python™

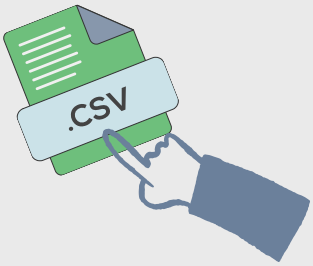
COMPUTER SCIENCE
PROJECT FILE

2025-2026

Presented by:

Name : Rajshree Routh

Class: XII-B



MySQL®

By Rajshree

Certificate

This is to certify that this project titled "Bank Management" using Python and MySQL has been successfully completed by Rajshree Routh . This project has been executed as a part of the Class 12 Computer Science Curriculum and meets the objectives prescribed by CBSE.

Date: _____

Signature: _____

Acknowlegement

I would like to express my sincere gratitude to my computer science teacher Mr. **Biplab Das** for guiding me throughout this project.His valuable suggestions and assistance have helped me complete this project successfully.

Table of contents

Objectives

The objectives of this project is to:

1. Develop a simple Banking System that allows creating accounts, depositing money, withdrawing money, checking balance, viewing mini statements, and listing all accounts.
2. Learn to use Python programming for real-life applications.
3. Learn to connect Python with a MySQL database for storing and retrieving data.
4. Understand basic database operations like INSERT, UPDATE, SELECT and more.

Introduction

Banking systems play an important role in financial management. This project is a simple simulation of a banking system that provides basic operations such as deposit, withdrawal, and balance inquiry. The project demonstrates the use of Python for application logic and MySQL for database management, showing the integration of programming with databases.

Tools Used

(software specifications)

- Python 3.11 (for coding logic and user interface)
- MySQL 8.0 (for backend database)
- mysql-connector-python library (to connect Python with MySQL)

Database Design

Database Name: Bank

Table: accounts — Fields Explanation

Column	Type	Meaning
Account_no	INT(250)	Account Number
name	VARCHAR(250)	Account Holder Name
phone	INT(250)	Phone Number
balance	FLOAT	Current Balance

Table: transactions — Fields Explanation

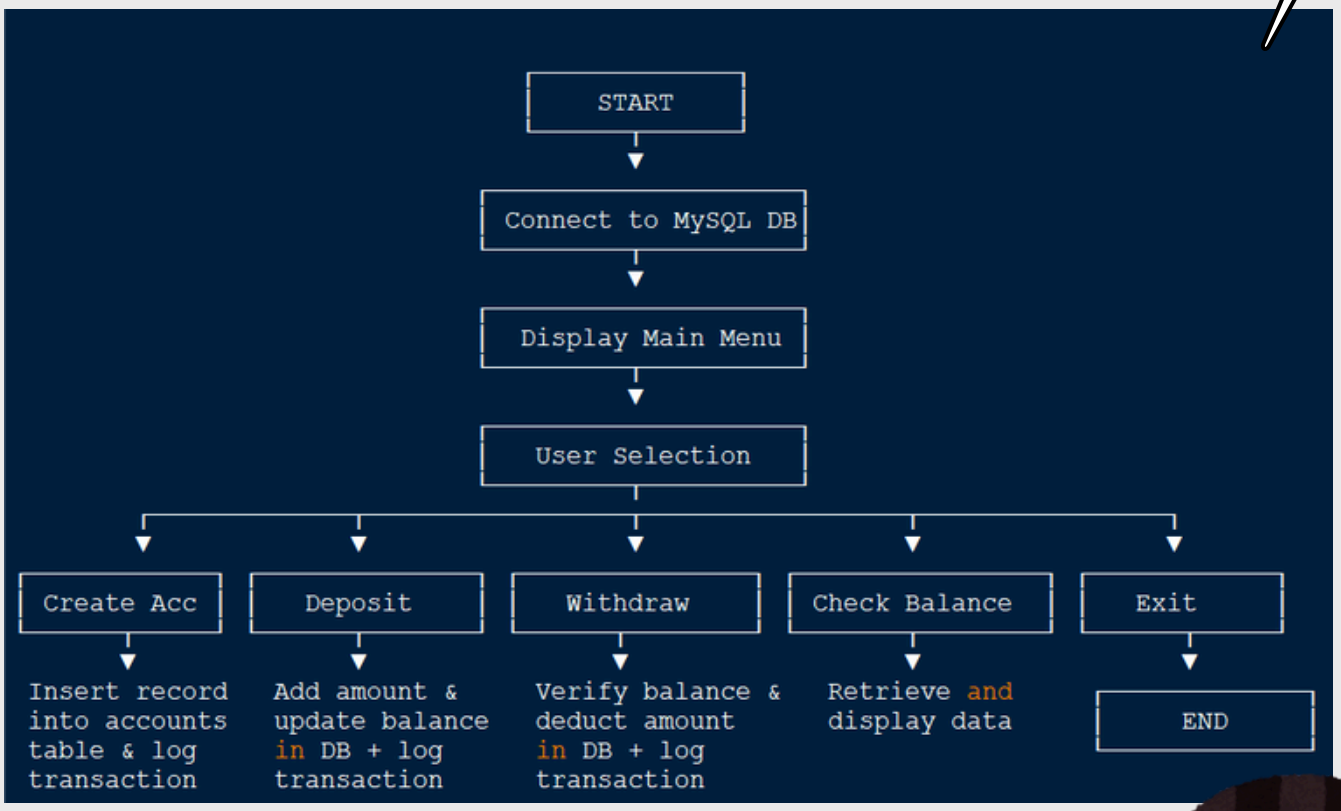
Column	Type	Meaning
id	INT(250)	Auto transaction number
account_no	INT(250)	Account Number
type	VARCHAR(100)	CREATE / DEPOSIT / WITHDRAW
amount	FLOAT	Money deposited / withdrawn
balance	FLOAT	Updated balance after transaction

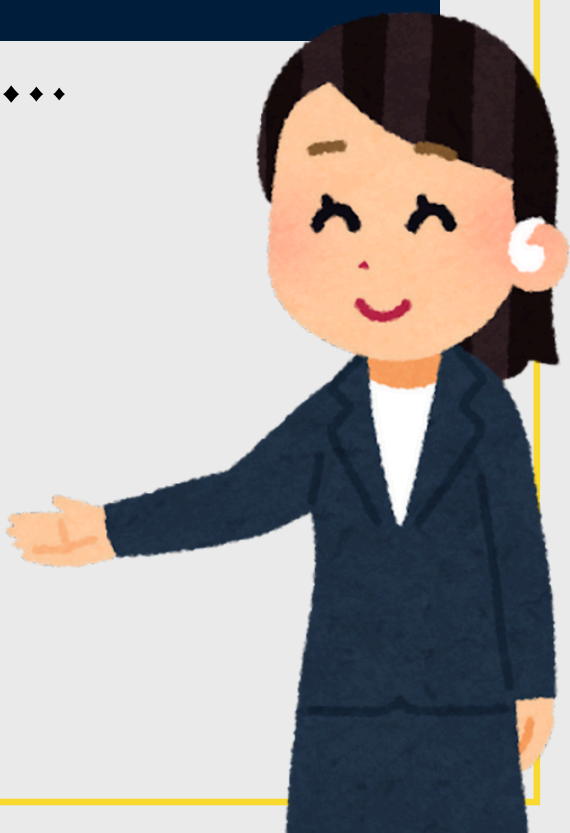
python™

MySQL commands

```
mysql> use bank;
Database changed
mysql> create table accounts(
    -> Account_no       int(250)        primary key,
    -> name     varchar(250),
    -> phone    int(250),
    -> balance  float);
Query OK, 0 rows affected, 2 warnings (0.06 sec)

mysql> create table transactions(
    -> id       int(250)        auto_increment  primary key,
    -> account_no       int(250),
    -> type     varchar(100),
    -> amount   float,
    -> balance  float);
Query OK, 0 rows affected, 2 warnings (0.06 sec)
```

Flowchart


START
Connect to MySQL DB
Display Main Menu
User Selection
Create Acc
Deposit
Withdraw
Check Balance
Exit
Insert record into accounts table & log transaction
Add amount & update balance in DB + log transaction
Verify balance & deduct amount in DB + log transaction
Retrieve and display data
END



THE CODE

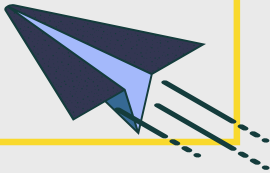
```
import mysql.connector

# ---------- Database Connection ----------
def connect_db():
    return mysql.connector.connect(host="localhost", user="root",
                                   password="12345", database="bank")
# ------------- Banking Functions ------------
# ---------- Funtion to create account ----------
def create_account():
    account_no = int(input("Enter New Account Number: "))
    name = input("Enter Account Holder Name: ")
    phone = int(input("Enter Phone Number: "))
    balance = float(input("Enter Initial Deposit: "))

    con = connect_db()
    cur = con.cursor()
    try:
        cur.execute("""INSERT INTO accounts
                    VALUES ('{}', '{}', '{}', {})"""
                    .format(account_no, name, phone, balance))
        cur.execute("""INSERT INTO transactions
                    (account_no, type, amount, balance)
                    VALUES ('{}', '{}', {}, {})"""
                    .format(account_no, "CREATE", balance, balance))
        con.commit()
        print("\nAccount Created Successfully!")
    except:
        print("\nAccount already exists!")
    con.close()

# ---------- Function to Deposit money ----------
def deposit():
    account_no = int(input("Enter Account Number: "))
    amount = float(input("Enter Amount to Deposit: "))

    con = connect_db()
    cur = con.cursor()
```

```
    cur.execute("SELECT balance FROM accounts WHERE account_no='{}'"
                .format(account_no))
    row = cur.fetchone()

    if not row:
        print("\nAccount not found!")
    else:
        new_balance = row[0] + amount
        cur.execute("""UPDATE accounts SET balance={}
                    WHERE account_no='{}'"""
                    .format(new_balance, account_no))
        cur.execute("""INSERT INTO transactions
                    (account_no, type, amount, balance)
                    VALUES ('{}', '{}', {}, {})"""
                    .format(account_no, "DEPOSIT", amount, new_balance))
        con.commit()
        print("\nDeposit Successful! New Balance:", new_balance)
    con.close()
# ---------- Function to Withdraw money ----------
def withdraw():
    account_no = int(input("Enter Account Number: "))
    amount = float(input("Enter Amount to Withdraw: "))

    con = connect_db()
    cur = con.cursor()

    cur.execute("SELECT balance FROM accounts WHERE account_no='{}'"
                .format(account_no))
    row = cur.fetchone()

    if not row:
        print("\nAccount not found!")
    elif amount > row[0]:
        print("\nInsufficient Balance!")
    else:
        new_balance = row[0] - amount
        cur.execute("""UPDATE accounts SET balance={}
                    WHERE account_no='{}'"""
                    .format(new_balance, account_no))
        cur.execute("""INSERT INTO transactions
                    (account_no, type, amount, balance)
                    VALUES ('{}', '{}', {}, {})"""
                    .format(account_no, "WITHDRAW", amount, new_balance))
        con.commit()
        print("\nWithdrawal Successful! New Balance:", new_balance)
    con.close()
```

```python
# ---------- Function for balance inquiry ----------
def balance_inquiry():
    account_no = int(input("Enter Account Number: "))

    con = connect_db()
    cur = con.cursor()
    cur.execute("""SELECT name, phone, balance
                FROM accounts WHERE account_no='{}'"""
                .format(account_no))
    row = cur.fetchone()

    if row:
        print("\nName: {}\nPhone: {}\nBalance: ₹{}"
              .format(row[0], row[1], row[2]))
    else:
        print("\nAccount not found!")
    con.close()
# ---------- Function for Mini Statements ----------
def mini_statement():
    account_no = int(input("Enter Account Number: "))

    con = connect_db()
    cur = con.cursor()
    cur.execute("""SELECT type, amount, balance
                FROM transactions WHERE account_no='{}'
                ORDER BY id DESC LIMIT 5"""
                .format(account_no))
    rows = cur.fetchall()

    if rows:
        print("\n===== MINI STATEMENT =====")
        for t in rows:
            print(" {} | Amount: ₹{} | Balance: ₹{} "
                  .format(t[0], t[1], t[2]))
    else:
        print("\nNo transactions found!")
    con.close()
# ---------- VIEW ALL ACCOUNTS ----------
def view_accounts():
    con = connect_db()
    cur = con.cursor()
    cur.execute("SELECT * FROM accounts")
    rows = cur.fetchall()

    if rows:
        print("\n===== ALL ACCOUNTS =====")
        for r in rows:
            print("{} | {} | {} | ₹{}"
                  .format(r[0], r[1], r[2], r[3]))
    else:
        print("\nNo accounts found!")
    con.close()
```

```python
# ---------- MAIN MENU ----------
def main():
    while True:
        print("\n===== BANKING SYSTEM MENU=====")
        print("1. Create Account")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. Balance Inquiry")
        print("5. Mini Statement")
        print("6. View All Accounts")
        print("7. Exit")

        choice = input("Enter your choice: ")

        if choice == "1": create_account()
        elif choice == "2": deposit()
        elif choice == "3": withdraw()
        elif choice == "4": balance_inquiry()
        elif choice == "5": mini_statement()
        elif choice == "6": view_accounts()
        elif choice == "7":
            print("\nThank you for using The Banking System")
            break
        else:
            print("\nInvalid choice!")



if __name__ == "__main__":
    main()
```

Waise secretly chat krne k liye yeh achchi jgh hai
no digital footprints



Outputs

```
===== BANKING SYSTEM MENU =====
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Balance Inquiry
5. Mini Statement
6. View All Accounts
7. Exit
Enter your choice: 1
Enter New Account Number: 1001
Enter Account Holder Name: Rajshree
Enter Phone Number: 8882668267
Enter Initial Deposit: 20000
Account Created Successfully!
```

```
===== BANKING SYSTEM MENU =====
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Balance Inquiry
5. Mini Statement
6. View All Accounts
7. Exit
Enter your choice: 2
Enter Account Number: 1001
Enter Amount to Deposit: 5000
Deposit Successful! New Balance: 25000.0
```

```
===== BANKING SYSTEM MENU =====
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Balance Inquiry
5. Mini Statement
6. View All Accounts
7. Exit
Enter your choice: 3
Enter Account Number: 1001
Enter Amount to Withdraw: 50
Withdrawal Successful! New Balance: 24950.0
```

```
===== BANKING SYSTEM MENU =====
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Balance Inquiry
5. Mini Statement
6. View All Accounts
7. Exit
Enter your choice: 4
Enter Account Number: 1001
Account Holder: Rajshree
Phone: 8882668267
Current Balance: 24950.0
```

```
===== BANKING SYSTEM MENU =====
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Balance Inquiry
5. Mini Statement
6. View All Accounts
7. Exit
Enter your choice: 5
Enter Account Number: 1001

===== Mini Statement =====
WITHDRAW | Amount: 50.0 | Balance: 24950.0
DEPOSIT | Amount: 5000.0 | Balance: 25000.0
CREATE | Amount: 20000.0 | Balance: 20000.0
==========================
```

My SQL Tables

Accounts Table

```
mysql> select * from accounts;
+--------+----------+------------+---------+
| acc_no | name     | phone      | balance |
+--------+----------+------------+---------+
| 1001   | Rajshree | 8882668267 |   24950 |
| 1002   | Suraj    | 9990392223 |   50440 |
| 1003   | Emon     | 9873735339 |   40600 |
+--------+----------+------------+---------+
3 rows in set (0.03 sec)
```

Transactions Table

```
mysql> select * from transactions;
+----+--------+----------+--------+---------+
| id | acc_no | type     | amount | balance |
+----+--------+----------+--------+---------+
|  1 | 1001   | CREATE   |  20000 |   20000 |
|  2 | 1001   | DEPOSIT  |   5000 |   25000 |
|  3 | 1001   | WITHDRAW |     50 |   24950 |
|  4 | 1002   | CREATE   |  50000 |   50000 |
|  5 | 1002   | DEPOSIT  |   1000 |   51000 |
|  6 | 1002   | WITHDRAW |    560 |   50440 |
|  7 | 1003   | CREATE   |  45000 |   45000 |
|  8 | 1003   | DEPOSIT  |   1200 |   46200 |
|  9 | 1003   | WITHDRAW |   5600 |   40600 |
+----+--------+----------+--------+---------+
9 rows in set (0.02 sec)
```

Learning Outcomes

- Learned to connect Python programs with MySQL databases.
- Understood SQL operations like INSERT, UPDATE, and SELECT.
- Improved coding and logical thinking through modular programming.
- Gained practical experience in database handling and user input validation.
- Learned to design and document a real-world application.

Future Scope

- A Graphical User Interface (GUI) using Tkinter or PyQt.
- Login authentication for secure access.
- Enable PDF/Excel export for account statements.
- Features like interest calculation and loan tracking.
- Expanding into a web-based banking system using Flask or Django.

Conclusion

This project demonstrates the integration of Python programming with MySQL databases to create a functional banking system. It helped in understanding database operations like insert, update, delete, and select, along with Python's ability to handle user interaction.

Bibliography

Codes made on:

python™ MySQL®

created on :

Canva

Links and sources

https://docs.python.org
https://dev.mysql.com/doc
Computer Science
textbook -- Preeti Arora

Done by :
CBSE Computer Science Project Guidelines