RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI
ENGINEERING COLLEGE

CB23332
SOFTWARE ENGINEERING LAB

Laboratory Record Note Book

Name : . . . .

Year / Branch / Section : . . . .

Register No. : . . . .

Semester : . . . .

Academic Year : . . . .

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)
RAJALAKSHMI NAGAR, THANDALAM – 602-105

BONAFIDE CERTIFICATE

NAME:______________________**REGISTER NO.:** _______________

ACADEMIC YEAR: 2024-25 **SEMESTER:** III **BRANCH:**__________B.E/B.Tech

This Certification is the bonafide record of work done by the above student in the

CB23332-SOFTWARE ENGINEERING - Laboratory during the year 2024 – 2025.

Signature of Faculty -in – Charge

Submitted for the Practical Examination held on ____________

Internal Examiner

External Examiner

INDEX

EX NO:1	**WRITE THE COMPLETE PROBLEM STATEMENT**
DATE:	

AIM:

To prepare PROBLEM STATEMENT for any project.

ALGORITHM:

1. Authentication and Authorization:
2. Employee Data Management:
3. Attendance and Leave Management:
4. Payroll Processing:
5. Performance Evaluation:
6. Data Security Measures:

INPUT:

1. The input to requirement engineering is the problem statement prepared by customer.
2. It may give an overview of the existing system along with broad expectations from the new system.
3. The first phase of requirements engineering begins with requirements elicitation i.e. gathering of information about requirements.
4. Here, requirements are identified with the help of customer and existing system processes.

Problem:

The absence of an efficient, integrated system for managing employee information creates significant inefficiencies, increases administrative workload, and risks data privacy breaches. An automated Employee Management System is essential for improving HR operations, safeguarding data privacy, and ensuring compliance.

Background:

The absence of an efficient, integrated system for managing employee information creates significant inefficiencies, increases administrative workload, and risks data privacy breaches. An automated Employee Management System is essential for improving HR operations, safeguarding data privacy, and ensuring compliance

Relevance:

An Employee Management System is critical for organizations to centralize employee information, automate administrative functions, and enable data-driven decision-making. As organizations grow, a centralized system becomes necessary to avoid data fragmentation, ensure accuracy, and facilitate timely access to information.

1. **Objectives:**

   Centralize Employee Data to enable easy access and retrieval for HR functions.

2. **Enhance Data Privacy and Security** by implementing encryption, role-based access, and compliance mechanisms.
3. **Automate Attendance and Payroll** processing to minimize manual work and reduce errors.
4. **Improve Employee Experience** by providing a user-friendly interface and transparency in HR processes.
5. **Streamline Performance Evaluation** processes to assist in data-driven decision-making.

Result:

EX NO:2	**WRITE THE SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT**
DATE:	

AIM:

To do requirement analysis and develop Software Requirement Specification Sheet(SRS) for any Project.

ALGORITHM:

1. **User Authentication and Authorization**
2. **Employee Data Management**
3. **Attendance Management**
4. **Payroll Processing**
5. **Performance Evaluation**
6. **Report Generation**
7. **Notifications and Alerts**
8. **Logout**

1. Introduction

1.1 Purpose

The Employee Management System (EMS) is designed to automate HR processes, including managing employee data, tracking attendance, processing payroll, and conducting performance evaluations. The system aims to improve operational efficiency and provide accurate data management.

1.2 Document Conventions

This document uses standard conventions, including the use of specific terminologies like EMS (Employee Management System), HR (Human Resources), and RBAC (Role-Based Access Control).

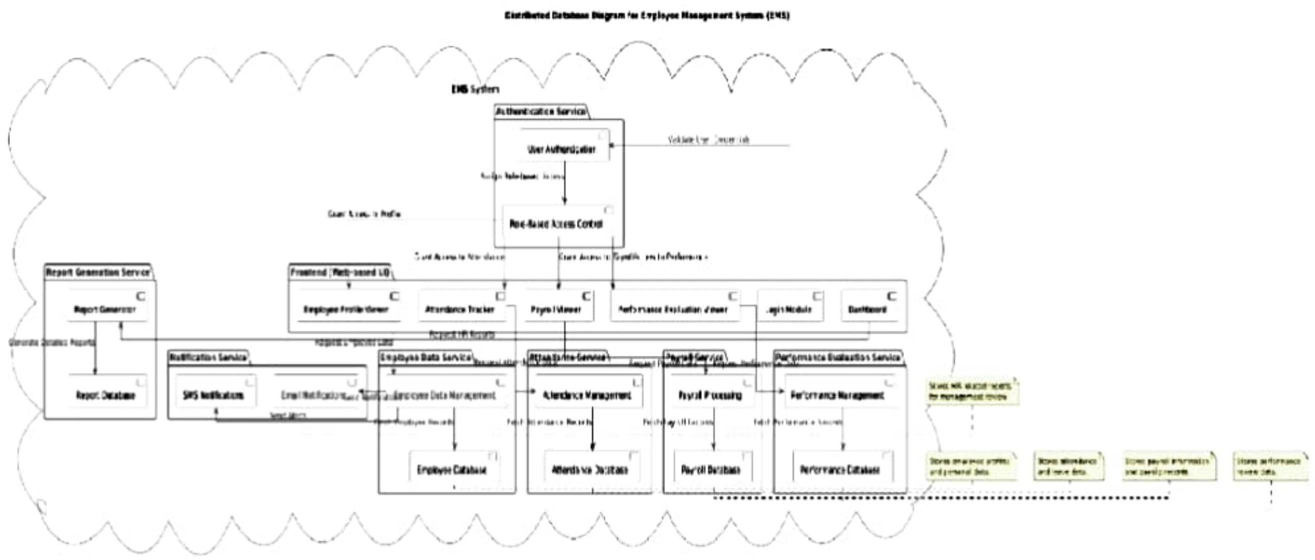
1.3. Intended Audience and Reading Suggestions

The primary audience for this document includes HR personnel, system developers, IT administrators, and management. It is recommended that HR personnel focus on the system features, while developers should review the detailed requirements and constraints.

1.4. Project Scope

The EMS will automate core HR functions such as employee data management, attendance tracking, payroll processing, and performance evaluations. The scope includes a web-based interface accessible by HR personnel, managers, and employees. The system excludes advanced features like AI-based predictive analytics and full integration with external HR systems.

Distributed database diagram:



1.5. References

- Company's HR Policy Manual
- ISO/IEC 27001 for Information Security Management
- GDPR Guidelines for Data Protection

2. Overall Description

2.1 Product Perspective

The EMS is a standalone web application that interfaces with existing payroll and attendance systems. It is designed to centralize employee data, streamline HR processes, and improve data accuracy and accessibility.

2.2 Product Features

Employee Data Management: Manage employee profiles, job roles, and personal information.

Attendance Management: Track daily attendance, manage leave requests, and generate attendance reports.

Payroll Processing: Automate salary calculations, deductions, and payslip generation.

Performance Evaluation: Facilitate performance reviews and store evaluation records.

2.3 User Class and Characteristics

HR Personnel: Full access to manage employee data, process payroll, and generate reports.

Managers: Access to performance evaluation tools, attendance tracking, and leave approvals.

Employees: Limited access to view personal data, attendance records, and payslips.

2.4 Operating Environment

The EMS will operate on a cloud-based server accessible via modern web browsers (e.g., Chrome, Firefox). The client-side will require minimal resources, and the system will support Windows, macOS, and Linux platforms.

2.5 Design and Implementation Constraints

The system must comply with data protection regulations (e.g., GDPR) and ensure scalability to accommodate growing employee numbers. Integration with existing systems must be seamless, and the system should be adaptable to future upgrades.

2.6. Assumptions and Dependencies

It is assumed that users are familiar with basic web applications and that the organization's IT infrastructure supports cloud-based applications. The EMS will depend on reliable internet connectivity for optimal performance.

3. Specific Requirements

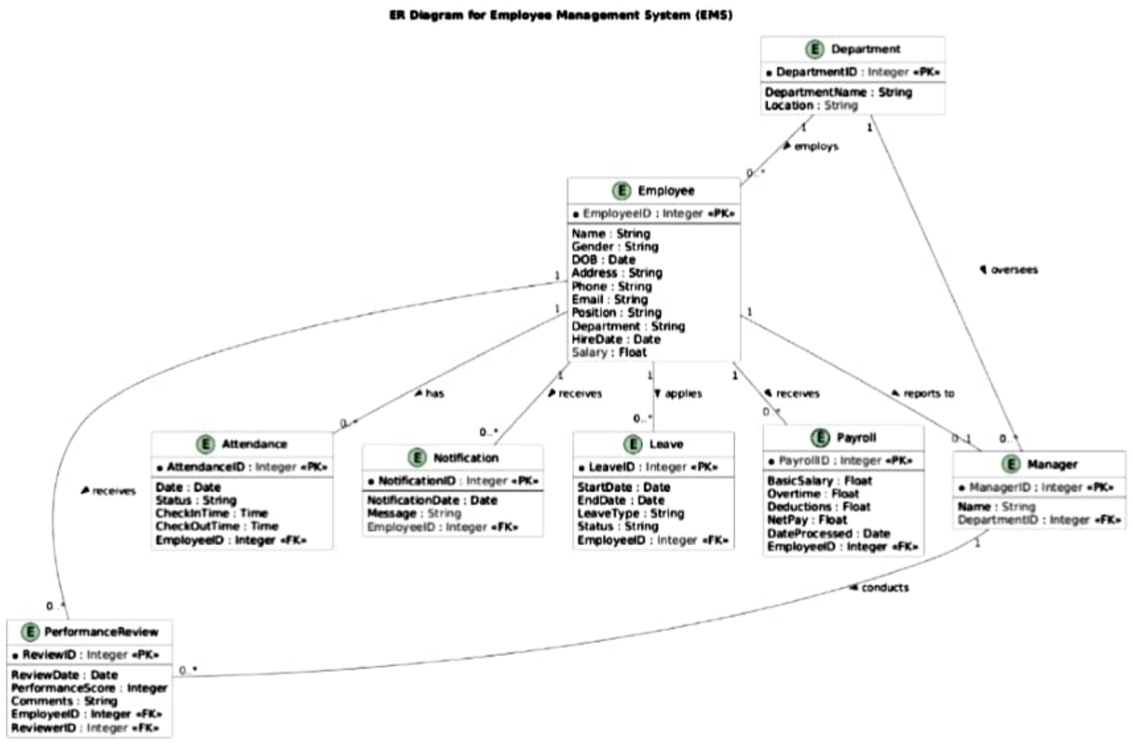
Description and Priority

The system prioritizes efficient employee management and enhanced user convenience, reducing administrative overhead and improving overall organizational efficiency.

Stimulus/Response Sequence

- **Employee logs in to the system:** Attendance is recorded.
- **Manager assigns tasks:** Task details are communicated to the employee, and updates are tracked.
- **Employee submits a request (e.g., leave, payroll):** Request is processed and status is updated.

ER Diagram:


ER Diagram for Employee Management System (EMS)
E Department
• DepartmentID : Integer «PK»
DepartmentName : String
Location : String
E Employee
• EmployeeID : Integer «PK»
Name : String
Gender : String
DOB : Date
Address : String
Phone : String
Email : String
Position : String
Department : String
HireDate : Date
Salary : Float
E Attendance
• AttendanceID : Integer «PK»
Date : Date
Status : String
CheckInTime : Time
CheckOutTime : Time
EmployeeID : Integer «FK»
E Notification
• NotificationID : Integer «PK»
NotificationDate : Date
Message : String
EmployeeID : Integer «FK»
E Leave
• LeaveID : Integer «PK»
StartDate : Date
EndDate : Date
LeaveType : String
Status : String
EmployeeID : Integer «FK»
E Payroll
• PayrollID : Integer «PK»
BasicSalary : Float
Overtime : Float
Deductions : Float
NetPay : Float
DateProcessed : Date
EmployeeID : Integer «FK»
E Manager
• ManagerID : Integer «PK»
Name : String
DepartmentID : Integer «FK»
E PerformanceReview
• ReviewID : Integer «PK»
ReviewDate : Date
PerformanceScore : Integer
Comments : String
EmployeeID : Integer «FK»
ReviewerID : Integer «FK»
employs
oversees
has
receives
applies
receives
reports to
receives
conducts
1
0..*
1
0..*
1
1
1
1
1
1
1
0..*
0..*
0..*
0..*
0..1
0..*
0..*
1
0..*

Functional Requirements

- **Real-Time Employee Tracking:**
- **Automated Attendance Management:**
- **Task Assignment and Progress Monitoring:**
- **Payroll Management:**
- **Leave and Request Management:**
- **Data Analytics and Reporting:**

4. External Interface Requirements

4.1 User Interfaces

The system shall have a web-based UI with login screens, dashboards, and forms for data entry and report generation. The UI shall be user-friendly and support responsive design for accessibility on different devices.

4.2 Hardware Interfaces

The EMS shall run on standard desktop or laptop computers with internet access. It may also support mobile devices for accessing limited functionalities.

4.3 Software Interfaces

The EMS shall interface with existing payroll and attendance software through APIs. It shall also integrate with the organization's email system for notifications.

4.4 Communication Interfaces

The system shall use HTTPS for secure communication between the client and server. Notifications and alerts shall be communicated via email or SMS.

5. Additional Requirements

5.1 Performance Requirements

The EMS shall support up to 1000 concurrent users with a response time of 2 seconds or less. The system shall be able to process payroll for up to 10,000 employees within 30 minutes.

5.2 Safety Requirements

The system shall include data backup and recovery mechanisms to ensure data safety. It shall also support disaster recovery with a maximum downtime of 2 hours.

5.3 Security Requirements

The EMS shall implement role-based access control (RBAC) to ensure that only authorized users can access specific data. Sensitive data, such as personal and payroll information, shall be encrypted both in transit and at rest. The system shall comply with GDPR and other relevant data protection regulations.

5.4 Software Quality Attributes

□ **Reliability:**

- The system should function without failures to ensure continuous employee management.
- Automatic data backup to prevent loss of critical employee information.

□ **Usability:**

- Intuitive interface for both employees and administrators, reducing the need for extensive training.
- Accessibility features, such as multi-language support and compatibility with assistive technologies.

□ **Scalability:**

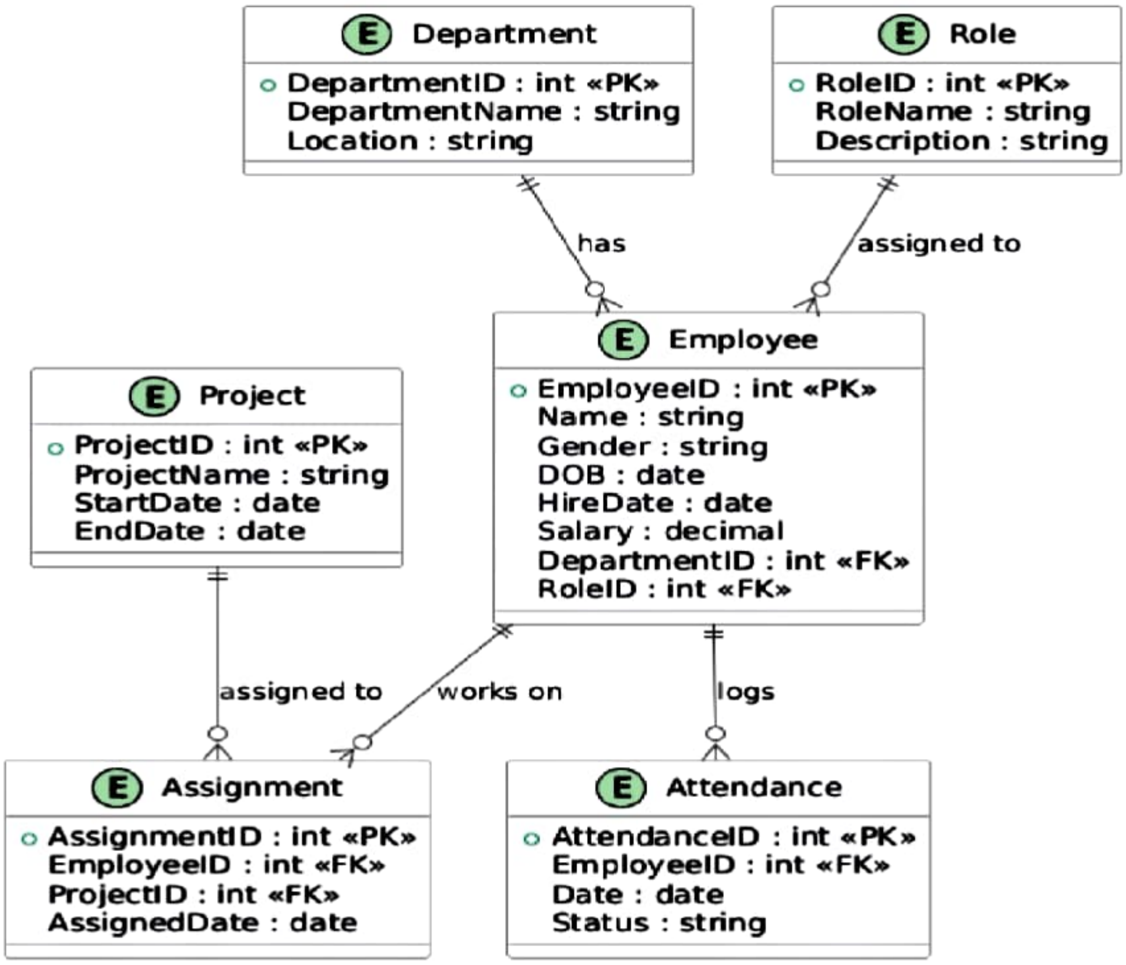
- Capable of handling an increasing number of employees and tasks as the organization grows.
- Efficient performance even with high traffic and simultaneous access by multiple users.

□ **Performance Efficiency:**

- Quick response times for task assignments, attendance recording, and payroll processing.
- Optimization for large data sets to avoid latency issues.

Result:

ER Diagram:



E Department
DepartmentID : int «PK»
DepartmentName : string
Location : string
E Role
RoleID : int «PK»
RoleName : string
Description : string
has
assigned to
E Employee
EmployeeID : int «PK»
Name : string
Gender : string
DOB : date
HireDate : date
Salary : decimal
DepartmentID : int «FK»
RoleID : int «FK»
E Project
ProjectID : int «PK»
ProjectName : string
StartDate : date
EndDate : date
assigned to
works on
logs
E Assignment
AssignmentID : int «PK»
EmployeeID : int «FK»
ProjectID : int «FK»
AssignedDate : date
E Attendance
AttendanceID : int «PK»
EmployeeID : int «FK»
Date : date
Status : string

EX NO:3	**DRAW THE ENTITY RELATIONSHIP DIAGRAM**
DATE:	

AIM:

To Draw the Entity Relationship Diagram for Employee Management System project.

ALGORITHM:

Step 1: Mapping of Regular Entity Types

Step 2: Mapping of Weak Entity Types

Step 3: Mapping of Binary 1:1 Relation Types

Step 4: Mapping of Binary 1:N Relationship Types.

Step 5: Mapping of Binary M:N Relationship Types.

Step 6: Mapping of Multivalued attributes.

INPUT:

Entities

Entity Relationship Matrix

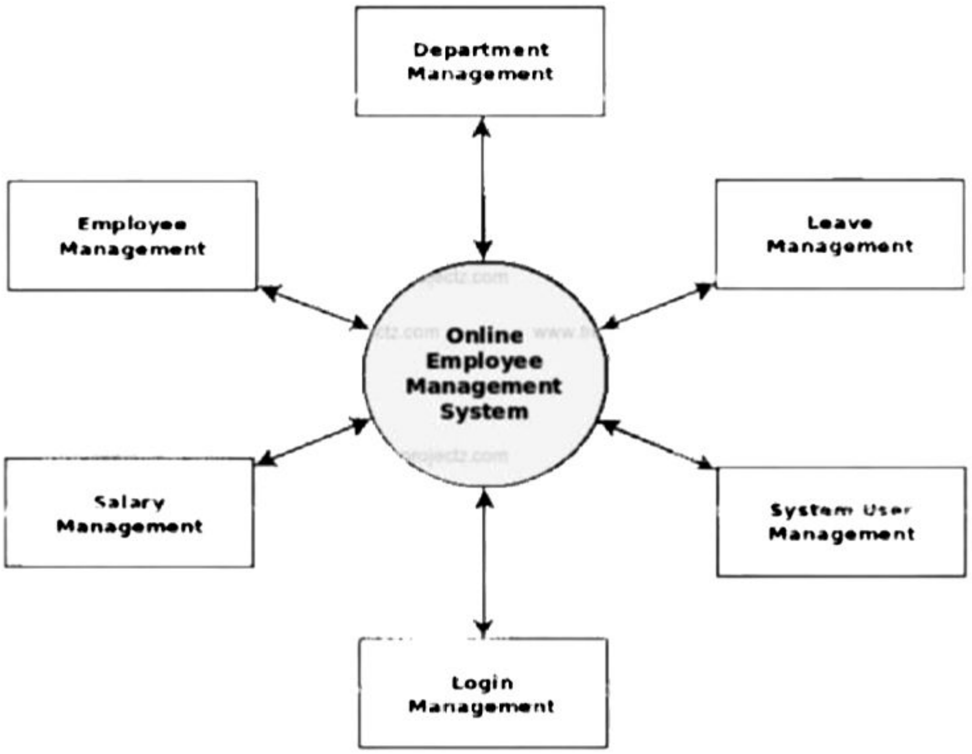
Primary Keys

Attributes

Mapping of Attributes with Entities

Result:

DFD Diagram:


Department Management
Employee Management
Leave Management
Online Employee Management System
Salary Management
System User Management
Login Management

EX NO:4	DRAW THE DATA FLOW DIAGRAMS AT LEVEL 0 AND LEVEL 1
DATE:	

AIM:

To Draw the Data Flow Diagram for any project and List the Modules in the Application.

ALGORITHM:

1. Open the Visual Paradigm to draw DFD (Ex.Lucidchart)
2. Select a data flow diagram template
3. Name the data flow diagram
4. Add an external entity that starts the process
5. Add a Process to the DFD
6. Add a data store to the diagram
7. Continue to add items to the DFD
8. Add data flow to the DFD
9. Name the data flow
10. Customize the DFD with colours and fonts
11. Add a title and share your data flow diagram

INPUT:

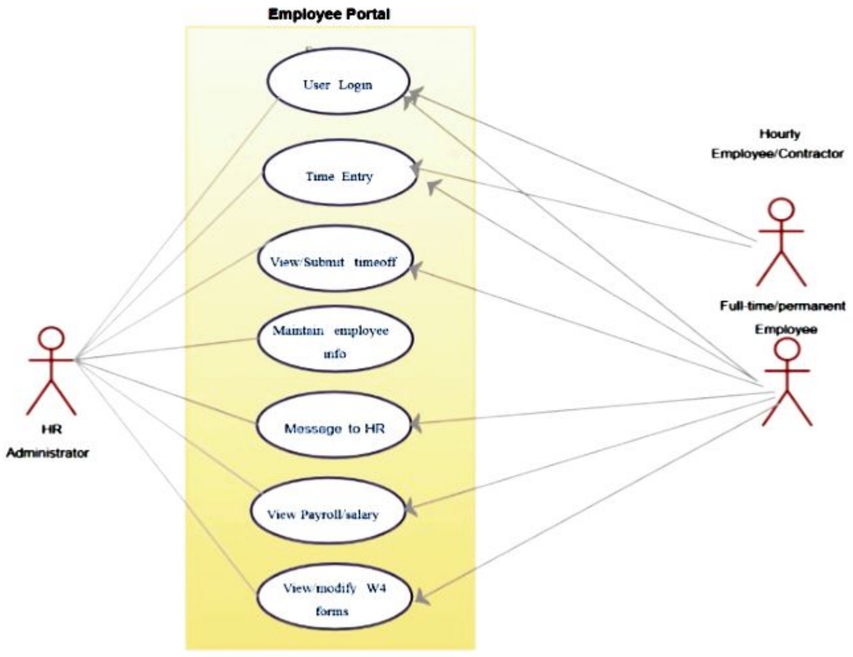
Processes

Datastores

External Entities

Result:

Use Case Diagram:



EX NO:5	**DRAW USE CASE DIAGRAM**
DATE:	

AIM:

To Draw the Use Case Diagram for any project

ALGORITHM:

Step 1: Identify Actors

Step 2: Identify Use Cases

Step 3: Connect Actors and Use Cases

Step 4: Add System Boundary

Step 5: Define Relationships

Step 6: Review and Refine

Step 7: Validate

INPUTS:

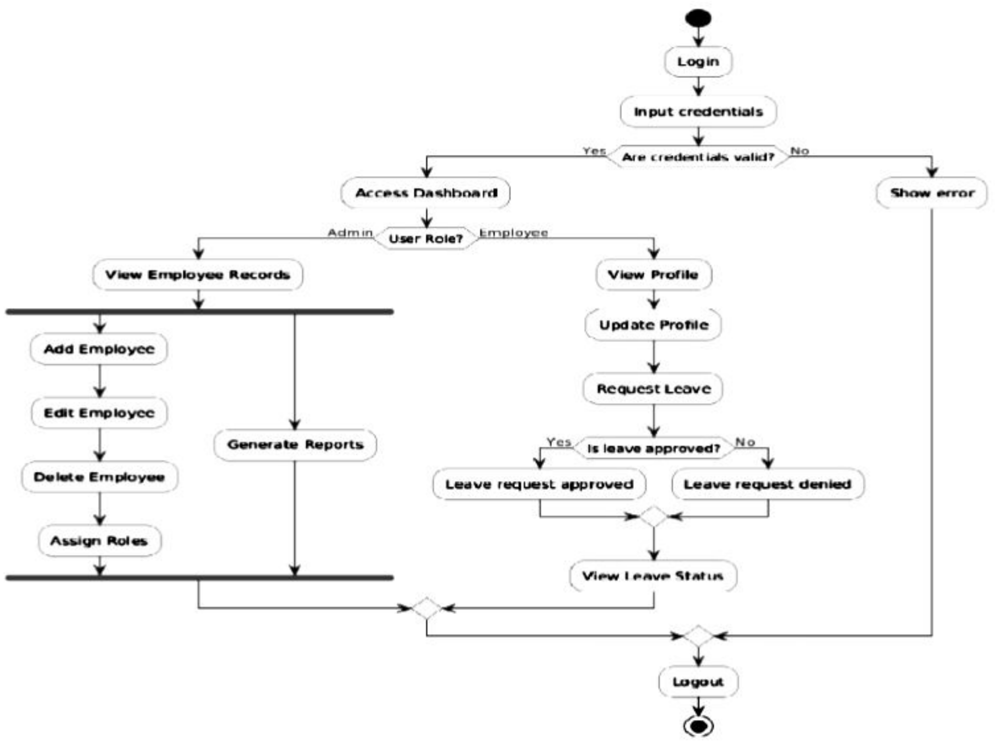
Actors

Use Cases

Relations

Result:

Activity Diagram:



EX NO:6	**DRAW ACTIVITY DIAGRAM OF ALL USE CASES.**
DATE:	

AIM:

To Draw the activity Diagram for Employee Management System project

ALGORITHM:

Step 1: Identify the Initial State and Final States

Step 2: Identify the Intermediate Activities Needed

Step 3: Identify the Conditions or Constraints

Step 4: Draw the Diagram with Appropriate Notations

INPUTS:

Activities

Decision Points

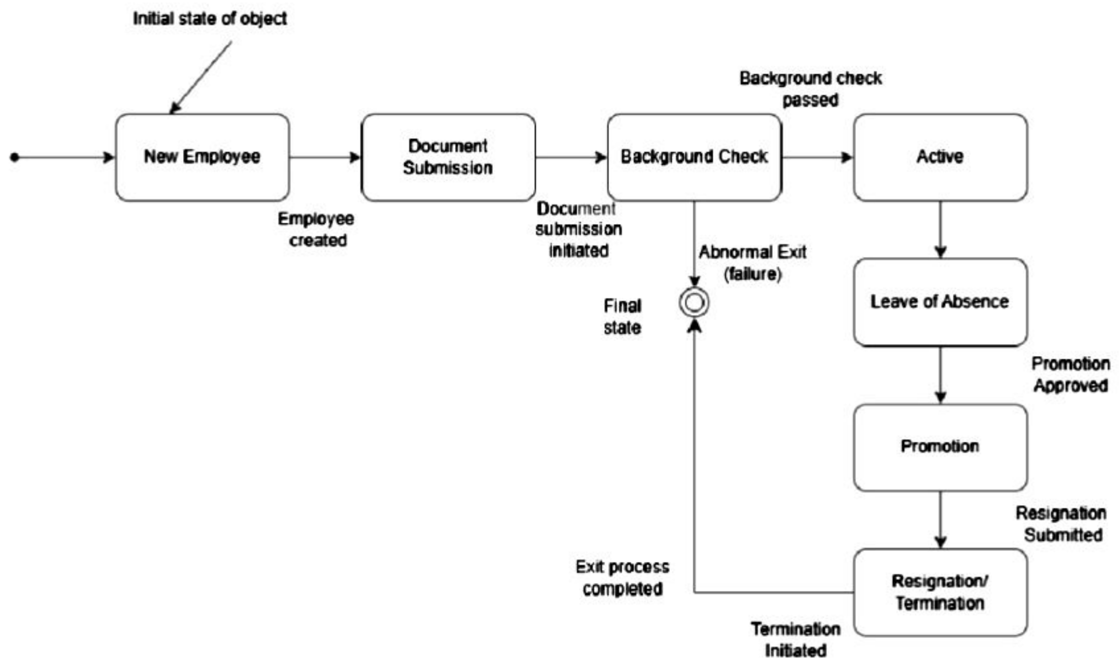
Guards

Parallel Activities

Conditions

Result:

State Chart Diagram:



EX NO:7	**DRAW STATE CHART DIAGRAM OF ALL USE CASES.**
DATE:	

AIM:

To Draw the State Chart Diagram for Employee Management System project

ALGORITHM:

STEP-1: Identify the important objects to be analysed.

STEP-2: Identify the states.

STEP-3: Identify the events.

INPUTS:

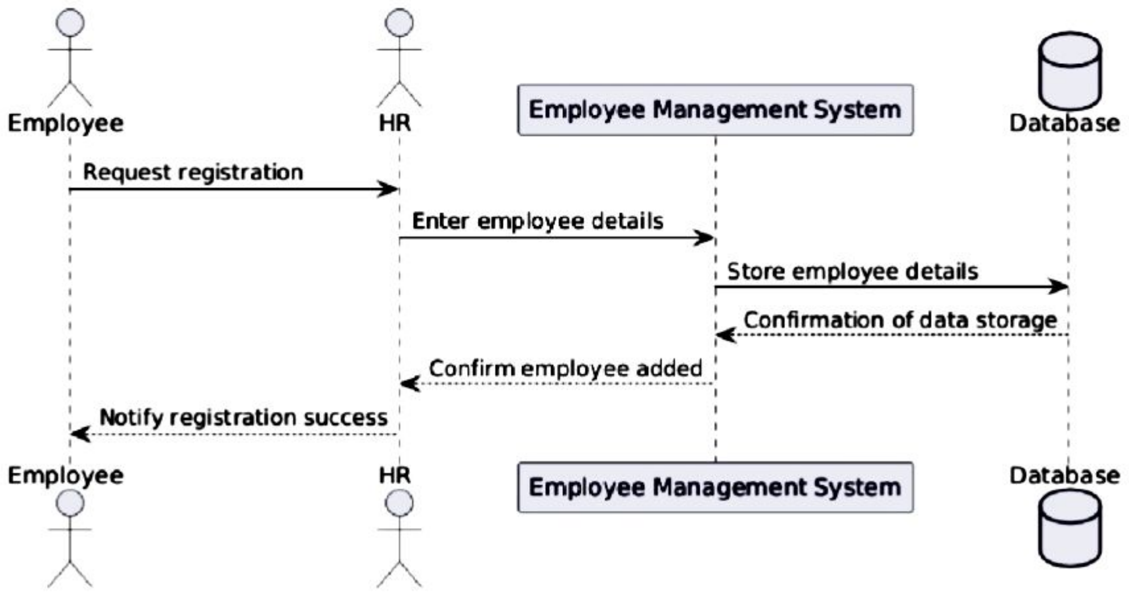
Objects

States

Events

Result:

Sequence Diagram:



EX NO:8	**DRAW SEQUENCE DIAGRAM OF ALL USE CASES.**
DATE:	

AIM: To Draw the Sequence Diagram for Employee Management System project

ALGORITHM:

1. Identify the Scenario

2. List the Participants

3. Define Lifelines

4. Arrange Lifelines

5. Add Activation Bars

6. Draw Messages

7. Include Return Messages

8. Indicate Timing and Order

9. Include Conditions and Loops

10. Consider Parallel Execution

11. Review and Refine

12. Add Annotations and Comments

13. Document Assumptions and Constraints

14. Use a Tool to create a neat sequence diagram

INPUTS:

Objects taking part in the interaction.

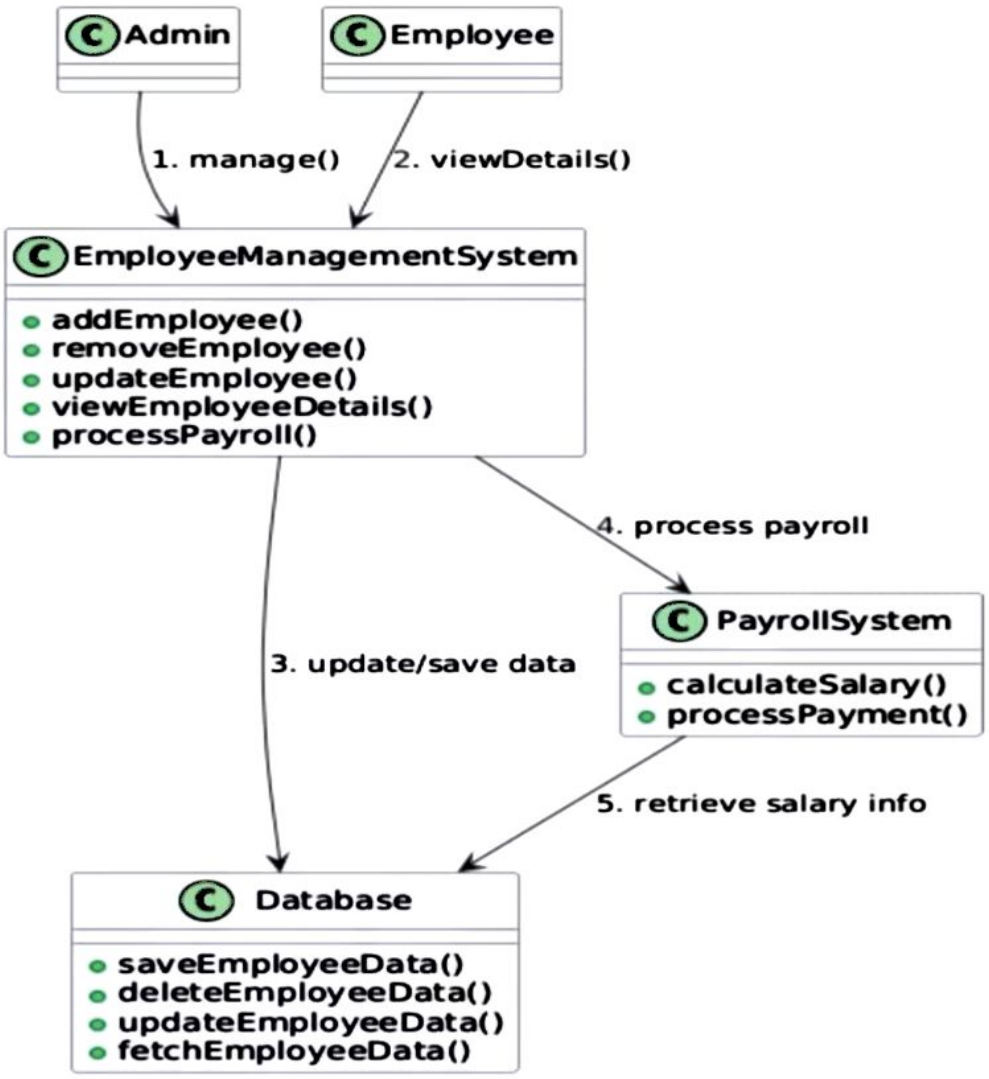
Message flows among the objects.

The sequence in which the messages are flowing.

Object organization.

Result:

Collaboration Diagram:



EX NO:9	**DRAW COLLABORATION DIAGRAM OF ALL USE CASES**
DATE:	

AIM:

To Draw the Collaboration Diagram for Employee Management System project

ALGORITHM:

Step 1: Identify Objects/Participants

Step 2: Define Interactions

Step 3: Add Messages

Step 4: Consider Relationships

Step 5: Document the collaboration diagram along with any relevant explanations or annotations.

INPUTS:

Objects taking part in the interaction.

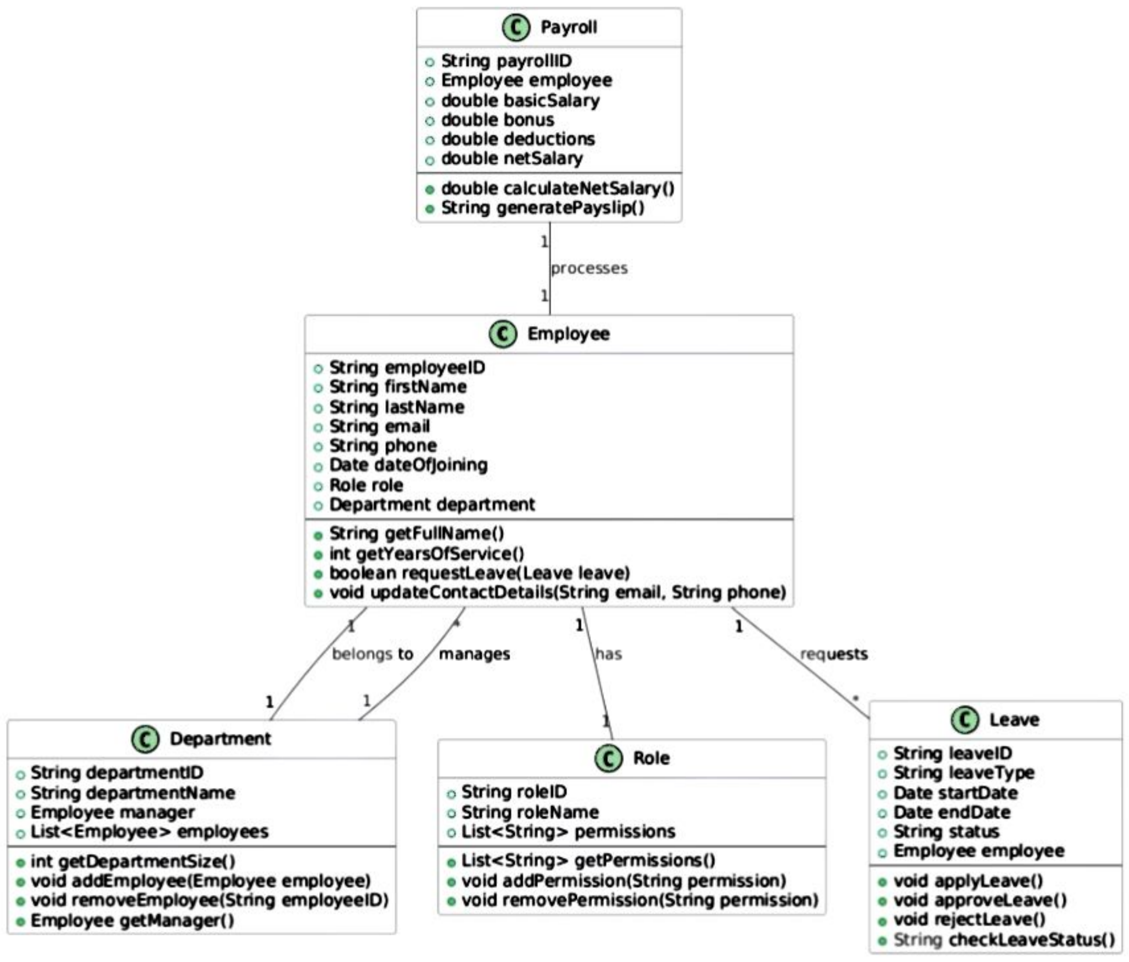
Message flows among the objects.

The sequence in which the messages are flowing.

Object organization.

Result:

Class Diagram:



EX NO:10	**ASSIGN OBJECTS IN SEQUENCE DIAGRAM TO CLASSES AND MAKE CLASS DIAGRAM.**
DATE:	

AIM:

To Draw the Class Diagram for any project

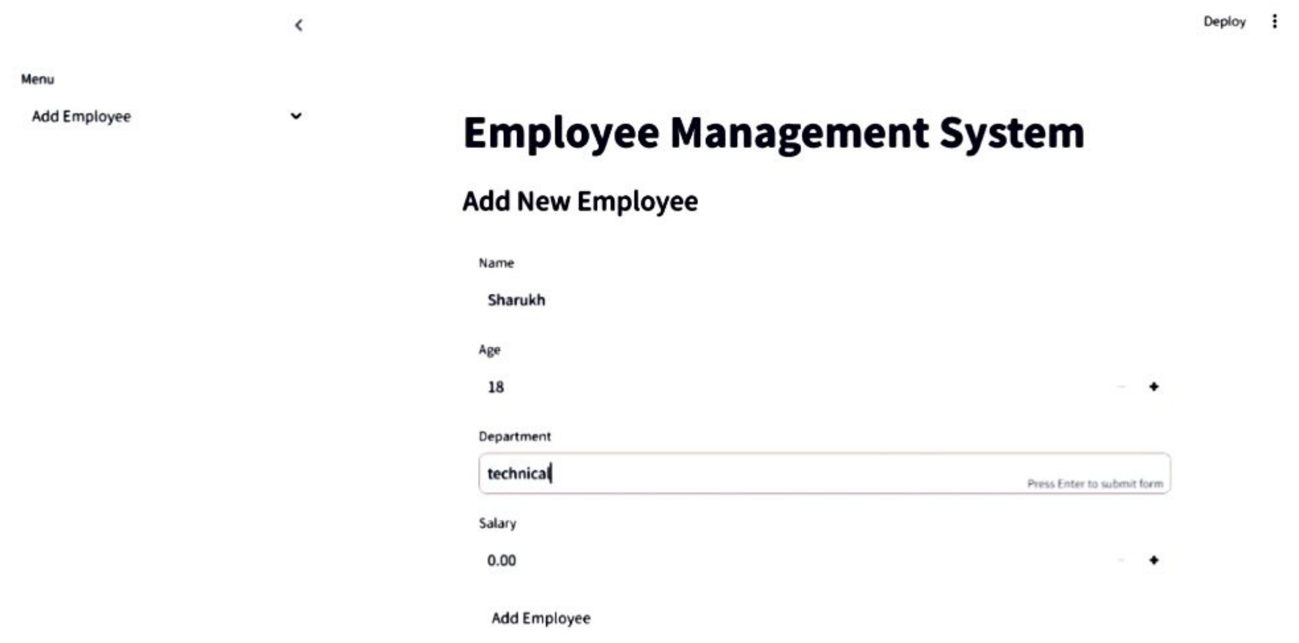
ALGORITHM:

1. Identify Classes
2. List Attributes and Methods
3. Identify Relationships
4. Create Class Boxes
5. Add Attributes and Methods
6. Draw Relationships
7. Label Relationships
8. Review and Refine
9. Use Tools for Digital Drawing

INPUTS:

1. Class Name
2. Attributes
3. Methods
4. Visibility Notation

RESULT:



Menu
Add Employee

Employee Management System

Add New Employee

Name
Sharukh

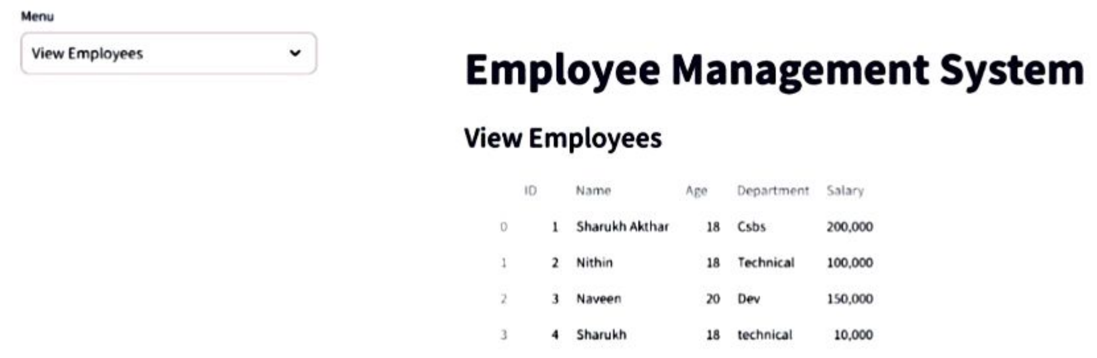
Age
18

Department
technical

Press Enter to submit form

Salary
0.00

Add Employee



Menu
View Employees

Employee Management System

View Employees

	ID	Name	Age	Department	Salary
0	1	Sharukh Akthar	18	Csbs	200,000
1	2	Nithin	18	Technical	100,000
2	3	Naveen	20	Dev	150,000
3	4	Sharukh	18	technical	10,000

EX NO:11	MINI PROJECT-EMPLOYEE MANAGEMENT SYSTEM
DATE:	

Aim:

A comprehensive plan and code implementation to develop an Employee Management System using Streamlit and MySQL that allows HR or administrators to efficiently manage employee details, track employee activity, and notify employees via email when necessary. The focus is on streamlining operations, enhancing convenience, and ensuring a reliable and user-friendly system.

Algorithm:

1. Database Connection Initialization
2. Streamlit Interface Setup
3. Operation Selection

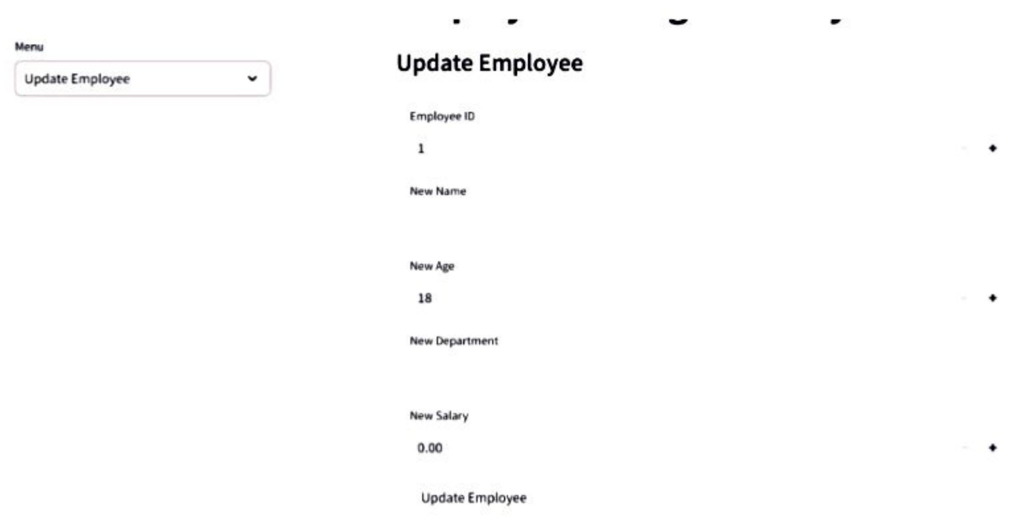
• Update Marks

• Display Marks

4. Database Query Execution
5. Email Notification (for Update Marks)
6. Feedback Display
7. Application Termination

Program:

```
from sqlalchemy.orm import declarative_base, sessionmaker

from sqlalchemy import create_engine, Column, Integer, String, Float

import streamlit as st

import pandas as pd


# Define the base for ORM

Base = declarative_base()


# Define the Employee table

class Employee(Base):

    __tablename__ = 'employees'

    id = Column(Integer, primary_key=True, autoincrement=True)

    name = Column(String(100), nullable=False)

    age = Column(Integer, nullable=False)

    department = Column(String(50), nullable=False)
```


Menu

Update Employee

Update Employee

Employee ID

1

New Name

New Age

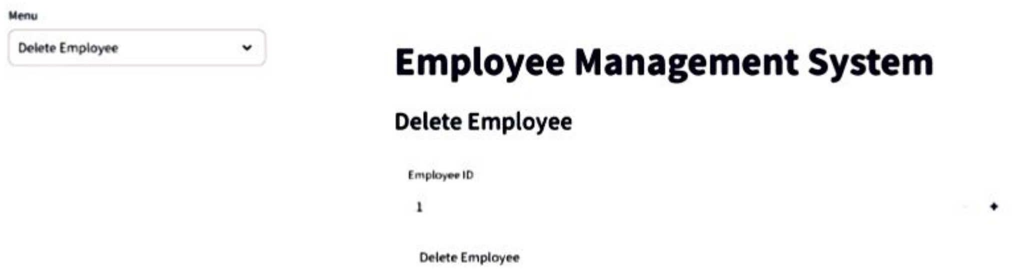
18

New Department

New Salary

0.00

Update Employee


Menu

Delete Employee

Employee Management System

Delete Employee

Employee ID

1

Delete Employee

```
    salary = Column(Float, nullable=False)

# Database setup
DATABASE_URL = "sqlite:///employees.db"
engine = create_engine(DATABASE_URL)
Base.metadata.create_all(engine)
Session = sessionmaker(bind=engine)
session = Session()

# Streamlit app title
st.title("Employee Management System")

# Menu options
menu = ["Add Employee", "View Employees", "Update Employee", "Delete Employee"]
choice = st.sidebar.selectbox("Menu", menu)

# Add Employee
if choice == "Add Employee":
    st.subheader("Add New Employee")
    with st.form("add_employee_form"):
        name = st.text_input("Name")
        age = st.number_input("Age", min_value=18, max_value=100, step=1)
        department = st.text_input("Department")
        salary = st.number_input("Salary", min_value=0.0, step=1000.0)
        submitted = st.form_submit_button("Add Employee")
        if submitted:
            if name and department:
                try:
                    new_employee = Employee(name=name, age=age, department=department, salary=salary)
                    session.add(new_employee)
                    session.commit()
                    st.success(f"Employee {name} added successfully!")
                except Exception as e:
                    st.error(f"Error adding employee: {e}")
```

```
        else:
            st.error("Please fill all fields.")

# View Employees
elif choice == "View Employees":
    st.subheader("View Employees")
    try:
        # Fetch all employees from the database
        employees = session.query(Employee).all()
        if employees:
            # Convert the result to a pandas DataFrame for better display
            data = pd.DataFrame(
                [(emp.id, emp.name, emp.age, emp.department, emp.salary) for emp in employees],
                columns=["ID", "Name", "Age", "Department", "Salary"]
            )
            st.dataframe(data)  # Display the data in a table format
        else:
            st.info("No employees found.")
    except Exception as e:
        st.error(f"Error fetching employees: {e}")

# Update Employee
elif choice == "Update Employee":
    st.subheader("Update Employee")
    with st.form("update_employee_form"):
        emp_id = st.number_input("Employee ID", min_value=1, step=1)
        new_name = st.text_input("New Name")
        new_age = st.number_input("New Age", min_value=18, max_value=100, step=1)
        new_department = st.text_input("New Department")
        new_salary = st.number_input("New Salary", min_value=0.0, step=1000.0)
        submitted = st.form_submit_button("Update Employee")
        if submitted:
            try:
                employee = session.query(Employee).filter(Employee.id == emp_id).first()
```

```
            if employee:
                employee.name = new_name if new_name else employee.name
                employee.age = new_age
                employee.department = new_department if new_department else employee.department
                employee.salary = new_salary
                session.commit()
                st.success("Employee updated successfully!")
            else:
                st.error("Employee not found.")
        except Exception as e:
            st.error(f"Error updating employee: {e}")

# Delete Employee
elif choice == "Delete Employee":
    st.subheader("Delete Employee")
    with st.form("delete_employee_form"):
        emp_id = st.number_input("Employee ID", min_value=1, step=1)
        submitted = st.form_submit_button("Delete Employee")
        if submitted:
            try:
                employee = session.query(Employee).filter(Employee.id == emp_id).first()
                if employee:
                    session.delete(employee)
                    session.commit()
                    st.success("Employee deleted successfully!")
                else:
                    st.error("Employee not found.")
            except Exception as e:
                st.error(f"Error deleting employee: {e}")
```

Conclusion:

The Employee Management System developed using Streamlit and SQLite provides an efficient and user-friendly interface for administrators and HR managers to manage employee data. This system centralizes key functionalities such as adding, updating, and viewing employee details, tracking performance and salary data, and generating notifications or reports for employee updates. By integrating simplicity and functionality, it ensures seamless management of employee records, improving productivity and decision-making for organizations.