EMPLOYEE PAYROLL MANAGEMENT SYSTEM

Using Python and MySQL

A Project Report

Computer Science (083)

TABLE OF CONTENTS

1. INTRODUCTION

Managing employee salary records manually is difficult and time-consuming. Errors in salary calculation may lead to dissatisfaction among employees. To overcome these problems, an Employee Payroll Management System using Python and MySQL database connectivity has been developed.

This project stores employee details permanently in a database and calculates salary efficiently using a computerized system. The system provides a user-friendly menu-driven interface to perform various operations on employee records.

2. OBJECTIVES OF THE PROJECT

The objectives of this project are:

- To understand Python–MySQL database connectivity
- To store employee data permanently in a database
- To perform insert, update, delete, and display operations
- To calculate employee salary automatically
- To develop a menu-driven database application

3. SOFTWARE AND HARDWARE REQUIREMENTS

Software Requirements

- Python 3.x
- MySQL Server
- MySQL Connector for Python
- Python IDE (IDLE / VS Code)

Hardware Requirements

- Computer / Laptop
- Minimum 2 GB RAM
- Keyboard and Mouse

4. DATABASE DESIGN

Database Name:

company

Table Name:

employee

Table Structure:

Field Name	Data Type
emp_id	INT (Primary Key)
emp_name	VARCHAR(30)
basic_salary	INT
allowance	INT
total_salary	INT

SQL Command to Create Database & Table:

```
CREATE DATABASE company;
USE company;
CREATE TABLE employee (
    emp_id INT PRIMARY KEY,
    emp_name VARCHAR(30),
    basic_salary INT,
    allowance INT,
    total_salary INT
);
```

5. THEORY / WORKING OF THE PROJECT

This project is a menu-driven application that connects Python with MySQL using the mysql.connector library. The system provides a simple console-based interface for managing employee records.

The system allows the user to:

1. Add employee records - Insert new employee data with automatic salary calculation
2. Display employee records - View all stored employee information
3. Update salary details - Modify basic salary and allowance for existing employees
4. Delete employee records - Remove employee records from the database
5. Exit the program - Safely close the application

All employee data is stored permanently in the MySQL database, ensuring data persistence across sessions. The total salary is automatically calculated as the sum of basic salary and allowance.

6. SOURCE CODE (PYTHON + MYSQL CONNECTIVITY)

Python Source Code:

```python
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="root",
    database="company"
)
mycursor = mydb.cursor()

def add_employee():
    eid = int(input("Enter Employee ID: "))
    name = input("Enter Employee Name: ")
    basic = int(input("Enter Basic Salary: "))
    allow = int(input("Enter Allowance: "))
    total = basic + allow
    sql = "INSERT INTO employee VALUES (%s, %s, %s, %s, %s)"
    val = (eid, name, basic, allow, total)
    mycursor.execute(sql, val)
    mydb.commit()
    print("Employee Record Added Successfully")

def display_employee():
    mycursor.execute("SELECT * FROM employee")
    result = mycursor.fetchall()
    for row in result:
        print(row)

def update_salary():
    eid = int(input("Enter Employee ID: "))
    basic = int(input("Enter New Basic Salary: "))
    allow = int(input("Enter New Allowance: "))
    total = basic + allow
    sql = "UPDATE employee SET basic_salary=%s, allowance=%s, total_salary=%s WHERE emp_id=%s"
    val = (basic, allow, total, eid)
    mycursor.execute(sql, val)
    mydb.commit()
    print("Salary Updated Successfully")

def delete_employee():
    eid = int(input("Enter Employee ID to Delete: "))
    sql = "DELETE FROM employee WHERE emp_id=%s"
    val = (eid,)
    mycursor.execute(sql, val)
    mydb.commit()
    print("Employee Record Deleted Successfully")

while True:
    print("\n1. Add Employee")
    print("2. Display Employees")
    print("3. Update Salary")
    print("4. Delete Employee")
    print("5. Exit")
    ch = int(input("Enter your choice: "))
    if ch == 1:
        add_employee()
    elif ch == 2:
        display_employee()
    elif ch == 3:
        update_salary()
    elif ch == 4:
        delete_employee()
    elif ch == 5:
        print("Thank You")
        break
    else:
        print("Invalid Choice")
```

7. SAMPLE OUTPUT WITH SCREENSHOTS

Output Screen 1: Main Menu

OUTPUT SCREEN 1: MAIN MENU

```
1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: _
```

Output Screen 2: Add Employee Records

OUTPUT SCREEN 2: ADD EMPLOYEE RECORD

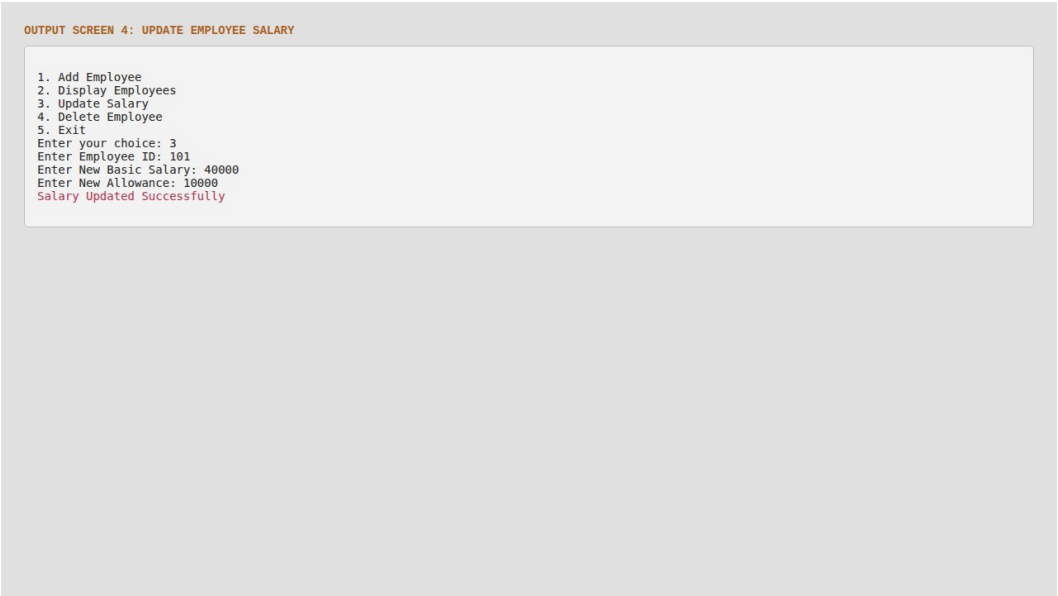
```
1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: 1
Enter Employee ID: 101
Enter Employee Name: Rajesh Kumar
Enter Basic Salary: 35000
Enter Allowance: 8000
Employee Record Added Successfully

1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: 1
Enter Employee ID: 102
Enter Employee Name: Priya Sharma
Enter Basic Salary: 42000
Enter Allowance: 9500
Employee Record Added Successfully

1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: 1
Enter Employee ID: 103
Enter Employee Name: Vikram Singh
Enter Basic Salary: 38000
Enter Allowance: 7500
Employee Record Added Successfully
```

Output Screen 3: Display Employee Records

Output Screen 4: Update Employee Salary



Output Screen 5: Display After Salary Update

OUTPUT SCREEN 5: DISPLAY AFTER SALARY UPDATE

```
1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: 2
(101, 'Rajesh Kumar', 40000, 10000, 50000)
(102, 'Priya Sharma', 42000, 9500, 51500)
(103, 'Vikram Singh', 38000, 7500, 45500)
```

Output Screen 6: Delete Employee Record

OUTPUT SCREEN 6: DELETE EMPLOYEE RECORD

```
1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: 4
Enter Employee ID to Delete: 103
Employee Record Deleted Successfully
```

Output Screen 7: Final Display

OUTPUT SCREEN 7: FINAL DISPLAY

```
1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: 2
(101, 'Rajesh Kumar', 40000, 10000, 50000)
(102, 'Priya Sharma', 42000, 9500, 51500)
```

Output Screen 8: Exit Program

OUTPUT SCREEN 8: EXIT PROGRAM

```
1. Add Employee
2. Display Employees
3. Update Salary
4. Delete Employee
5. Exit
Enter your choice: 5
Thank You
```

8. LIMITATIONS OF THE PROJECT

- No login authentication or user security
- No graphical user interface (console-based only)
- Single-user system - cannot handle concurrent access
- No input validation for data entry
- No backup and recovery mechanism

9. FUTURE SCOPE

- Adding password security and user authentication
- Graphical User Interface (GUI) using Tkinter or PyQt
- Tax and deduction calculation features
- Report generation in PDF/Excel format
- Multi-user support with role-based access
- Cloud database integration for remote access

10. CONCLUSION

The Employee Payroll Management System effectively demonstrates how Python and MySQL can be used together to manage employee records efficiently. This project has provided valuable hands-on experience in:

- Database connectivity using Python
- CRUD operations (Create, Read, Update, Delete)
- Menu-driven application development
- SQL query execution from Python
- Real-world application development

This project enhanced understanding of database connectivity and demonstrates the practical application of programming concepts in solving real-world problems.

11. BIBLIOGRAPHY

6. NCERT Computer Science Class XII Textbook
7. Python Documentation - https://docs.python.org/
8. MySQL Documentation - https://dev.mysql.com/doc/
9. MySQL Connector Python Documentation
10. GeeksforGeeks - Python MySQL Connectivity