Table 2.18: Model Performance Comparison

Model	Class	Precision	Recall	F1-Score	Accuracy
Random Forest	0	0.95	0.92	0.94	**0.94**
	1	0.93	0.95	0.94	
Logistic Regression	0	0.62	0.66	0.64	**0.64**
	1	0.66	0.62	0.64	
K-Nearest Neighbors	0	0.73	0.69	0.71	**0.73**
	1	0.72	0.76	0.74	
Support Vector Machine	0	0.60	0.67	0.63	**0.63**
	1	0.65	0.59	0.62	
Naive Bayes	0	0.59	0.73	0.65	**0.62**
	1	0.67	0.52	0.59	

Note: Class 0 means no attrition (the employee stays), and class 1 means attrition (the employee is likely to leave).

Among all models tested, the Random Forest Classifier achieved the highest performance across all metrics, making it the most reliable model for predicting employee attrition. It balances accuracy, precision, and recall, which is essential for HR decisions where both false positives and false negatives carry consequences.

The Random Forest model considers various features such as job satisfaction, years at company, salary, department, and distance from home. When a new employee is added or an existing record is updated, the system automatically processes the data, makes a churn prediction, and updates the database with the risk status.

These insights are displayed on an interactive dashboard accessible to HR, helping them prioritize retention strategies for high-risk employees.

2.3.3 Coding Standards and Conventions

The project uses simple and clear coding practices to keep the code easy to read and manage. Comments are added in the code to explain what each part does, making it easier for others to understand or update later. The code is organised into smaller parts called modules, and variables and functions have names that make sense. Errors are handled correctly so that the program doesn't crash, and encryption keeps important information like passwords safe. These simple steps help keep the software clean, dependable, and easy to make better.

2.3.4 Security Considerations

Security is a big part of how the system is built to keep sensitive company and employee data safe. Passwords are stored safely with bcrypt hashing, which makes it hard for people who shouldn't be able to get to them to do so. Role-based access control makes sure that users can only see and use data and features that are relevant to their role. This lowers the chance of unauthorised access. User sessions are carefully watched over and end when users log out.

The system also checks and cleans all user inputs to protect against common attacks like SQL injection. Data sent between the user's browser and the server is protected so that it can't be changed or intercepted. We use safe ways to send automated emails, like leave approval notifications, to keep communication safe. These steps help keep the system and its data private, correct, and accessible.

2.3.5 Testing

Testing is the process of checking a software system or its parts for bugs and making sure it meets the requirements that have been set. There are different levels of testing:

1. **Unit Testing**: Unit testing was conducted to make sure that each module, like user authentication, leave submission, and churn prediction logic, was working correctly. To make sure they were accurate and reliable, each function was tested on its own with dummy data and mock sessions that showed what the expected input and output would be.
2. **Integration Testing**: Integration testing was carried out to make sure that the different parts of the system work together correctly. This included checking how the login module works with the database and session management, as well as how the leave request form processes and shows data from both the employee and leave tables.
3. **System Testing**: System testing was performed to make sure that the whole application worked as it should. We tested all the modules, such as authentication, role-based access, dashboards, and database updates, from start to finish to make sure the system meets the requirements.

4. **User Acceptance Testing (UAT)**: User acceptance testing involved testing the system from the end-user's perspective, simulating real-world HR and employee usage scenarios. Feedback was gathered on usability, role-based access, and workflow correctness to ensure the system aligns with user expectations and business needs.

The following test cases were designed and executed based on the testing approaches outlined above, ensuring each module functions correctly and meets the project's functional requirements.

2.3.5.1 Test Cases

Test cases are detailed steps to follow during testing, showing what to do, what to enter, and what result to expect to make sure everything works right.

Test Case ID	Test Scenario	Test Steps	Expected Output	Actual Output	Status
TC01	User Login with Valid Credentials	1. Open login page 2. Enter valid email and password 3. Click login	User is redirected to dashboard or employee view based on role	User redirected to dashboard	Pass
TC02	User Login with Invalid Password	1. Open login page 2. Enter valid email and invalid password 3. Click login	Error message "Invalid email or password" is displayed	Error message displayed	Pass
TC03	Job Application Submission	1. Open job application form 2. Enter all required details 3. Submit application	Application is submitted and confirmation message is shown	Confirmation message displayed	Pass
TC04	Edit Employee Details	1. Login as HR 2. Go to Employees 3. Edit an employee's details 4. Save changes	Employee details updated and attrition prediction re-calculated	Details updated, prediction re-calculated	Pass
TC05	Delete Employee	1. Login as HR 2. Go to	Employee and their user	Employee deleted	Pass

		Employees 3. Delete an employee	account are removed from the database		
TC06	Employee Applies for Leave	1. Login as Employee 2. Go to Leave Request 3. Fill leave form 4. Submit	Leave request is saved and appears in leave request list with status "Pending"	Leave request status "Pending"	Pass
TC07	HR Approves Leave Request	1. Login as HR 2. Go to Leave Requests 3. Approve a pending leave	Leave request status updated to "Approved", employee receives approval email	Status "Approved", email sent	Pass
TC08	Employee Updates Profile Details	1. Login as Employee 2. Go to Edit Profile 3. Update details 4. Save	Profile is updated, attrition prediction re-calculated, success message displayed	Profile updated, prediction re-calculated	Pass
TC09	Employee Submits Feedback (After 1 Year)	1. Login as Employee (>1 year at company) 2. Go to Feedback Portal 3. Submit feedback	Feedback saved, attrition prediction re-calculated, success message displayed	Feedback saved, prediction re-calculated	Pass
TC10	Predict Attrition for All Employees (HR)	1. Login as HR 2. Go to Predict Attrition 3. Trigger prediction	All employees with high attrition risk are listed on the page	High-risk employees listed	Pass

2.3.6 Deployment

This project was run and tested locally on a Lenovo IdeaPad3 laptop using Python and Flask as the main tools. The MySQL database was set up on the same machine to store all data. The application runs through Flask's built-in development server and is accessed via a web browser on the local computer.

No live or cloud deployment was performed; the system is currently for local use only. For future use, deployment on a web server or cloud platform is recommended to allow access from multiple devices and improve scalability.

CHAPTER III: CONCLUSION AND RECOMMENDATION

3.1 Summary

This project created a web-based system to help manage employee data and leave requests while providing insights through data analysis. It includes a machine learning model to predict which employees may leave, supporting better HR decisions. Overall, the system aims to improve workforce management and reduce employee turnover.

3.1.1 Findings

The Employee Analytics and Churn Prediction System proved to be an effective and practical solution for streamlining HR operations while integrating machine learning for data-driven decision-making. Among the machine learning models tested—Logistic Regression, K-Nearest Neighbors, Support Vector Machine, and Naive Bayes—the Random Forest model delivered the best results. It achieved a high accuracy of 94%, with well-balanced precision and recall, making it the most reliable option for predicting employee attrition. The system analyzes key factors such as job satisfaction, department, years at the company, and salary to determine attrition risk.

The confusion matrix below summarize the performance of the Random Forest model in distinguishing between employees likely to stay (class 0) and those at risk of leaving (class 1):

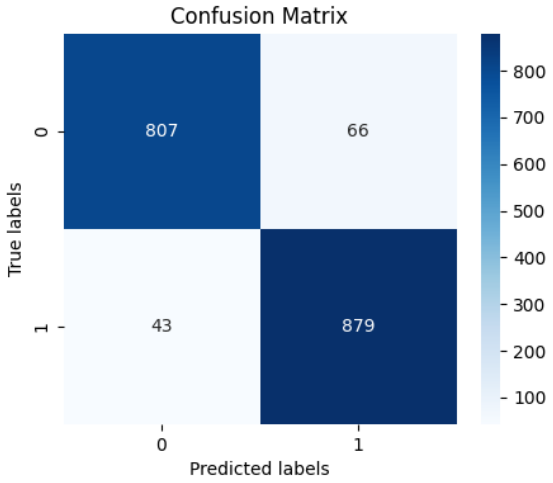

Figure 3.1: Confusion Matrix of Random Forest Model

Beyond prediction, the system includes secure login with role-based access, a job application module, employee record management, and a leave management system with automated email alerts. Visual dashboards offer insights into employee trends, helping HR make informed decisions. This project shows that even with a lightweight tech stack like Flask and MySQL, it's possible to create a functional, scalable HR tool that boosts efficiency and supports strategic planning.

3.1.2 Limitations

The system meets its main goals, but there were some problems found during its development and testing. The most important limits are:

1. The system is only hosted and tested in a local environment right now, which makes it harder to access and scale without being properly deployed on a cloud or production server.
2. The machine learning model is accurate, but it was trained on a small dataset, so it might not work well in bigger or more diverse organizations.
3. This version of the system doesn't have important HR features like tracking attendance, processing payroll, evaluating performance, and helping with hiring.
4. Email notifications use a simple SMTP setup, which might not work well or be reliable for handling more complicated or larger amounts of communication.
5. The system doesn't work with mobile devices, which can make it harder for people who like to do HR tasks on their phones or tablets to use it.

These problems show where the system can be better, and they can be fixed in future versions.

3.1.3 Future Enhancements

There are a number of suggested improvements that could be made in the future to make the system work better and be more useful for organizations in the real world. The goal of these upgrades is to get around the system's current problems and make it more useful and efficient:

1. Use a bigger and more varied dataset to retrain the churn prediction model to make sure it works better and can be used in a wider range of industries and job roles.
2. Add the ability to track attendance by connecting to biometric or IoT-based devices that are common in today's workplaces.
3. Deploy the system on cloud platforms like Heroku or AWS so that people can access it from anywhere and it can grow with bigger businesses.
4. Make a version of the system that works on mobile devices, especially for employees, so that tasks like checking profiles and applying for leave are easier.
5. Add payroll processing, attendance tracking, advanced performance evaluation tools, and an advanced applicant tracking system (ATS) to the system.

   These improvements will make the system a more complete HR solution that can meet the needs of a wide range of organizations.

3.1.4 Lessons Learned

The Employee Analytics and Churn Prediction System taught us a lot about both the technical and practical sides of making software. One important lesson was that HR staff and employees have different needs, so the system needed to be designed so that each The project also showed how important it is to have data that is clean and well-prepared when you make machine learning models. Getting very accurate predictions depended a lot on choosing the right features and preprocessing them. When you put

together different parts like databases, APIs, and machine learning modules, it showed how important it is to write code that is easy to maintain and modular.

It was very helpful to have worked with Flask before, especially for learning how to use basic features like routing, session management, and API development. These are all important for making web apps that are both secure and scalable.

The project showed how important it is to find a balance between how well something works and how easy it is to use. To make a system that was useful and worked well, we had to design simple, easy-to-use interfaces and make sure that the back-end operations were reliable. Also, clear documentation, following coding standards, and careful planning were all important to making sure that development went smoothly and that the project could grow in the future.

3.2 Conclusion

The Employee Analytics and Churn Prediction System shows that combining data analytics with HR management can improve the way a company runs and how it makes decisions. The system helps HR teams deal with workforce problems before they happen by giving them tools to manage employee data, handle leave requests, and predict when employees will leave. With the help of machine learning to predict churn, easy-to-use dashboards, and role-based access, both HR staff and employees can have a smooth and safe experience.

The project has some problems, like not being able to manage payroll or track attendance, but it lays a strong foundation for future improvements. In general, the system shows how organizations of all sizes can use modern technology to boost productivity, cut costs, and promote data-driven HR practices.

This project adds to what we know about computer science by using machine learning techniques in a real-world web application framework. It shows how to effectively combine backend development, database management, and data science. It shows how organized, role-based data access and analytics can help improve business processes and decision-making in human resource management. It also gives practical examples of how data-driven systems can help with operational efficiency.

3.3 Recommendation

The Employee Analytics and Churn Prediction System could make HR processes much better by giving them more information. The following short suggestions are made to make it more useful and valuable:

1. To make churn predictions more accurate, use a bigger and more varied dataset.
2. To make the HR tool more useful, add tracking for attendance and payroll.
3. Deploy on cloud platforms to make it easier to scale and access from anywhere.
4. Make the interface work well on mobile devices so that employees can get to it more easily.

5. Keep your documentation and code clean and clear so that you can make updates in the future.
6. Add advanced analytics and customizable reports to get more information.

Making these changes will make the system stronger, more flexible, and better suited to the needs of real organizations. This will help HR teams work smarter and keep employees interested.