

EMPLOYEE MANAGEMENT SYSTEM -PYTHON PROJECT

Group Project By :-

Name: **APRAJITA (RA2311026030003)**

ANANYA RAJ (RA2311026030030)

COURSE: Bachelor of Technology

CLASS: CSE(AIML)

SECTION: A

SUBJECT: Advanced Programming Practice (21CSC203P)

YEAR: 2023-27

A Python Application with MySQL Integration

- Objective:** To develop a simple employee management system with GUI and database.
- Overview:** The system allows for creating a database, managing employee records (CRUD operations), and displaying results.

Technologies Used-

- Programming Language: Python
- Library:
 - `mysql.connector`: For MySQL database connectivity.
 - Tkinter : for GUI
- Database: MySQL
- Database: MySQL

Database Design

- **Database Name:** employee
- **Table:** emp
- **Columns:**
 - id: Integer (Primary Key)
 - ename: Varchar(15)
 - salary: Float
- **Functionality:** Stores employee details, enabling CRUD operations.

Key Features

- **CRUD Operations:**
 - **Create Database & Table:** Create the employee database and employee table.
 - **Insert Records:** Add new employee records.
 - **Update Records:** Modify existing employee details.
 - **Delete Records:** Remove employee records.
 - **Search Records:** Find employee records by ID, name, or salary.
 - **Display Records:** Show all employee records in a formatted manner.

Menu Function

- **Function:** menu()
- **Purpose:** Displays the main menu and handles user choices.
- **Loop:** Continues until the user decides to exit.

Database Functions Overview

- **Function Calls:**

- `create_database()`: Creates the employee database.
- `show_databases()`: Lists all databases.
- `create_table()`: Creates the employee table.
- `show_tables()`: Lists all tables in the current database.
- `insert_record()`: Inserts a new employee record.
- `update_record()`: Updates an existing employee record.
- `delete_record()`: Deletes an employee record.
- `search_record()`: Searches for records based on user criteria.
- `display_record()`: Displays all employee records.

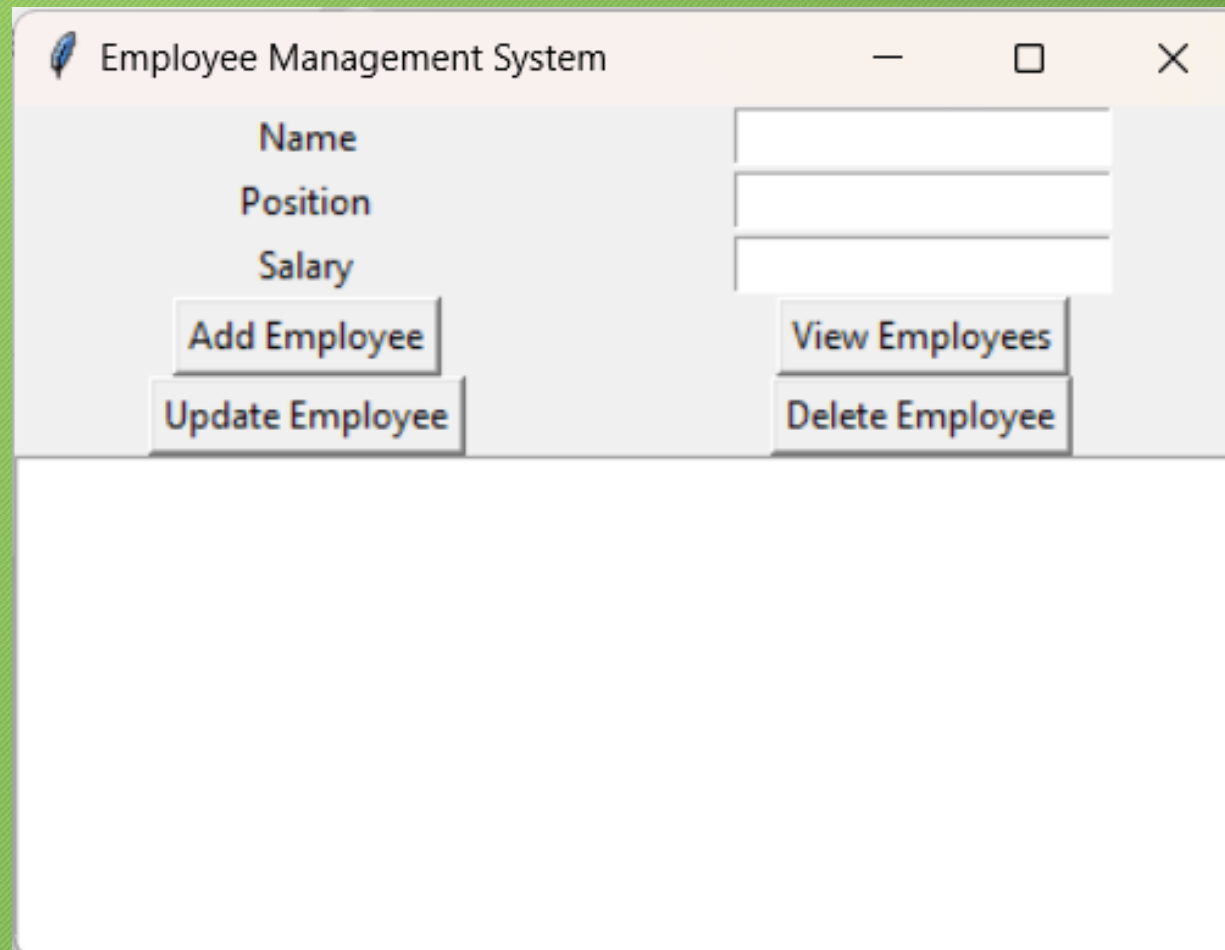
Example Code - Search Record

```
def search_record():
    con = driver.connect(
        host='localhost',
        user='root',
        passwd='mysql@27_85$',
        charset='utf8',
        database='employee'
    )
    cur = con.cursor()
    print("ENTER THE CHOICE ACCORDING TO YOU WANT TO SEARCH RECORD: ")
    print("1. ACCORDING TO ID")
    print("2. ACCORDING TO NAME")
    print("3. ACCORDING TO SALARY")
    print()
    choice = int(input("ENTER THE CHOICE (1-3) : "))

    if choice == 1:
        d = int(input("Enter Employee ID which you want to search : "))
        query1 = "select * from emp where id=%s" % (d)
    elif choice == 2:
        name = input("Enter Employee Name which you want to search : ")
        query1 = "select * from emp where ename='%s'" % (name)
    elif choice == 3:
        sal = float(input("Enter Employee Salary which you want to search : "))
        query1 = "select * from emp where salary=%s" % (sal)
    else:
        print("Wrong Choice")

    cur.execute(query1)
    rec = cur.fetchall()
    count = cur.rowcount
    print("Total no. of records found: ", count)
    for i in rec:
        print(i)
    print("Record Searched")
    con.close()
```


GUI OF THE APPLICATION



The screenshot displays a window titled "Employee Management System" with standard window controls (minimize, maximize, close). The interface is divided into two main sections. The top section contains three input fields for "Name", "Position", and "Salary", each with a corresponding label to its left. Below these fields are four buttons: "Add Employee" and "Update Employee" on the left, and "View Employees" and "Delete Employee" on the right. The bottom section of the window is a large, empty white area, likely intended for displaying a list of employees or other data.

| Field | Input |
|----------|----------------------|
| Name | <input type="text"/> |
| Position | <input type="text"/> |
| Salary | <input type="text"/> |

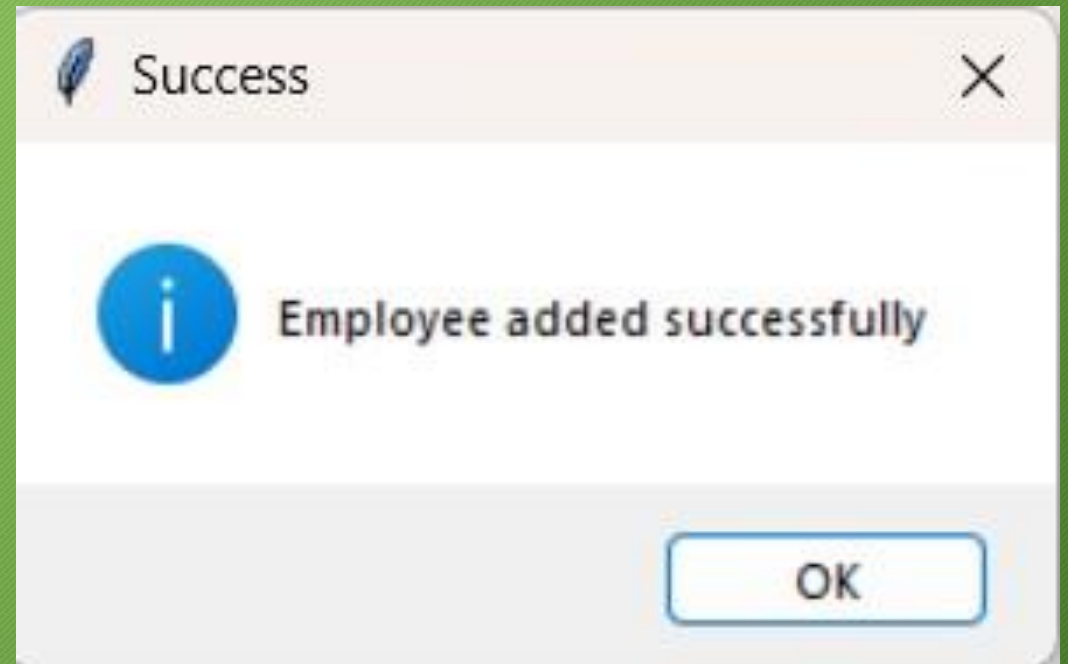
Buttons:

- Add Employee
- Update Employee
- View Employees
- Delete Employee

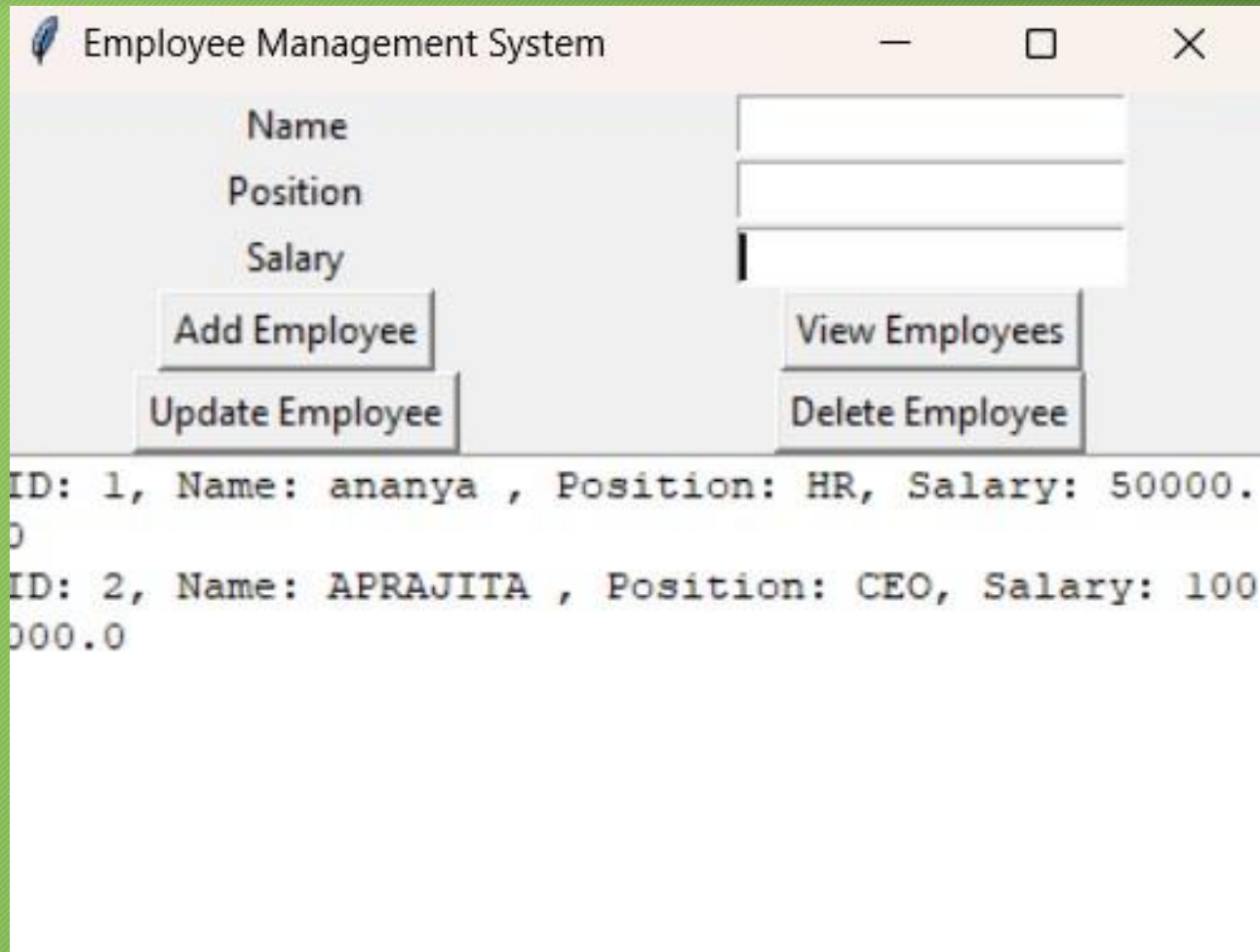
Adding Employee Details-

Employee Management System

| | |
|----------|-------------------------------------|
| Name | <input type="text" value="ananya"/> |
| Position | <input type="text" value="HR"/> |
| Salary | <input type="text" value="50000"/> |



Viewing Employee Details-



Employee Management System

Name

Position

Salary

Add Employee View Employees

Update Employee Delete Employee

ID: 1, Name: ananya , Position: HR, Salary: 50000.
0

ID: 2, Name: APRAJITA , Position: CEO, Salary: 100
000.0

Error Handling

- **Error Handling Practices:**
 - Check database connection status before executing queries.
 - Prompt user with relevant messages on errors.
 - Ensure the application does not crash due to exceptions.

CONCLUSION

- The Employee Management System implemented in Python provides a comprehensive solution for managing employee records through a command-line interface. By utilizing MySQL for database management, the system effectively supports key operations such as creating databases, adding, updating, deleting, and searching for employee records. This project not only demonstrates foundational programming and database skills but also serves as a practical tool for managing employee data efficiently

THANK YOU