**Internship Project Report**

**Name :** Hariom Vishwakarma

**Email ID :** [hariomvishwakarma115@gmail.com](mailto:hariomvishwakarma115@gmail.com)

**Task Start Date** : 27/02/2024

**Task Submition** **Date** : 04/02/2024

**Task Title** : Java Development Internship Project

**Task Description :** The task involved creating an ATM interface program using Java to simulate the

functionalities of a real ATM machine. The program allows users to perform banking

operations such as checking balance, withdrawing money, and depositing money.

Users are prompted to enter their user ID and PIN for authentication, and error

handling mechanisms are implemented to handle invalid user input and insufficient

funds.

**Github link :** [**https://github.com/Hariom115/ATM-Interface**](https://github.com/Hariom115/ATM-Interface)

**Contant**

1. **Steps Taken**
2. **Challenges Faced**
3. **Solutions Implemented**
4. **Learnings**
5. **Project Update**
6. **Reporting Benefits**
7. **Source Code**
8. **Resource**

**Steps Taken :**

1. Created a Java project in my preferred IDE.

2. Defined a User class to represent each user of the ATM, including attributes like userID, userPIN, and

accountBalance.

3. Defined an ATM class encapsulating ATM functionalities, including methods for operations like checking

balance, withdrawing money, and depositing money.

4. Implemented user authentication by prompting users to enter their user ID and PIN upon startup and

validating the entered credentials against stored user data.

5. Utilized Java's input/output functionalities to interact with users, displaying appropriate messages and

prompts to guide users through the ATM interface.

6. Implemented error handling mechanisms to deal with invalid user input and insufficient funds,

providing informative error messages to assist users in correcting their mistakes.

7. Tested the program with various scenarios to ensure functionality and reliability, debugging any issues

encountered during testing and making necessary adjustments to the code.

8. Documented the code using comments to explain the purpose of each class, method, and significant

block of code. Included a README file with instructions on running the program and other relevant

information.

**Challenges Faced :**

- Ensuring secure authentication and handling of user PINs.

- Implementing error handling mechanisms to cover all possible scenarios.

- Ensuring intuitive user interface design and interaction flow.

**Solutions Implemented :**

- Used secure methods for storing and validating user PINs.

- Implemented comprehensive error handling logic to cover various scenarios.

- Designed a user-friendly interface with clear instructions and feedback messages.

**Learnings :**

- Enhanced my understanding of Java programming, especially GUI development.

- Learned about implementing secure authentication and error handling in real-world applications.

- Improved my skills in designing intuitive user interfaces for better user experience.

**Project Update :**

-The ATM interface program has been successfully developed, tested, and documented. It provides users with a secure and intuitive way to perform banking operations. The program is ready for deployment and use.

**Reporting Benefits**

- Regularly updating the project progress through reports enhances transparency and facilitates effective communication.

- Reporting provides an opportunity to showcase accomplishments, challenges faced, and solutions implemented to the technical team, fostering a collaborative environment.

**Source Code**

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class ATMInterface extends JFrame {

    private float balance = 1000.0f;

    private int pin = 2223;

    private boolean pinVerified = false;

    private JLabel title, balanceLabel, label, labelname, labeltask, labelpswd;

    private JTextField pinField;

    private JButton verifyButton, checkBalance, withdrawButton, depositButton, cencelButton;

    public ATMInterface() {

        setTitle("ATM Interface");

        setBounds(300, 100, 300,400);

        getContentPane().setBackground(new Color(21,66, 155));

        setLayout(null);

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        labeltask = new JLabel("Task-2 ||");

        labeltask.setBounds(0,0,50,0);

        labeltask.setFont(new Font("Tahoma", Font.PLAIN, 10));

        labeltask.setForeground(Color.WHITE);

        labelname = new JLabel("By Hariom Vishwakarma ||");

        labelname.setBounds(0,10,200,0);

        labelname.setFont(new Font("Tahoma", Font.PLAIN, 10));

        labelname.setForeground(Color.WHITE);

        labelpswd = new JLabel("PIN : 2223");

        labelpswd.setBounds(0,20,50,0);

        labelpswd.setFont(new Font("Tahoma", Font.PLAIN, 10));

        labelpswd.setForeground(Color.WHITE);

        title = new JLabel("ATM INTERFACE");

        title.setBounds(0,80,300, 50);

        title.setFont(new Font("Tahoma", Font.BOLD, 24));

        title.setForeground(Color.WHITE);

        balanceLabel = new JLabel("");

        balanceLabel.setBounds(50, 150, 300, 30);

        balanceLabel.setFont(new Font("Tahoma", Font.PLAIN, 24));

        balanceLabel.setForeground(new Color(255,0, 6));

        label = new JLabel("Enter PIN : ");

        label.setBounds(50, 100, 50, 30);

        label.setFont(new Font("Tahoma", Font.PLAIN, 24));

        label.setForeground(Color.WHITE);

        pinField = new JTextField(10);

        pinField.setBounds(50, 150, 50, 20);

        pinField.setFont(new Font("Tahoma", Font.PLAIN, 24));

        verifyButton = new JButton("Verify PIN");

        verifyButton.setBounds(30, 250, 50, 30);

        verifyButton.setFont(new Font("Tahoma", Font.PLAIN, 22));

        verifyButton.setBackground(new Color(241,8, 119));

        verifyButton.setForeground(Color.WHITE);

        checkBalance = new JButton("Check Balance");

        checkBalance.setBounds(80, 250, 50, 30);

        checkBalance.setFont(new Font("Tahoma", Font.PLAIN, 22));

        checkBalance.setBackground(new Color(8,186, 245));

        checkBalance.setForeground(Color.WHITE);

        withdrawButton = new JButton("Withdraw");

        withdrawButton.setBounds(30, 300, 50, 30);

        withdrawButton.setFont(new Font("Tahoma", Font.PLAIN, 22));

        withdrawButton.setBackground(new Color(8,186, 245));

        withdrawButton.setForeground(Color.WHITE);

        depositButton = new JButton("Deposit");

        depositButton.setBounds(80, 350, 50, 30);

        depositButton.setFont(new Font("Tahoma", Font.PLAIN, 22));

        depositButton.setBackground(new Color(8,186, 245));

        depositButton.setForeground(Color.WHITE);

        cencelButton = new JButton("cancel");

        cencelButton.setBounds(100, 400, 50, 30);

        cencelButton.setFont(new Font("Tahoma", Font.PLAIN, 22));

        cencelButton.setBackground(new Color(8,186, 245));

        cencelButton.setForeground(Color.WHITE);

        cencelButton.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                String msg =e.getActionCommand();

                if(msg.equals("cancel")){

                    setVisible(false);

                     }

            }

        });

        verifyButton.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                int enteredPin = Integer.parseInt(pinField.getText());

                if (enteredPin == pin) {

                    pinVerified = true;

                    verifyButton.setEnabled(false);

                    withdrawButton.setEnabled(true);

                    depositButton.setEnabled(true);

                    checkBalance.setEnabled(true);

                    JOptionPane.showMessageDialog(ATMInterface.this, "PIN verified");

                } else {

                    JOptionPane.showMessageDialog(ATMInterface.this, "Invalid PIN");

                }

            }

        });

        checkBalance.setEnabled(false);

        checkBalance.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                int enteredPin = Integer.parseInt(pinField.getText());

                if (enteredPin == pin) {

                    JOptionPane.showMessageDialog(ATMInterface.this, "Current Balance: Rs" + balance);

                } else {

                    JOptionPane.showMessageDialog(ATMInterface.this, "Invalid PIN");

                }

                pinField.setText("");

            }

        });

        withdrawButton.setEnabled(false);

        withdrawButton.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                String amountStr = JOptionPane.showInputDialog(ATMInterface.this, "Enter amount to withdraw:");

                float amount = Float.parseFloat(amountStr);

                if (amount <= balance) {

                    balance -= amount;

                    balanceLabel.setText("Balance: Rs " + balance);

                    JOptionPane.showMessageDialog(ATMInterface.this, "Withdrawal successful");

                } else {

                    JOptionPane.showMessageDialog(ATMInterface.this, "Insufficient balance");

                }

            }

        });

        depositButton.setEnabled(false);

        depositButton.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                String amountStr = JOptionPane.showInputDialog(ATMInterface.this, "Enter amount to deposit:");

                float amount = Float.parseFloat(amountStr);

                balance += amount;

                balanceLabel.setText("Balance: Rs " + balance);

                JOptionPane.showMessageDialog(ATMInterface.this, "Deposit successful");

            }

        });

        setLayout(new FlowLayout());

        add(labeltask);

        add(labelname);

        add(labelpswd);

        add(title);

        add(balanceLabel);

        add(label);

        add(pinField);

        add(verifyButton);

        add(checkBalance);

        add(withdrawButton);

        add(depositButton);

        add(cencelButton);

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(new Runnable() {

            @Override

            public void run() {

                ATMInterface atm = new ATMInterface();

                atm.setVisible(true);

            }

        });

    }

}

**Resource**

**Overall, the project was a valuable learning experience, allowing me to enhance my Java programming skills and gain practical experience in developing real-world applications.**

