

**INDIAN INSTITUTE OF TECHNOLOGY,**

**KHARAGPUR**

**A Project Report**

**On**

**Bank Management System**

|  |
| --- |
|  |

**By:**

1. Kulal Aditya Gajanan

2. Lokre Digvijay Shivaji

In

Year: 2023-2024



**INDIAN INSTITUTE OF TECHNOLOGY,**

**KHARAGPUR**

###### CERTIFICATE

This is to certify that the project entitled **“Bank Management System”** has been carried out by **Kulal Aditya , Lokre Digvijay** in partial fulfillment of the project i.e. bank Management System.

|  |  |  |
| --- | --- | --- |
| Mr.Abishek Chavle  **Guide** |  | Mr.Pavan Goel  **Computer Eng .Department** |
|  |  |  |

###### ACKNOWLEDGEMENT

We would like to convey our gratitude to **Mr. Ashok Pawar** , Voice Club **Latur** who gave us necessary information and guidance for project.

We are grateful to **Indian Institute Of Technology,Kharagpur** for giving an opportunity to deliver project.

We would like to thank Project In-Charge and Project Guide **Mr.Abishek Chavle** guided us through doing these project development process, provided with invaluable advice, helped us in difficult periods and provided practical assistant for our project. Their willingness to motivate us contributed tremendously to the success of this project.

We would like to express our special thanks of gratitude to our Head of the Department of Computer Engineering **Dr. Pavan Goel** who helped us a lot in finalizing this project.

Besides we would like to thank all staff members who helped us by giving advice and providing equipment which we needed.

Last but not in least we would like to thank all who helped and motivated us.

**With Sincere Thanks,**

**Kulal Aditya Gajanan**

**Lokre Digvijay Shivaji**

**Index**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Topic Name** | **Page No.** |
| **1** | **Abstract** | 5 |
| **2** | **Introduction:** | 6 |
|  | 2.1 Project Overview | 7 |
|  | 2.2 Project Plan (Gantt chart) | 8 |
| **3** | **Project Requirement:** | 8 |
|  | 3.1 Hardware Requirement |  |
|  | 3.2 Software Requirement |  |
| **4** | **System Design:** | 9 |
|  | 4.1 Database Design (Tables Schema) | 9 |
|  | 4.2 E-R Diagram | 10 |
|  | 4.3 Data Flow Diagram (First Level) | 11 |
| **5** | **Designing:** | 13 |
| **6** | **Coding:** | 20 |
| **7** | **Future scope of project:** | 21 |
| **8** | **Conclusion:** | 22 |
| **9** | **Bibliography:** | 22 |
|  | 9.1 Book(s) <Use MLA referencing style> |  |
|  | 9.2 Website(s) <Use Full URL web references> |  |

**Abstract:**

The Bank Management System is a software application that facilitates the efficient management of various banking operations. It provides a centralized platform for banks to handle customer accounts, transactions, and other banking processes. This system aims to streamline operations, improve customer service, and enhance overall efficiency in the banking sector.

The Bank Management System allows customers to create accounts, deposit and withdraw funds, transfer money, and access their account information securely. It provides functionalities for bank administrators to manage customer accounts, track transactions, generate reports, and perform administrative tasks. The system also incorporates security measures to protect customer data and prevent unauthorized access.

Overall, the Bank Management System is a comprehensive software solution that empowers banks to efficiently manage their operations, enhance customer satisfaction, and adapt to the evolving demands of the banking industry.

**Introduction –**

A bank management system is a comprehensive software solution designed to facilitate the efficient and effective management of banking operations and resources. It serves as a centralized platform that integrates various functionalities and processes within a bank, enabling seamless communication, streamlined workflows, and enhanced customer service.

In today's fast-paced and technology-driven banking industry, a robust bank management system is crucial for maintaining a competitive edge and meeting the evolving demands of customers. The system combines advanced technology, data management capabilities, and banking expertise to support the day-to-day operations of a bank while adhering to regulatory requirements and ensuring data security.

The introduction of a bank management system marks a significant shift from traditional manual processes to automated and digitized operations. It replaces cumbersome paper-based tasks with efficient digital workflows, enabling faster transaction processing, accurate record-keeping, and improved customer experiences.

The primary objectives of a bank management system are to optimize operational efficiency, enhance customer service, mitigate risks, and enable informed decision-making. By automating routine tasks, such as account management, fund transfers, and loan processing, the system frees up valuable time for bank employees, allowing them to focus on more complex and value-added activities.

**Project Overview:**

1. **Signup Module:**

This is used to signup into our system. By using this username & password can enter into our project.

1. **Login Module:**

Through this module we enter into our project. By using username and password we login the project.

1. **Create Account Module**

This module create new account of user in a bank. In this module 3 options are include for create new account i.e: customer name , customer new account number & initial balance.

1. **Deposit Module**

User use this module for deposit money in their account. In this module 2 options are include for deposit the amount i.e: account number & how many amount user deposit in their account.

1. **Withdraw Module**

With the help of withdraw module user withdraw amount . In this module 2 options are include for withdraw the amount.

1. **Display Details Module**

This module shows the details of the user. On the basis of correct account number it shows the details of user.

1. **Update Module**

Some time wrong data entered by the employee that time update module is worked.

1. **Close Module**

This module is use for close the account of user , if he/she want to close.

**Project Plan:**

**Gantt chart:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr.no** | **Task Name** | **09-jun** | **11-jun** | **15-jun** | **17-jun** | **19- jun** |
| **1** | **Requirement Gathering** |  |  |  |  |  |
| **2** | **Planning** |  |  |  |  |  |
| **3** | **Designing** |  |  |  |  |  |
| **4** | **Coding** |  |  |  |  |  |
| **5** | **Testing and Deployment** |  |  |  |  |  |

**Project Requirements:**

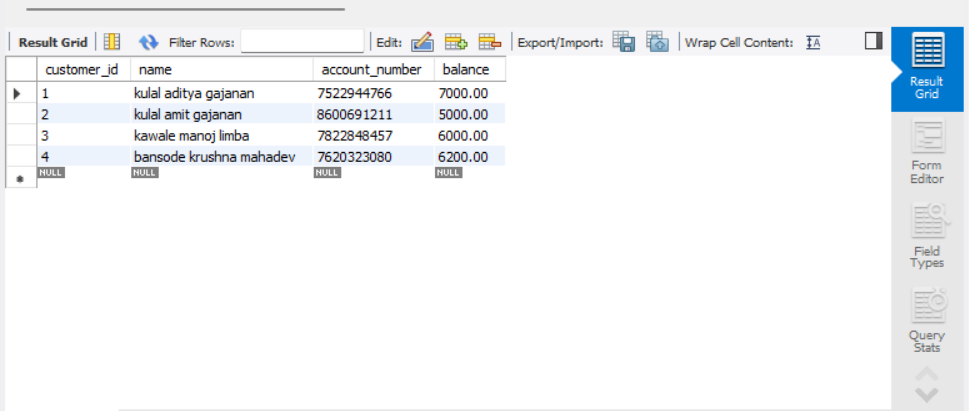
* **Hardware Requirements:**

1. Processor: p4 and above
2. RAM: 512mb and above
3. Hard Disk: 5 GB or above
4. Input device: keyboard, Mouse
5. Output device : Monitor or LCD/LED, Printer

* **Software Requirement:**
* Frontend-IDE: Visual studio 2008
* Language : Python
* Backend-Database: MS SQL Workbench 8.0 CE

**System Design:**

* + - * + **System Design :**

****

**E R Diagram:**

Login

Signup

Bank Facilities

to

User

**Data Flow Diagram (First Level)**

A first-level Data Flow Diagram (DFD) for a Bank Management System typically represents the main processes and data flows within the system. Here's an example of a first-level DFD for a Bank Management System:

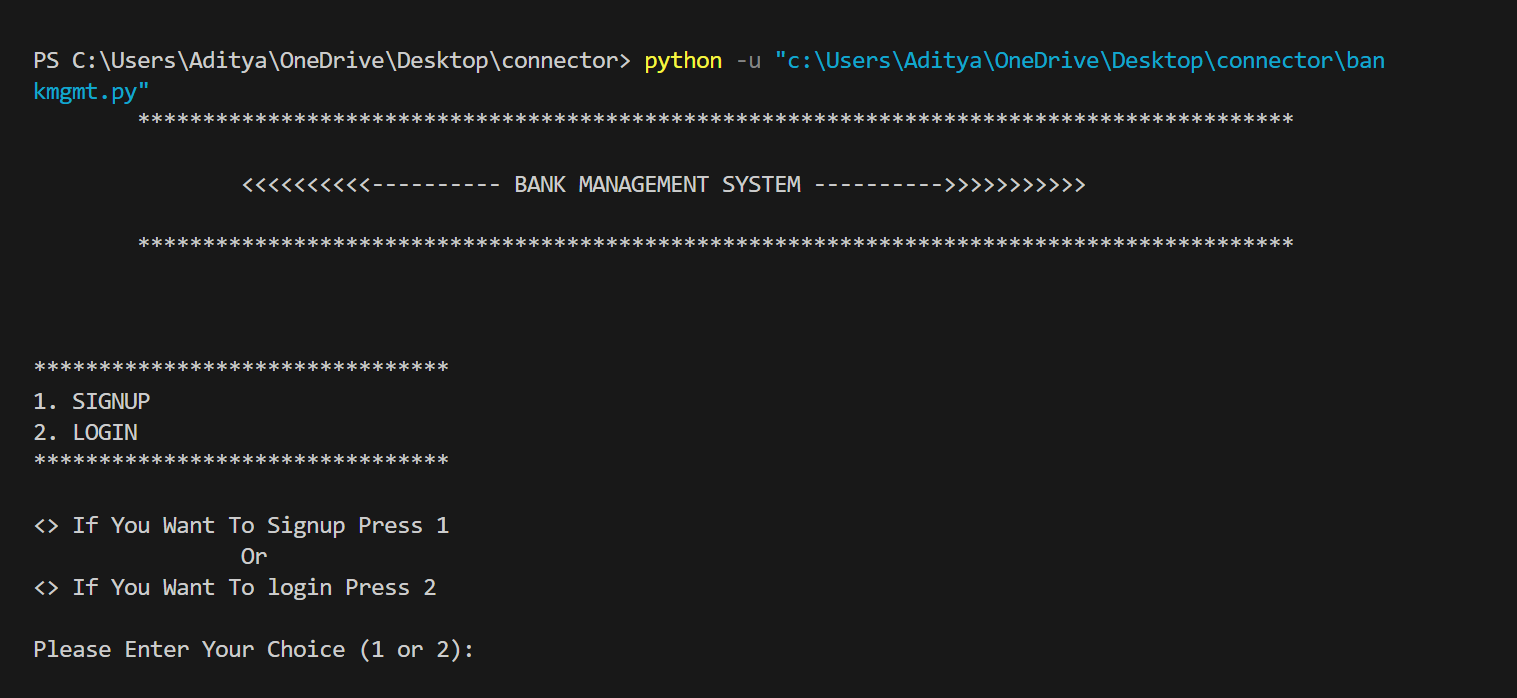
Customer

Report

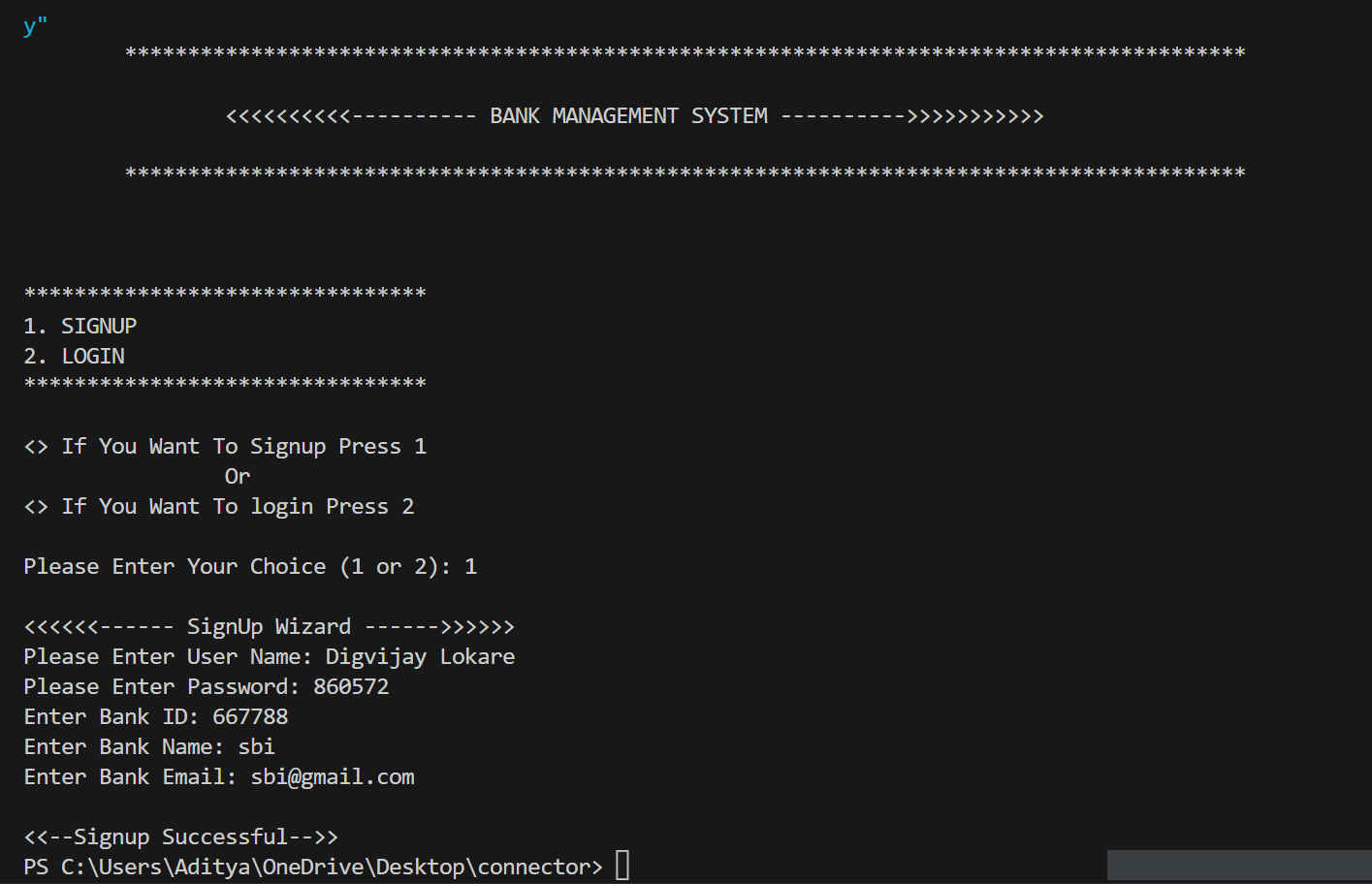
Transaction

Account

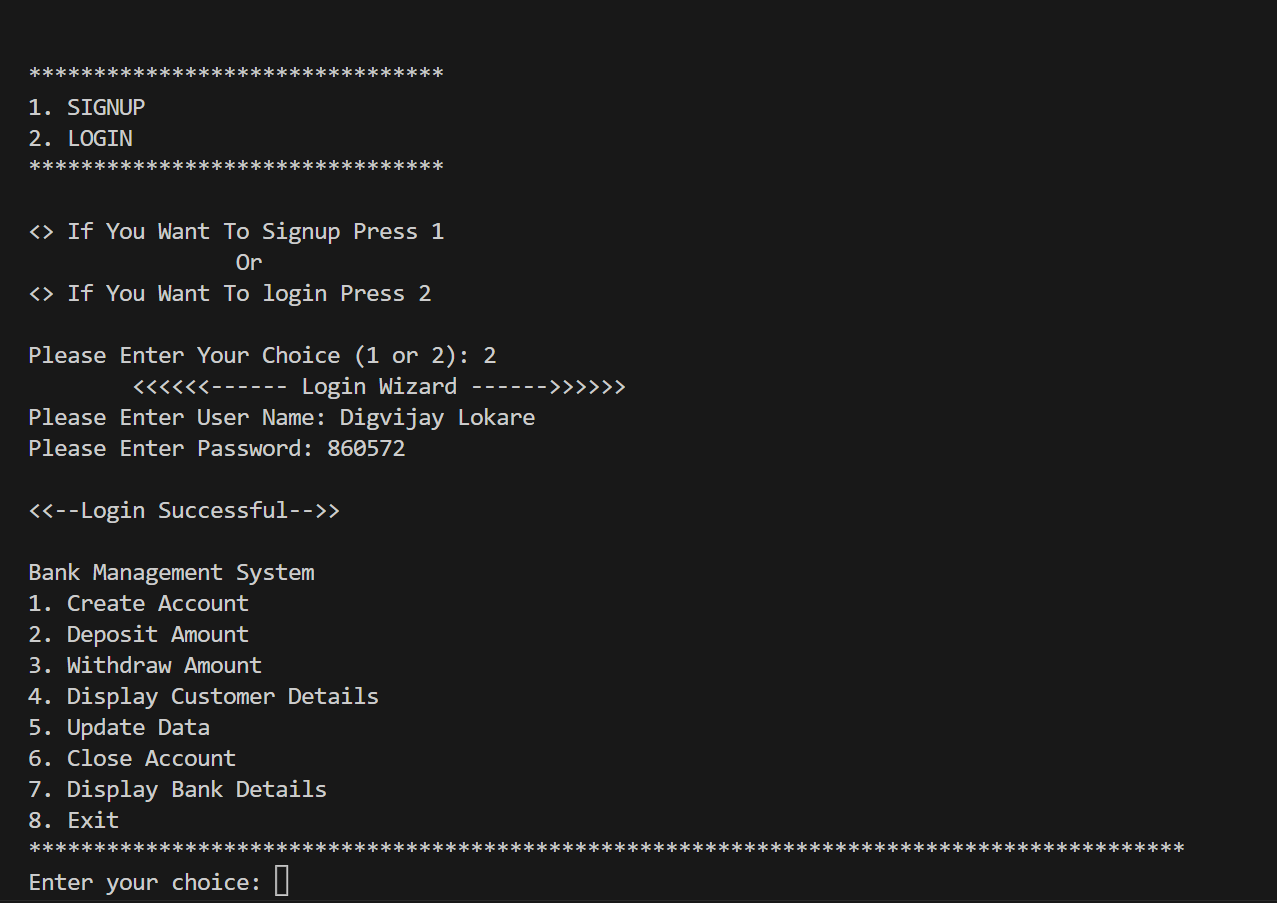
**Designing:**

****

SUGN\_UP page

****

Login page

****

**Coding:**

import mysql.connector

from prettytable import PrettyTable

# Establish database connection

conn = mysql.connector.connect(

host='localhost',

user='root',

password='myfamily@12345',

database='bankmgmt'

)

cursor = conn.cursor()

# Create tables if they don't exist

cursor.execute("""

CREATE TABLE IF NOT EXISTS banks (

bank\_id INT AUTO\_INCREMENT PRIMARY KEY,

bank\_name VARCHAR(100),

bank\_email VARCHAR(100)

)

""")

cursor.execute("""

CREATE TABLE IF NOT EXISTS customers (

customer\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100),

account\_number VARCHAR(20),

balance DECIMAL(10, 2),

bank\_id INT,

FOREIGN KEY (bank\_id) REFERENCES banks(bank\_id)

)

""")

cursor.execute("""

CREATE TABLE IF NOT EXISTS signup (

username VARCHAR(30) PRIMARY KEY,

password VARCHAR(30)

)

""")

cursor.execute("""

CREATE TABLE IF NOT EXISTS customers (

customer\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100),

account\_number VARCHAR(20),

balance DECIMAL(10, 2),

bank\_id INT

)

""")

conn.commit()

# Function to handle user login

def login():

print("\t<<<<<<------ Login Wizard ------>>>>>> ")

username = input("Please Enter User Name: ")

password = input("Please Enter Password: ")

cursor.execute("SELECT username FROM signup WHERE username = %s AND password = %s", (username, password))

user = cursor.fetchone()

if user:

print("\n<<--Login Successful-->>")

return True

else:

print("\n<<--Wrong Password or Username-->>")

return False

# Function to handle user signup

# Function to handle user signup

def signup():

print("\n<<<<<<------ SignUp Wizard ------>>>>>>")

username = input("Please Enter User Name: ")

password = input("Please Enter Password: ")

bank\_id = int(input("Enter Bank ID: "))

bank\_name = input("Enter Bank Name: ")

bank\_email = input("Enter Bank Email: ")

# Insert bank data into the banks table

cursor.execute("""

INSERT INTO banks (bank\_id, bank\_name, bank\_email)

VALUES (%s, %s, %s)

""", (bank\_id, bank\_name, bank\_email))

conn.commit()

# Insert user signup data into the signup table

cursor.execute("INSERT INTO signup (username, password) VALUES (%s, %s)", (username, password))

conn.commit()

print("\n<<--Signup Successful-->>")

# Function to create a new customer account

def create\_account():

name = input("Enter customer name: ")

account\_number = input("Enter account number: ")

balance = float(input("Enter initial balance: "))

bank\_id = int(input("Enter bank ID: "))

# Insert customer data into the database

cursor.execute("""

INSERT INTO customers (name, account\_number, balance, bank\_id)

VALUES (%s, %s, %s, %s)

""", (name, account\_number, balance, bank\_id))

conn.commit()

print("\n<<--Account created successfully-->>")

# Function to update customer data

def update():

cus\_id = int(input("Please enter the customer ID: "))

up\_name = input("Enter customer name you want to change: ")

up\_account\_number = input("Enter account number you want to change: ")

cursor.execute("""

UPDATE customers

SET name = %s, account\_number = %s

WHERE customer\_id = %s

""", (up\_name, up\_account\_number, cus\_id))

conn.commit()

print("<<<<<<------Updation Successful------>>>>>>")

# Function to deposit amount into a customer's account

def deposit():

account\_number = input("Enter account number: ")

amount = float(input("Enter amount to deposit: "))

# Update the customer's balance in the database

cursor.execute("""

UPDATE customers

SET balance = balance + %s

WHERE account\_number = %s

""", (amount, account\_number))

conn.commit()

print("\n<<--Amount deposited successfully-->>")

# Function to withdraw amount from a customer's account

def withdraw():

account\_number = input("Enter account number: ")

amount = float(input("Enter amount to withdraw: "))

# Check if the customer has sufficient balance

cursor.execute("""

SELECT balance

FROM customers

WHERE account\_number = %s

""", (account\_number,))

balance = cursor.fetchone()

if balance:

if balance[0] >= amount:

# Update the customer's balance in the database

cursor.execute("""

UPDATE customers

SET balance = balance - %s

WHERE account\_number = %s

""", (amount, account\_number))

conn.commit()

print("\n<<--Amount withdrawn successfully-->>")

else:

print("\n<<--Insufficient balance-->>")

else:

print("\n<<--Customer not found-->>")

# Function to display customer details

def display\_details():

account\_number = input("Enter account number: ")

# Retrieve customer details from the database

cursor.execute("""

SELECT name, account\_number, balance

FROM customers

WHERE account\_number = %s

""", (account\_number,))

customer = cursor.fetchone()

if customer:

table = PrettyTable(["Name", "Account Number", "Balance"])

table.add\_row(customer)

print("\n" + str(table))

else:

print("\n<<--Customer not found-->>")

# Function to close a customer's account

def close():

account\_number = input("Enter account number: ")

# Check if the customer exists

cursor.execute("""

SELECT name

FROM customers

WHERE account\_number = %s

""", (account\_number,))

customer = cursor.fetchone()

if customer:

# Delete the customer's account from the database

cursor.execute("""

DELETE FROM customers

WHERE account\_number = %s

""", (account\_number,))

conn.commit()

print("\n<<--Account closed successfully-->>")

else:

print("\n<<--Customer not found-->>")

# Function to display bank details

def display\_bank\_details():

cursor.execute("""

SELECT bank\_id, bank\_name, bank\_email

FROM banks

""")

banks = cursor.fetchall()

if banks:

table = PrettyTable(["Bank ID", "Bank Name", "Bank Email"])

for bank in banks:

table.add\_row(bank)

print("\n" + str(table))

else:

print("\n<<--No banks found-->>")

# Bank management system menu

print("\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

print("\t\t<<<<<<<<<<---------- BANK MANAGEMENT SYSTEM ---------->>>>>>>>>>>\n")

print("\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\n")

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("1. SIGNUP")

print("2. LOGIN")

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

choice = input("\n<> If You Want To Signup Press 1\n\t\tOr\n<> If You Want To login Press 2\n\nPlease Enter Your Choice (1 or 2): ")

if choice == "1":

signup()

elif choice == "2":

logged\_in = login()

if logged\_in:

while True:

print("\nBank Management System")

print("1. Create Account")

print("2. Deposit Amount")

print("3. Withdraw Amount")

print("4. Display Customer Details")

print("5. Update Data")

print("6. Close Account")

print("7. Display Bank Details")

print("8. Exit")

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

choice = input("Enter your choice: ")

if choice == "1":

create\_account()

elif choice == "2":

deposit()

elif choice == "3":

withdraw()

elif choice == "4":

display\_details()

elif choice == "5":

update()

elif choice == "6":

close()

elif choice == "7":

display\_bank\_details()

elif choice == "8":

break

else:

print("\nInvalid choice!")

else:

exit()

else:

print("\nInvalid choice!")

exit()

# Close database connection

conn.close()

**Future scope of project:**

The Bank Management System is a foundational software application for managing banking operations. Here are some potential future scope areas for further development and enhancement of the project:

1. Enhanced Security Features: Strengthen the security measures to protect customer data and prevent unauthorized access. Implement multi-factor authentication, encryption, and intrusion detection systems to ensure the confidentiality and integrity of sensitive information.

2. Integration with Payment Gateways: Enable seamless integration with external payment gateways to facilitate online transactions and digital payments. This can provide customers with convenient options for fund transfers, bill payments, and online purchases.

3. Mobile Banking and Internet Banking: Develop mobile applications and web interfaces to offer customers the flexibility of accessing their accounts and performing transactions through mobile devices and web browsers. Ensure a user-friendly interface and optimize the system for mobile responsiveness.

* **Conclusion:**

The bank management system code provided is a basic implementation that allows users to sign up, log in, create customer accounts, deposit and withdraw funds, update customer data, display customer details, close customer accounts, and display bank details.

The code utilizes MySQL database for data storage and retrieval, and it establishes a connection to the database using the mysql.connector library. It includes error handling for incorrect login credentials, insufficient funds during withdrawals, and customer/account not found scenarios.

* **Bibliography –**

* Books:

1. "Bank Management and Financial Services" by Peter S. Rose and Sylvia C. Hudgins

* Website:
  + 1. MySQL Documentation: <https://dev.mysql.com/doc/>
    2. GeeksforGeeks: https://www.geeksforgeeks.org/banking-system-2/