```
# importing libraries
import os
from datetime import datetime
from getpass import getpass

import mysql.connector
from mysql.connector import Error
from playsound import playsound
from tabulate import tabulate
from twilio.rest import Client
from dotenv import load_dotenv

load_dotenv(".env")
USER_NAME = os.getenv("USER_NAME")
PASSWORD = os.getenv("PASSWORD")
HOSTNAME = os.getenv("HOSTNAME")
DB_NAME = os.getenv("DATABASE_NAME")

# creating database
def create_db(user_name, user_pass,host_name,db_name):
    try:
        mydb = mysql.connector.connect(
```

```
        host=host_name,
        user=user_name,
        passwd=user_pass,
      )
      mycursor = mydb.cursor()
      query_create_db = "CREATE DATABASE IF NOT EXISTS " + db_name
      mycursor.execute(query_create_db)

      print("\n***Database is successfully created***\n")

   except Error as e:
      print(f"\nERROR : {e} occurred !\n")



# 3 tables :-
# customers -> Stores info about customers
# transactions -> Stores info about any credited/debited transactions
# auth -> Stores customer's password
```

```
# query to create customers table
query_create_table_customers = '''CREATE TABLE IF NOT EXISTS customers(
    acc_num int AUTO_INCREMENT,
    f_name varchar(20) NOT NULL,
    l_name varchar(20) NOT NULL,
    aadhar_num varchar(20) NOT NULL,
    dob varchar(10) NOT NULL,
    city varchar(20) NOT NULL,
    area varchar(20) NOT NULL,
    pincode varchar(12) NOT NULL,
    phone_num varchar(12) NOT NULL,
    email_id varchar(30) NOT NULL,
    account_type varchar(10) NOT NULL,
    sms_banking varchar(2),
    current_amount float NOT NULL,
    PRIMARY KEY(acc_num)
);'''
```

```
# query to create transactions table
query_create_table_transactions = '''
CREATE TABLE IF NOT EXISTS transactions(
   trans_id int AUTO_INCREMENT,
   acc_num int,
   amount float,
   type varchar(10),
   date varchar(20),
   PRIMARY KEY(trans_id),
   FOREIGN KEY (acc_num) REFERENCES customers(acc_num)
);'''

# query to create auth table
query_create_table_auth = '''
CREATE TABLE IF NOT EXISTS auth(
   acc_num int,
   password varchar(100) NOT NULL,
   FOREIGN KEY (acc_num) REFERENCES customers(acc_num)
);'''
```

```
# admin credential
admin_id = 0000
admin_passwd = 'root'



# function to make a connection to MySql server
# and connect to database
def create_connection(user_name, user_pass, host_name,db_name):
    connection = None
    try:
        connection = mysql.connector.connect(
            host=host_name,
            user=user_name,
            passwd=user_pass,
            database=db_name
        )
        print("\n***Connection to MySQL database is successfull***\n")

    except Error as e:
        print(f"\nERROR : {e} occurred !\n")

    return connection
```

```
# function to execute SQL query
def execute_query(connection, query):
    cursor = connection.cursor()

    try:
        cursor.execute(query)
        connection.commit()
        return 1

    except Error as e:
        print(f"\nERROR : {e} occurred !\n")
        return 0
```

```
# function to print table
def read_table(connection, table_name="", query=""):
    cursor = connection.cursor()
    result = None
    if query == "":
        read_table_query = "SELECT * FROM " + table_name
    else:
        read_table_query = query


    try:
        cursor.execute(read_table_query)
        result = cursor.fetchall()
        column_names = [description[0] for description in
cursor.description]
        print(tabulate(result, headers=column_names, tablefmt='psql'))
        # return column_names,result


    except Error as e:
        print(f"\nERROR : {e} occurred !\n")
```

```
# function to authenticate admin/customer
def authenticate(connection, level):

    usr_id = int(input("Enter your id : "))
    pass_word = getpass("Enter your password : ")

    if level == 'admin':
        if usr_id == admin_id and pass_word == admin_passwd:
            return True

    elif level == 'customer':
        p = getDetail(connection, "auth",("acc_num", usr_id, 'int'),
"password")
        if pass_word == p:
            return True
    return False
```

```
# function to handle operation available for admin
def main_menu_admin(connection):

    auth = authenticate(connection, "admin")
    if auth == True:
        while True:
            print('\n######## Main Menu #######')
            print("\n1.  Open New Account")
            print('\n2.  Update existing Account')
            print('\n3.  Close existing Account')
            print('\n4.  See all Customers details')
            print('\n5.  See all Transactions details')
            print('\n6.  Exit')

            print('\n\n')

            choice = int(input('Enter your choice :-  '))

            if choice == 1:
                add_new_account(connection)
            elif choice == 2:
                update_account(connection)
```

```
        elif choice == 3:
            close_account(connection)
        elif choice == 4:
            read_table(connection, "customers")
        elif choice == 5:
            read_table(connection, "transactions")
        elif choice == 6:
            break
        else:
            print("\nInvalid Input !\n")
else:
    playsound("Sounds/error_sound.mp3")
    print("\nWrong credentials !\n")
```

```
# function to handle operation available for customer
def main_menu_customer(connection):

    auth = authenticate(connection, "customer")
    if auth == True:
        while True:
            print('\n######## Main Menu ########')
            print('\n1.  Transaction Menu')
            print('\n2.  Report Menu')
            print('\n3.  Exit')
            print('\n\n')

            choice = int(input())

            if choice == 1:
                transaction_menu(connection)
            elif choice == 2:
                report_menu()
            elif choice == 3:
                break
            else:
                print("\nInvalid Input !\n")
```

```
else:
    playsound("Sounds/error_sound.mp3")
    print("\nWrong credentials !\n")
```

```
# function to add/create new account
def add_new_account(connection):
    os.system('cls' if os.name == 'nt' else 'clear')
    f_name = input("Enter your first name : ")
    l_name = input("Enter your last name : ")
    aadhar_num = input("Enter your Aadhar number : ")
    dob = input("Enter your date of birth (Format ->  DD/MM/YYYY): ")
    addr_city = input("Enter your city name :")
    addr_area = input("Enter your area name :")
    addr_pincode = input("Enter your pin code : ")
    phone_num = input("Enter your phone number : ")
    email_id = input("Enter your email-id : ")
    account_type = input("Enter your account type [Current/Saving] : ")
    sms_banking = input("Do you want to activate SMS Banking service ?
[Y/N] ")
    current_amount = float(input("Enter opening amount : "))
    password = getpass("Enter your password : ")


    query_insert_new_account = '''INSERT INTO customers
(f_name,l_name,aadhar_num,dob,city,area,


pincode,phone_num,email_id,account_type,sms_banking,current_amo
unt) VALUES ('''' + \
```

```
        """+f_name+""" + "," + """+l_name+""" + "," +
"""+str(aadhar_num)+""" + "," + \
        """+dob+""" + "," + """+addr_city+""" + "," + """+addr_area+""" + "," +
"""+addr_pincode+ \
        """ + "," + """+phone_num+""" + "," + """+email_id+""" + "," +
"""+account_type+""" + \
        "," + """+sms_banking+""" + "," + str(current_amount) + ");"
    status1 = execute_query(connection, query_insert_new_account)


    acc_n = getDetail(connection, "customers",("aadhar_num",
aadhar_num, 'str'), "acc_num")
    query_insert_auth = "INSERT INTO auth (acc_num, password)
VALUES (" + str(acc_n) + \
                        "," + """ + str(password) + """ + ");"
    status2 = execute_query(connection, query_insert_auth)


    if status1 and status2 == 1:
        print("\n***Record inserted successfully...***\n")
```

```
# function to update existing account details
def update_account(connection):
    os.system('cls' if os.name == 'nt' else 'clear')
    acc_num = int(input("Enter the account number whose details will be updated : "))
    print("\nWhat do you want to update? ")
    print("\n1. First Name")
    print("\n2. Last Name")
    print("\n3. Date of Birth")
    print("\n4. City")
    print("\n5. Area")
    print("\n6. Pincode")
    print("\n7. Contact Number")
    print("\n8. Email-ID")
    print("\n9. Account Type")
    print("\n10. SMS-Banking")
    print("\n11. Password")
    print("\n12. Aadhar Number")

    choice = int(input())
    if choice == 1:
        place = 'f_name'
```

```
    new_data = input("Enter your new first name : ")
elif choice == 2:
    place = 'last_name'
    new_data = input("Enter your new last name : ")
if choice == 3:
    place = 'dob'
    new_data = input(
        "Enter your new date of birth (Format ->  DD/MM/YYYY): ")
elif choice == 4:
    place = 'city'
    new_data = input("Enter your new city name :")
elif choice == 5:
    place = 'area'
    new_data = input("Enter your new area name :")
elif choice == 6:
    place = 'pincode'
    new_data = int(input("Enter your new pin code : "))
elif choice == 7:
    place = 'phone_num'
    new_data = input("Enter your new phone number : ")
elif choice == 8:
    place = 'email_id'
```

```
        new_data = input("Enter your new email-id : ")
    elif choice == 9:
        place = 'account_type'
        new_data = input("Enter your new account type [Current/Saving] :
")
    elif choice == 10:
        place = 'sms_banking'
        new_data = input("Do you want to activate SMS Banking service ?
[Y/N] ")
    elif choice == 11:
        place = 'password'
        new_data = input("Enter your new password : ")
    elif choice == 12:
        place = "aadhar_num"
        new_data = input("Enter your correct aadhar number : ")



    else:
        print("\nInvalid Input !\n")



    if place == 'password':
```

```
        query_update_existing_account = "UPDATE auth SET " + place + " =
" + \
        """ + str(new_data) + """ + " WHERE  acc_num = " + str(acc_num) +
';'
    else:
        query_update_existing_account = "UPDATE customers SET " +
place + " = " + \
        """ + str(new_data) + """ + " WHERE  acc_num = " + str(acc_num) +
';'


    status = execute_query(connection, query_update_existing_account)
    if status == 1:
        sms = getDetail(connection, "customers",
                        ("acc_num", acc_num, 'int'), "sms_banking")
        if sms == 'Y':
            try:
                sendSMS(connection, acc_num, " updated " + " for "+place,0)
            except:
                print("\nCan't send sms report for this transaction. Kindly
check your internet connection or mobile number !\n")
        print("\n***Record updated successfully...***\n")
```

```
# function to close account
def close_account(connection):
    acc_num = int(input("Enter your account number : "))

    query_delete_transactiondata = "DELETE FROM transactions WHERE acc_num = " + \
        str(acc_num) + ';'
    query_delete_authdata = "DELETE FROM auth WHERE acc_num = " + \
        str(acc_num) + ';'
    query_delete_accountdata = "DELETE FROM customers WHERE acc_num = " + \
        str(acc_num) + ';'
    # add_to_deleted_tables(acc_num)

    status = execute_query(connection, query_delete_authdata) and execute_query(connection, query_delete_transactiondata) and execute_query(connection, query_delete_accountdata)
    if status == 1:
        print("\n***Record deleted successfully...***\n")
        print("\nSad to see you go, Come back soon!\n")
```

```
# functions to handle transactions
def sendSMS(connection, acc_num, cat, amount=0):
    account_sid = 'YOUR_ACCOUNT_SID'
    auth_token = 'YOUR_AUTH_TOKEN'
    to_num = '+91'+getDetail(connection, "customers",
                    ("acc_num", acc_num, 'int'), "phone_num")
    client = Client(account_sid, auth_token)
    f_name = getDetail(connection, "customers",
                    ("acc_num", acc_num, 'int'), "f_name")
    l_name = getDetail(connection, "customers",
                    ("acc_num", acc_num, 'int'), "l_name")
    message = client.messages.create(
        from_='<YOUR_TWILIO_MOBILE_NUMBER>',
        body='Dear ' + str(f_name) + " " + str(l_name) + ', your account
number ' +
        str(acc_num) + " is " + str(cat) + str(amount) + ".",
        to= to_num
    )
```

```
# function to tell if customer's account exists
def doesAccountExist(connection, acc_num):
    query = 'SELECT * FROM customers WHERE acc_num = ' +
str(acc_num) + ';'


    cursor = connection.cursor()
    try:
        cursor.execute(query)
        records = cursor.fetchall()
        n = len(records)
        if(n <= 0):
            return 0
        else:
            return 1
    except Error as e:
        print(f"\n\nERROR : {e} occurred !\n\n")
```

```
#function to get given detail from given table according to given condition
def getDetail(connection, table_name, cond, detail):  # ex- cond = ("acc_num",1,'int')
    if cond[2] == 'str':
        query = 'SELECT ' + detail + ' FROM ' + table_name + ' WHERE ' + \
            str(cond[0]) + '=' + "'" + str(cond[1]) + "'" + ';'

    elif cond[2] == 'int':
        query = 'SELECT ' + detail + ' FROM ' + table_name + \
            ' WHERE ' + str(cond[0]) + '=' + str(cond[1]) + ';'

    cursor = connection.cursor()
    try:
        cursor.execute(query)
        data = cursor.fetchall()
        data = data[0][0]
    except Error as e:
        print(f"\n\nERROR : {e} occurred !\n\n")
    return data
```

```
# function to credit/deposit money
def deposit_money(connection, acc_num):
    amount = float(input("Enter amount to deposit : "))
    today = datetime.now()
    query_deposit_money = "UPDATE customers SET current_amount = current_amount + " + \
        str(amount) + " WHERE acc_num = " + str(acc_num) + ";"

    query_insert_transactions = "INSERT INTO transactions(acc_num,amount,type,date) VALUES ( " + \
        str(acc_num) + "," + str(amount) + "," + "'" + "Credited" + "'" + "," "'" + \
        str(today)[:19] + "'" + ");"

    status = execute_query(connection, query_deposit_money) and execute_query(connection,
                                                query_insert_transactions)

    if status == 1:
        sms = getDetail(connection, "customers",
                    ("acc_num", acc_num, 'int'), "sms_banking")
        if sms == 'Y':
            try:
```

```
            sendSMS(connection, acc_num, " credited with ", amount)
        except:
            print("\nCan't send sms report for this transaction. Kindly check your internet connection or mobile number!\n")
        print("\n***Amount deposited successfully...***\n")
```

```
#function to withdraw money
def withdraw_money(connection, acc_num):
    amount = float(input("Enter amount to withdraw : "))
    curr_amnt = getDetail(connection, "customers",
                          ("acc_num", acc_num, 'int'), "current_amount")
    if amount > curr_amnt:
        print("\nYou don't have sufficient balance!\n")
    else:
        today = datetime.now()
        query_withdraw_money = "UPDATE customers SET
current_amount  =   current_amount - " + \
            str(amount) + " WHERE acc_num = " + str(acc_num) + ";"


        query_insert_transactions = "INSERT INTO
transactions(acc_num,amount,type,date) VALUES ( " + \
            str(acc_num) + "," + str(amount) + "," + "'" + "Debited" + "'" + ","
+ "'" + \
            str(today)[:19] + "'" + ");"


        status = execute_query(connection, query_withdraw_money) and
execute_query(connection,
                                                query_insert_transactions)
        if status == 1:
```

```
        playsound("Sounds/withdraw_sound.mp3")
        sms = getDetail(connection, "customers",
                    ("acc_num", acc_num, 'int'), "sms_banking")
        if sms == 'Y':
            try:
                sendSMS(connection, acc_num, " debited by ", amount)
            except:
                print(
                    "\nCan't send sms report for this transaction. Kindly check
your internet connection or mobile number!\n")
        print("\n***Amount Withdrawn successfully...***\n")
```

```
#function for transaction menu
def transaction_menu(connection):
    os.system('cls' if os.name == 'nt' else 'clear')
    acc_num = int(input("Enter your account number : "))

    while True:
        print('\n######## Transaction Menu ########')
        print("\n1.  Deposit")
        print('\n2.  WithDraw')
        print("\n3.  Balance Enquiry")
        print('\n4.  Back to Main Menu')
        print('\n\n')

        choice = int(input())

        if choice == 1:
            if doesAccountExist(connection, acc_num) == 1:
                deposit_money(connection, acc_num)
            else:
                print(acc_num, "doesn't exists !")

        elif choice == 2:
```

```
            if doesAccountExist(connection, acc_num) == 1:
                withdraw_money(connection, acc_num)
            else:
                print(acc_num, "doesn't exists !")

        elif choice == 3:
            print("Your current balance is : ₹", getDetail(connection,
                  "customers", ("acc_num", acc_num, 'int'),
"current_amount"))

        elif choice == 4:
            break
```

```
#function to make a report menu
def report_menu():
    # coming soon...
    pass



# driver code


# creating database and initiating connection
host_name = input("Enter hostname : ") or HOSTNAME
user_name = input("Enter username : ") or USER_NAME
user_pass = getpass("Enter user password : ") or PASSWORD
db_name = input("Enter the name of database : ") or DB_NAME
print(user_pass, user_name, host_name, db_name)
status = create_db(user_name, user_pass,host_name,db_name)
if status == 1:
    print("\nDatabase is ready.\n\n")



connection =
create_connection(user_name,user_pass,host_name,db_name)
```

```
# creating required tables
status = execute_query(connection, query_create_table_customers) and execute_query(connection,
        query_create_table_transactions) and execute_query(connection, query_create_table_auth)

if status == 1:
    print("\nTables are initalized. You are ready to go...\n\n")

print("\n######## Login Panel ########\n\n")
print("1. Admin Login\n2. Customer Login\n")
choice = int(input())

if choice == 1:
    main_menu_admin(connection)
elif choice == 2:
    main_menu_customer(connection)
else:
    print("\nInvalid Input !\n")
```