

**A  
Project Report  
On  
Employees Management System**



**by  
Jitendra Kumar Dubey  
(206612)  
Under the guidance  
Of  
Sachin Vasav**

# Introduction

The Employee Management System (EMS) is a comprehensive software application designed to streamline and automate various employee-related tasks within an organization. This project aims to simplify the management of employee data, from recruitment and onboarding to performance tracking, salary management, and employee exits.

Managing employees is one of the most critical aspects of any business, as employees are the backbone of an organization's productivity and growth. As businesses grow, handling employee records and tracking performance manually becomes cumbersome and prone to errors. Therefore, the need for an efficient, automated system becomes crucial.

This project focuses on providing a user-friendly interface for both administrators and HR personnel to manage essential employee details like personal information, employment history, attendance, salary records, and performance reviews. Additionally, the system will ensure compliance with labour regulations by storing accurate and up-to-date employee information.

The Employee Management System is built using Python for the backend and MySQL for database management, ensuring a robust, scalable, and secure solution for managing employee records. It offers key functionalities such as user registration, login authentication, employee data management, and role-based access, making it a comprehensive tool for organizations of all sizes.

This project not only addresses the challenges of managing a growing workforce but also enhances overall organizational efficiency by enabling real-time access to employee information, reducing paperwork, and improving decision-making processes related to human resources.

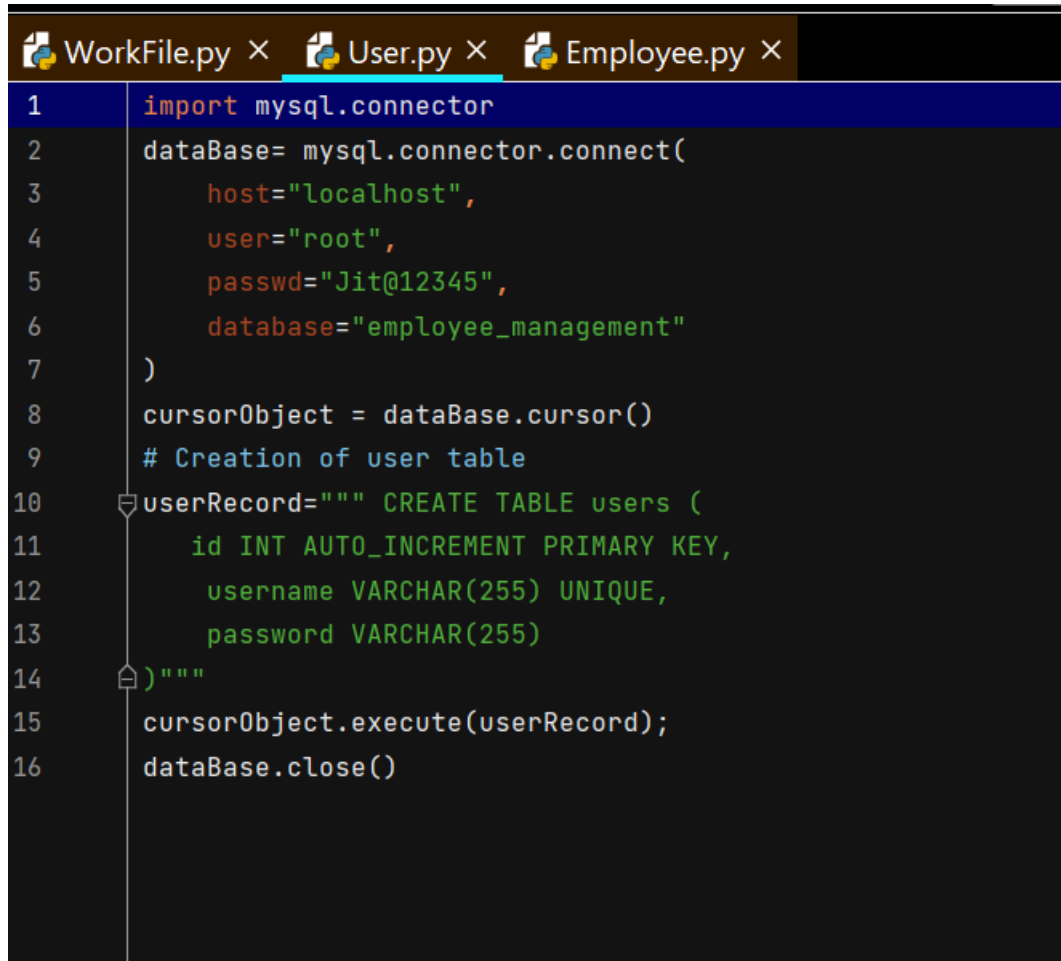
# Importance of Employee Management:

- **Increases Productivity:** Proper employee management ensures that employees are focused, efficient, and working towards organizational goals.
- **Reduces Turnover:** Engaged and satisfied employees are less likely to leave, reducing recruitment costs and retaining institutional knowledge.
- **Improves Workplace Culture:** Employee management fosters a culture of respect, teamwork, and collaboration, creating a positive and motivating environment.
- **Ensures Compliance:** Managing employees in line with legal requirements helps to avoid fines and legal disputes.
- **Drives Business Growth:** When employees are properly managed, they are more likely to innovate and contribute to the success of the company.

Many companies use Employee Management Systems (EMS) or Human Resource Management Systems (HRMS), which are software solutions to help automate and streamline many of these processes. These systems typically include features for payroll, employee data management, performance tracking, and more.

Effective employee management leads to better employee satisfaction, enhanced productivity, and the overall success of the organization.

# Code Snippets



The image shows a code editor with three tabs: 'WorkFile.py', 'User.py', and 'Employee.py'. The 'User.py' tab is active. The code is as follows:

```
1 import mysql.connector
2 dataBase= mysql.connector.connect(
3     host="localhost",
4     user="root",
5     passwd="Jit@12345",
6     database="employee_management"
7 )
8 cursorObject = dataBase.cursor()
9 # Creation of user table
10 userRecord=""" CREATE TABLE users (
11     id INT AUTO_INCREMENT PRIMARY KEY,
12     username VARCHAR(255) UNIQUE,
13     password VARCHAR(255)
14 )"""
15 cursorObject.execute(userRecord);
16 dataBase.close()
```

WorkFile.py × User.py × Employee.py ×

```
1 import mysql.connector
2 dataBase =mysql.connector.connect(
3     host="localhost",
4     user="root",
5     passwd="Jit@12345"
6 )
7 cursorObject = dataBase.cursor()
8 # employee_management database creation
9 cursorObject.execute("CREATE DATABASE employee_management")
10 dataBase.close()
11 dataBase= mysql.connector.connect(
12     host="localhost",
13     user="root",
14     passwd="Jit@12345",
15     database="employee_management"
16 )
17 cursorObject = dataBase.cursor()
18 # creating a employee table
19 employeeRecord=""" CREATE TABLE employees (
20     id INT AUTO_INCREMENT PRIMARY KEY,
21     name VARCHAR(255),
22     age INT,
23     department VARCHAR(255),
24     salary DECIMAL(10, 2)
25 )"""
26 cursorObject.execute(employeeRecord);
27 dataBase.close()
```

```
WorkFile.py × User.py × Employee.py ×
1 import mysql.connector
2 import bcrypt
3
4 # making a function to connect database
5 def connect_to_db():
6     return mysql.connector.connect(
7         host="localhost",
8         user="root",
9         password="Jit@12345",
10        database="employee_management"
11    )
12
13 # hashing
14 def hash_password(password):
15     return bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())
16
17 # for verifying password
18 def verify_password(stored_password, provided_password):
19     return bcrypt.checkpw(provided_password.encode('utf-8'), stored_password.encode('utf-8'))
20
21 # Check user is exist in database or not
22 def user_exists(username):
23     conn = connect_to_db()
24     cursor = conn.cursor()
25     query = "SELECT * FROM users WHERE username = %s"
26     cursor.execute(query, (username,))
```

```
WorkFile.py × User.py × Employee.py ×
26     cursor.execute(query, (username,))
27     result = cursor.fetchone()
28     cursor.close()
29     conn.close()
30     return result is not None
31
32 # or
33 # if result:
34 #     return True
35 # else:
36 #     return False
37
38 # Function to register a new employee user
39 def register_employee_user():
40     username = input("Enter a new username: ")
41     if user_exists(username):
42         print("Username already exists. Try logging in.")
43         return False
44     password = input("Enter a new password: ")
45     hashed_password = hash_password(password)
46     conn = connect_to_db()
47     cursor = conn.cursor()
48
49     # Insert the new user into the users table
50     query = "INSERT INTO users (username, password) VALUES (%s, %s)"
51     cursor.execute(query, (username, hashed_password.decode('utf-8')))
```

```

WorkFile.py × User.py × Employee.py ×
52     print("User registered successfully!")
53     # Once it registered, then we will add Employee details
54     name = input("Enter employee name: ")
55     age = int(input("Enter employee age: "))
56     department = input("Enter employee department: ")
57     salary = float(input("Enter employee salary: "))
58
59     query = "INSERT INTO employees (name, age, department, salary) VALUES (%s, %s, %s, %s)"
60     cursor.execute(query, (name, age, department, salary))
61     conn.commit()
62     print("Employee registered successfully!")
63     cursor.close()
64     conn.close()
65     return True
66
67
68     # Function to log in the user
69     def login():
70         username = input("Enter username: ")
71         password = input("Enter password: ")
72
73         conn = connect_to_db()
74         cursor = conn.cursor()
75
76         query = "SELECT password FROM users WHERE username = %s"
77         cursor.execute(query, (username,))

```

```

WorkFile.py × User.py × Employee.py ×
77         cursor.execute(query, (username,))
78         # Here we are fetching the password
79         result = cursor.fetchone()
80
81         cursor.close()
82         conn.close()
83
84         if result:
85             stored_password = result[0]
86             if verify_password(stored_password, password):
87                 print("Login successful!")
88                 return True
89             else:
90                 print("Incorrect password. Try again.")
91                 return False
92         else:
93             print("Username not found. Would you like to register as a new employee?")
94             register_choice = input("Enter 'yes' to register or 'no' to cancel: ").lower()
95             if register_choice == 'yes':
96                 return register_employee_user()
97             else:
98                 return False
99
100
101     # Function to add a new employee (for general purposes, not the login/registration part)
102     def add_employee():

```

```
WorkFile.py × User.py × Employee.py ×
101 # Function to add a new employee (for general purposes, not the login/registration part)
102 def add_employee():
103     name = input("Enter employee name: ")
104     age = int(input("Enter employee age: "))
105     department = input("Enter employee department: ")
106     salary = float(input("Enter employee salary: "))
107
108     conn = connect_to_db()
109     cursor = conn.cursor()
110
111     query = "INSERT INTO employees (name, age, department, salary) VALUES (%s, %s, %s, %s)"
112     cursor.execute(query, (name, age, department, salary))
113
114     conn.commit()
115     print("Employee added successfully!")
116     cursor.close()
117     conn.close()
118
119
120 # Function to view all employees
121 def view_employees():
122     conn = connect_to_db()
123     cursor = conn.cursor()
124
125     query = "SELECT * FROM employees"
126     cursor.execute(query)
```

```
WorkFile.py × User.py × Employee.py ×
126     cursor.execute(query)
127     result = cursor.fetchall()
128
129     if result:
130         print("\nEmployee List:")
131         for row in result:
132             print(f"ID: {row[0]}, Name: {row[1]}, Age: {row[2]}, Department: {row[3]}, Salary: {row[4]}")
133     else:
134         print("No employees found.")
135
136     cursor.close()
137     conn.close()
138
139
140 # Function to update employee details
141 def update_employee():
142     emp_id = int(input("Enter employee ID to update: "))
143
144     print("What do you want to update?")
145     print("1. Name")
146     print("2. Age")
147     print("3. Department")
148     print("4. Salary")
149
150     choice = int(input("Enter choice: "))
151
152     conn = connect_to_db()
153     user_exists()
```



```
WorkFile.py × User.py × Employee.py ×
152     conn = connect_to_db()
153     cursor = conn.cursor()
154
155     if choice == 1:
156         new_name = input("Enter new name: ")
157         query = "UPDATE employees SET name = %s WHERE id = %s"
158         cursor.execute(query, (new_name, emp_id))
159     elif choice == 2:
160         new_age = int(input("Enter new age: "))
161         query = "UPDATE employees SET age = %s WHERE id = %s"
162         cursor.execute(query, (new_age, emp_id))
163     elif choice == 3:
164         new_department = input("Enter new department: ")
165         query = "UPDATE employees SET department = %s WHERE id = %s"
166         cursor.execute(query, (new_department, emp_id))
167     elif choice == 4:
168         new_salary = float(input("Enter new salary: "))
169         query = "UPDATE employees SET salary = %s WHERE id = %s"
170         cursor.execute(query, (new_salary, emp_id))
171     else:
172         print("Invalid choice.")
173         return
174
175     conn.commit()
176     print("Employee updated successfully!")
177     cursor.close()
178     conn.close()
```

```
WorkFile.py × User.py × Employee.py ×
177     cursor.close()
178     conn.close()
179
180     # Function to delete an employee
181     def delete_employee():
182         emp_id = int(input("Enter employee ID to delete: "))
183
184         conn = connect_to_db()
185         cursor = conn.cursor()
186
187         query = "DELETE FROM employees WHERE id = %s"
188         cursor.execute(query, (emp_id,))
189
190         conn.commit()
191         print("Employee deleted successfully!")
192         cursor.close()
193         conn.close()
194
195     # Function to search for an employee by ID or name
196     def search_employee():
197         print("Search by:")
198         print("1. ID")
199         print("2. Name")
200
201         choice = int(input("Enter choice: "))
202
203
```

```
projectmanagement\obj\Scripts\WorkFile.py
WorkFile.py x User.py x Employee.py x
201
202     choice = int(input("Enter choice: "))
203
204     conn = connect_to_db()
205     cursor = conn.cursor()
206
207     if choice == 1:
208         emp_id = int(input("Enter employee ID: "))
209         query = "SELECT * FROM employees WHERE id = %s"
210         cursor.execute(query, (emp_id,))
211     elif choice == 2:
212         emp_name = input("Enter employee name: ")
213         query = "SELECT * FROM employees WHERE name LIKE %s"
214         cursor.execute(query, (f"%{emp_name}%",))
215     else:
216         print("Invalid choice.")
217         return
218
219     result = cursor.fetchall()
220
221     if result:
222         for row in result:
223             print(f"ID: {row[0]}, Name: {row[1]}, Age: {row[2]}, Department: {row[3]}, Salary: {row[4]}")
224     else:
225         print("Employee not found.")
226
227     cursor.close()
```

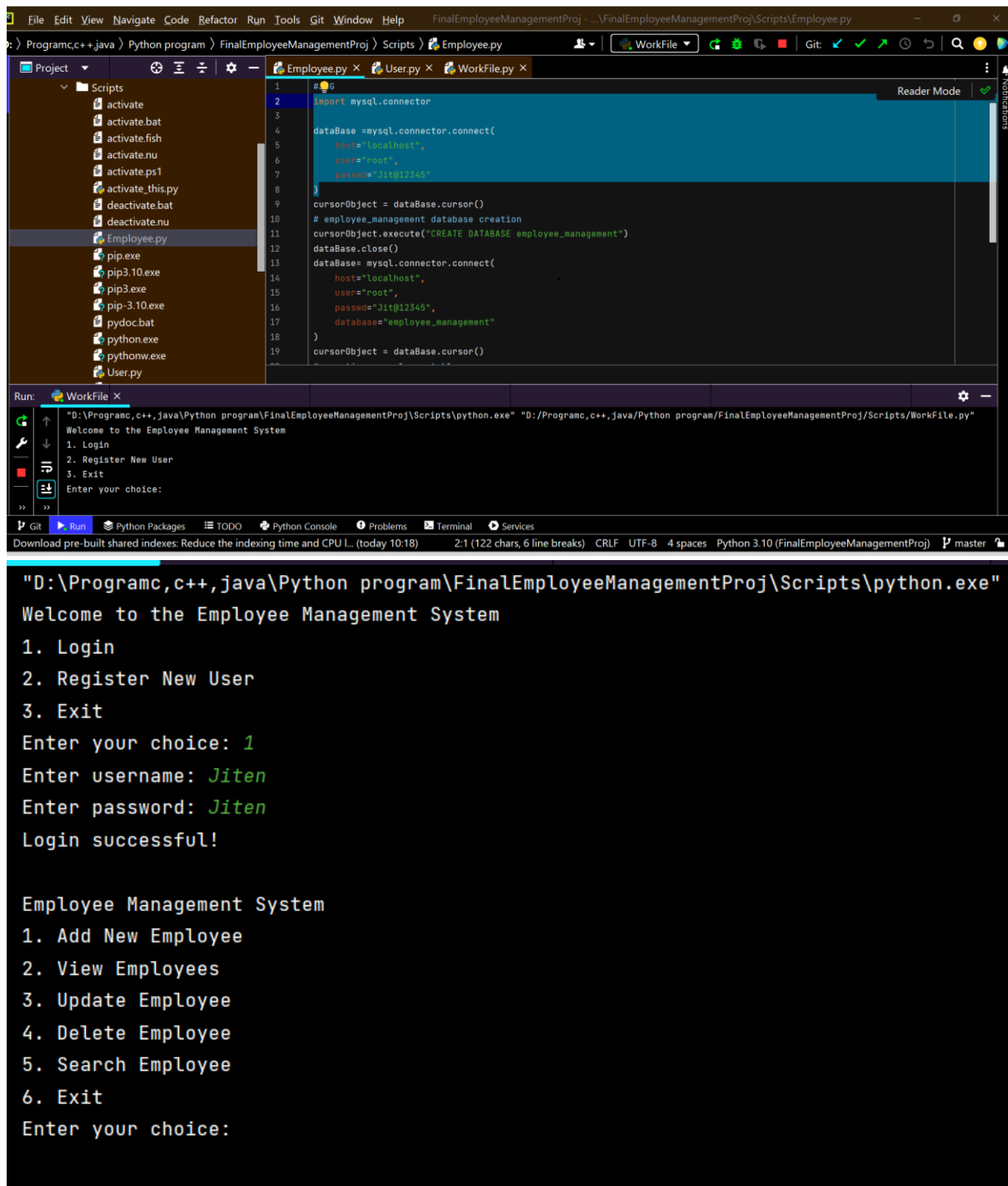
```
WorkFile.py x User.py x Employee.py x
227     cursor.close()
228     conn.close()
229
230
231     # Menu
232     def main_menu():
233         while True:
234             print("\nEmployee Management System")
235             print("1. Add New Employee")
236             print("2. View Employees")
237             print("3. Update Employee")
238             print("4. Delete Employee")
239             print("5. Search Employee")
240             print("6. Exit")
241
242             choice = int(input("Enter your choice: "))
243
244             if choice == 1:
245                 add_employee()
246             elif choice == 2:
247                 view_employees()
248             elif choice == 3:
249                 update_employee()
250             elif choice == 4:
251                 delete_employee()
252             elif choice == 5:
253                 search_employee()
```

```

WorkFile.py × User.py × Employee.py ×
253     search_employee()
254     elif choice == 6:
255         print("Exiting program...")
256         break
257     else:
258         print("Invalid choice so please choose again.")
259
260
261 # Code for start the program
262 # if __name__ == "__main__":
263     print("Welcome to the Employee Management System")
264     while True:
265         print("1. Login")
266         print("2. Register New User")
267         print("3. Exit")
268         choice = int(input("Enter your choice: "))
269
270         if choice == 1:
271             if login():
272                 main_menu()
273         elif choice == 2:
274             register_employee_user()
275         elif choice == 3:
276             print("Exiting the program.")
277             break
278         else:
279             print("Invalid choice! Please try again.")

```

# Output snippets



The image shows a screenshot of an IDE (Visual Studio Code) with a Python file named `Employee.py` open. The code connects to a MySQL database, creates a database named `employee_management`, and then connects to it. The terminal output shows the program's execution, including a welcome message, a menu, and user input for login.

```
1 # coding: utf-8
2 import mysql.connector
3
4 dataBase = mysql.connector.connect(
5     host="localhost",
6     user="root",
7     password="Jit@12345"
8 )
9
10 cursorObject = dataBase.cursor()
11 # employee_management database creation
12 cursorObject.execute("CREATE DATABASE employee_management")
13 dataBase.close()
14 dataBase = mysql.connector.connect(
15     host="localhost",
16     user="root",
17     password="Jit@12345",
18     database="employee_management"
19 )
20 cursorObject = dataBase.cursor()
```

Run: WorkFile x

"D:\Programc,c++,java\Python program\FinalEmployeeManagementProj\Scripts\python.exe" "D:/Programc,c++,java/Python program/FinalEmployeeManagementProj/Scripts/WorkFile.py"

Welcome to the Employee Management System

1. Login

2. Register New User

3. Exit

Enter your choice:

1

Enter username: Jiten

Enter password: Jiten

Login successful!

Employee Management System

1. Add New Employee

2. View Employees

3. Update Employee

4. Delete Employee

5. Search Employee

6. Exit

Enter your choice:

```
Employee Management System
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Enter your choice: 1
Enter employee name: Suraj
Enter employee age: 24
Enter employee department: MCA
Enter employee salary: 25000
Employee added successfully!
```

```
Employee Management System
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
```

```
6. Exit
```

```
Enter your choice: 2
```

```
Employee List:
```

```
ID: 1, Name: Jiten, Age: 23, Department: IT, Salary: 2000000.00
ID: 2, Name: Anshul, Age: 23, Department: IT, Salary: 85000.00
ID: 3, Name: Kapil, Age: 24, Department: CS, Salary: 54000.00
ID: 5, Name: abc, Age: 21, Department: jk, Salary: 7899.00
ID: 6, Name: mansi, Age: 23, Department: IT, Salary: 70000.00
ID: 7, Name: jittu, Age: 23, Department: iy, Salary: 1000000.00
ID: 8, Name: Rohit, Age: 24, Department: HR, Salary: 450000.00
ID: 10, Name: Suraj, Age: 24, Department: MCA, Salary: 25000.00
```

```
Employee Management System
```

```
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
```

```
Enter your choice:
```

Run Python Packages TODO Python Console Problems Terminal Services

```
Employee Management System
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Enter your choice: 3
Enter employee ID to update: 10
What do you want to update?
1. Name
2. Age
3. Department
4. Salary
Enter choice: 4
Enter new salary: 50000
Employee updated successfully!
```

```
Employee Management System
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Enter your choice:
```

```
Employee Management System
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Enter your choice: 4
Enter employee ID to delete: 5
Employee deleted successfully!
```

```
Employee Management System
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Enter your choice:
```

```
Employee Management System
1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Enter your choice: 5
Search by:
1. ID
2. Name
Enter choice: 1
Enter employee ID: 8
ID: 8, Name: Rohit, Age: 24, Department: HR, Salary: 450000.00
```

## Employee Management System

1. Add New Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit

Enter your choice: 5

Search by:

1. ID
2. Name

Enter choice: 2

Enter employee name: Rohit

ID: 8, Name: Rohit, Age: 24, Department: HR, Salary: 450000.00

Enter your choice: 6

Exiting program....

1. Login
2. Register New User
3. Exit

Enter your choice: 2

Enter a new username: NewUser

Enter a new password: new123

User registered successfully!

Enter employee name: NewUser

Enter employee age: 43

Enter employee department: Management

Enter employee salary: 400000

Employee registered successfully!

1. Login
2. Register New User
3. Exit