**Online Banking System**

Manvith.S.Rao-200953242

Vignesh.Yogesh-200953245

**Class Diagram :**

**Diagram

Description automatically generated**

**Source Code :**

Bank :

package Bank;

import java.io.Serializable;

import javax.swing.DefaultListModel;

import Exceptions.AccNotFound;

import Exceptions.InvalidAmount;

import Exceptions.MaxBalance;

import Exceptions.MaxWithdraw;

public class Bank implements Serializable {

    /\*\*

     \*

     \*/

    private static final long serialVersionUID = 1L;

    private BankAccount[] accounts= new BankAccount[100];

    public int addAccount(BankAccount acc)

    {

        int i=0;

        for(i=0;i<100;i++)

        {

            if(getAccounts()[i]==null)

            {

                break;

            }

        }

        getAccounts()[i]=acc;

        return i;

    }

    public int addAccount(String name, double balance, double maxWithLimit )

    {

        SavingsAccount acc=new SavingsAccount(name, balance, maxWithLimit);

        return this.addAccount(acc);

    }

    public int addAccount(String name, double balance, String tradeLicense) throws Exception

    {

        CurrentAccount acc = new CurrentAccount(name, balance,tradeLicense);

        return this.addAccount(acc);

    }

    public int addAccount(String name, String  institutionName, double balance, double min\_balance)

    {

        StudentAccount acc= new StudentAccount(name,balance,institutionName);

        return this.addAccount(acc);

    }

    public BankAccount findAccount(String aacountNum)

    {

        int i;

        for(i=0;i<100;i++)

        {

            if(getAccounts()[i]==null)

            {

                break;

            }

            if(getAccounts()[i].acc\_num.equals(aacountNum))

            {

                return getAccounts()[i];

            }

        }

        return null;

    }

    public void deposit(String aacountNum, double amt) throws InvalidAmount,AccNotFound

    {

        if(amt<0)

        {

            throw new InvalidAmount("Invalid Deposit amount");

        }

        BankAccount temp=findAccount(aacountNum);

        if(temp==null)

        {

            throw new AccNotFound("Account Not Found");

        }

        if(temp!=null)

        {

            temp.deposit(amt);

        }

    }

    public void withdraw(String aacountNum, double amt) throws MaxBalance,AccNotFound, MaxWithdraw, InvalidAmount

    {

        BankAccount temp=findAccount(aacountNum);

        if(temp==null)

        {

            throw new AccNotFound("Account Not Found");

        }

        if(amt<=0)

        {

            throw new InvalidAmount("Invalid Amount");

        }

        if(amt>temp.getbalance())

        {

            throw new MaxBalance("Insufficient Balance");

        }

        if(temp!=null)

        {

            temp.withdraw(amt);

        }

    }

    public DefaultListModel<String> display()

    {

        DefaultListModel<String> list=new DefaultListModel<String>();

        int i;

//      String type=null;

        for(i=0;i<100;i++)

        {

            if(getAccounts()[i]==null)

            {

                break;

            }

//          if(getAccounts()[i].getClass().toString().equals("class Bank.SavingsAccount"))

//          {

//              type="Type: Savings Account";

//          }

//

//          else if(getAccounts()[i].getClass().toString().equals("class Bank.CurrentAccount"))

//          {

//              type="Type: Current Account";

//          }

//

//          else if(getAccounts()[i].getClass().toString().equals("class Bank.StudentAccount"))

//          {

//              type="Type: Student Account";

//          }

            list.addElement(getAccounts()[i].toString());

        }

        return list;

    }

    public BankAccount[] getAccounts() {

        return accounts;

    }

    public void setAccounts(BankAccount[] accounts) {

        this.accounts = accounts;

    }

}

Data :

package Data;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import Bank.\*;

public class FileIO {

public static Bank bank=null;

    public static void Read()

    {

//      Bank bank =null;

        FileInputStream fis =null;

        ObjectInputStream oin=null;

        try {

            fis =new FileInputStream("data");

            oin=new ObjectInputStream(fis);

            FileIO.bank=(Bank)oin.readObject();

            }

        catch (Exception en) {

            FileIO.bank=new Bank();

                }

        finally{

            try{

                if(oin!=null) oin.close();

            if(fis!=null) fis.close();

            }

            catch (IOException en) {

                    }

        }

        //return bank;

    }

    public static void Write()

    {

        try {

            FileOutputStream fout=new FileOutputStream("data");

            ObjectOutputStream out=new ObjectOutputStream(fout);

            out.writeObject(FileIO.bank);

            out.flush();

            fout.close();

            }

            catch(Exception en)

            {

            }

    }

}

Account not found :

package Exceptions;

public class AccNotFound extends Exception {

    /\*\*

     \*

     \*/

    private static final long serialVersionUID = 1L;

    public AccNotFound(String s)

    {

        super(s);

    }

}

Login :

package GUI;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

import java.awt.Font;

import javax.swing.JTextField;

import javax.swing.JButton;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

public class Login {

    public JFrame frame;

    private JTextField textField;

    private JPasswordField textField\_1;

    /\*\*

     \* Launch the application.

     \* @throws FileNotFoundException

//   \*/

//  public static void main(String[] args)

//  {

//      EventQueue.invokeLater(new Runnable() {

//          public void run() {

//              try {

//                  Login window = new Login();

//                  window.frame.setVisible(true);

//              } catch (Exception e) {

//                  e.printStackTrace();

//              }

//          }

//      });

//  }

    /\*\*

     \* Create the application.

     \*/

    public Login() {

        initialize();

    }

    /\*\*

     \* Initialize the contents of the frame.

     \*/

    private void initialize() {

        frame = new JFrame();

        frame.setBounds(100, 100, 450, 300);

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        frame.setTitle("Banking System");

        frame.getContentPane().setLayout(null);

        JLabel label = new JLabel("Banking System");

        label.setFont(new Font("Tahoma", Font.BOLD, 17));

        label.setBounds(147, 11, 151, 41);

        frame.getContentPane().add(label);

        JLabel lblLoginScreen = new JLabel("Login Screen");

        lblLoginScreen.setFont(new Font("Tahoma", Font.PLAIN, 13));

        lblLoginScreen.setBounds(170, 63, 101, 23);

        frame.getContentPane().add(lblLoginScreen);

        JLabel lblUsername = new JLabel("Username:");

        lblUsername.setFont(new Font("Tahoma", Font.PLAIN, 12));

        lblUsername.setBounds(55, 119, 64, 23);

        frame.getContentPane().add(lblUsername);

        JLabel lblPassword = new JLabel("Password:");

        lblPassword.setFont(new Font("Tahoma", Font.PLAIN, 12));

        lblPassword.setBounds(55, 159, 64, 23);

        frame.getContentPane().add(lblPassword);

        textField = new JTextField();

        textField.setBounds(130, 121, 86, 20);

        frame.getContentPane().add(textField);

        textField.setColumns(10);

        textField.setText("admin");

        textField\_1 = new JPasswordField();

        textField\_1.setBounds(130, 161, 86, 20);

        frame.getContentPane().add(textField\_1);

        textField\_1.setColumns(10);

        JButton btnLogin = new JButton("Login");

        btnLogin.addActionListener(new ActionListener() {

            @SuppressWarnings("deprecation")

            public void actionPerformed(ActionEvent e) {

                String user,pass;

                textField.setText("admin");

                user="admin";

                //user=textField.getText();

                pass=textField\_1.getText();

                if((user.equals("admin")&&(pass.equals("admin"))))

                        {

                            JOptionPane.showMessageDialog(frame.getComponent(0), "Login Successfully");

                            frame.setVisible(false);

                            GUIForm.menu.setVisible(true);

                        }

                else

                {

                    JOptionPane.showMessageDialog(frame.getComponent(0), "Login Failed");

                }

            }

        });

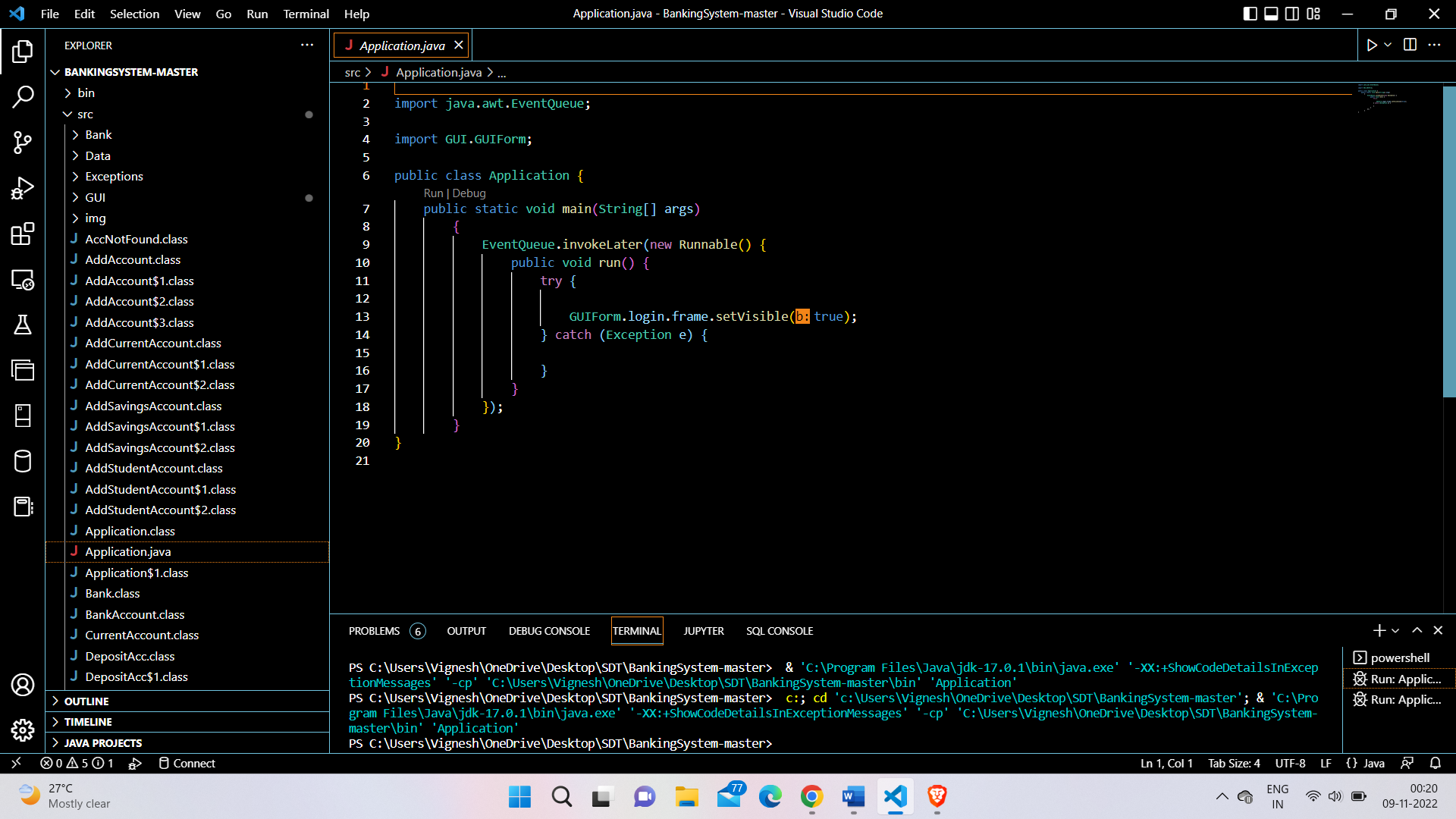
        btnLogin.setBounds(260, 138, 89, 23);

        frame.getContentPane().add(btnLogin);

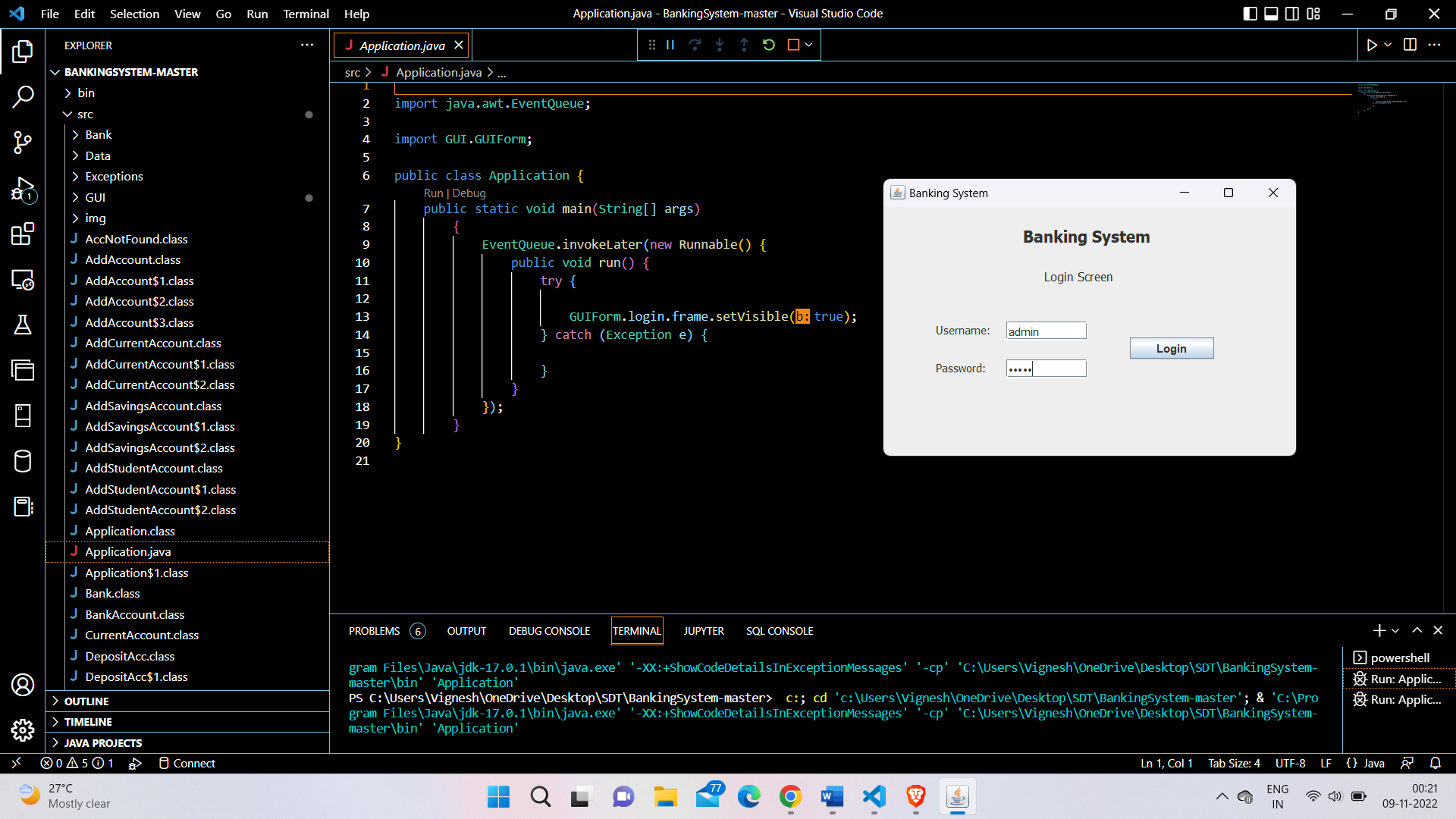
    }

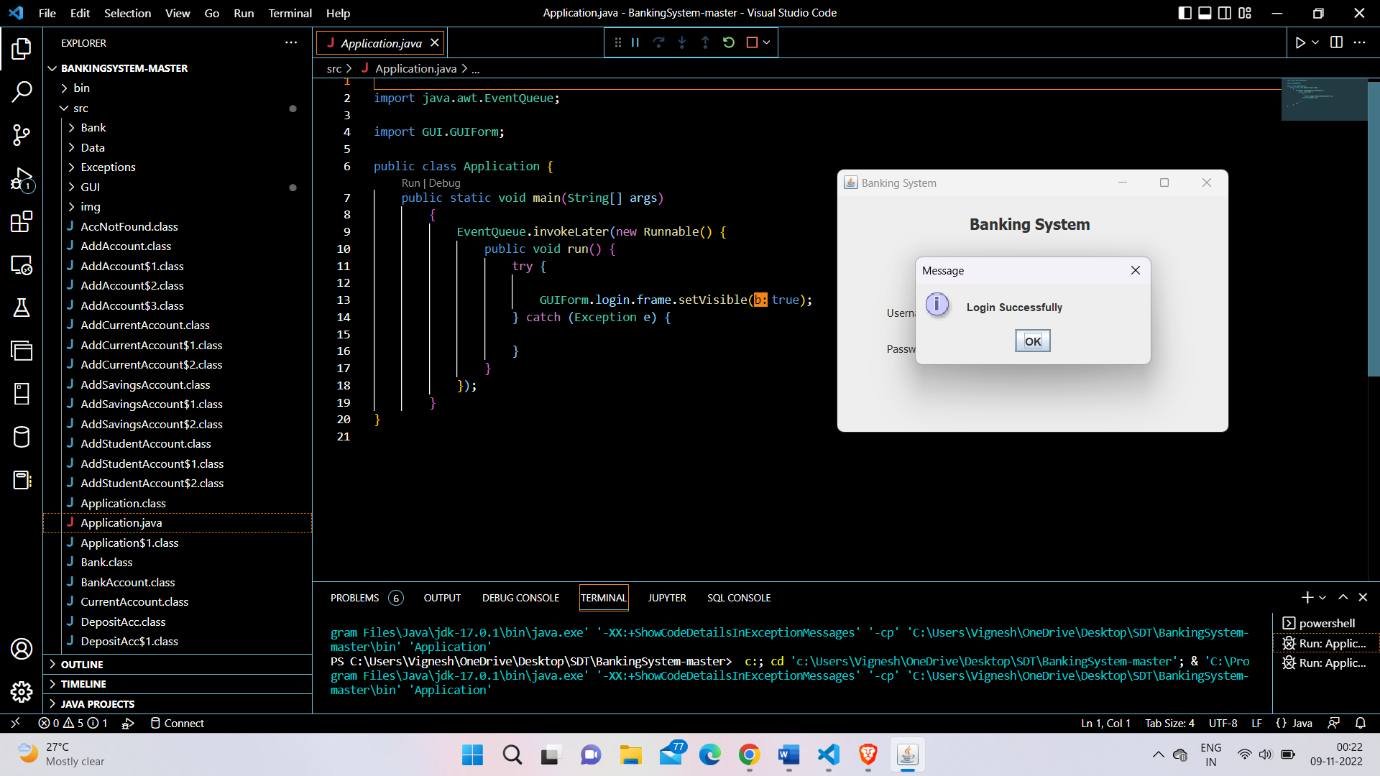
}

**Main Application :**

****

**Screenshots of User Interface :**

****

****

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**JUNIT :**

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.DisplayName;

import org.junit.jupiter.api.Test;

public class TestBankAccountNumberServices {

@DisplayName("Test check for Invalid bank account")

@Test

public void testCheckForInvalidBankAccountNumber() {

BankAccountNumberServices bankAccountNumberServicesObject = new BankAccountNumberServices();

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod1(null));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod1(""));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod1("00000000000000"));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod1("SA123450000000"));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod1("1234567"));

// Using Method 2

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod2(null));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod2(""));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod2("00000000000000"));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod2("SA123450000000"));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberMethod2("1234567"));

// Using regular expression check

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression(null));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression(""));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression("00000000000000"));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression("SA123450000000"));

assertEquals(false, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression("1234567"));

}

@DisplayName("Test check for Valid bank account")

@Test

public void testCheckForvalidBankAccountNumber() {

BankAccountNumberServices bankAccountNumberServicesObject = new BankAccountNumberServices();

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberMethod1("12345678901234"));

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberMethod1("11223344551234"));

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberMethod1("11022033012346"));

// Using method 2

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberMethod2("12345678901234"));

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberMethod2("11223344551234"));

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberMethod2("11022033012346"));

// Using regular expression check

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression("12345678901234"));

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression("11223344551234"));

assertEquals(true, bankAccountNumberServicesObject.isValidAccountNumberUsingRegularExpression("11022033012346"));

}

}

**Graphical user interface, text, application, email

Description automatically generated**