SOLAPUR EDUCATION SOCIETY'S

S.E.S. POLYTECHNIC, SOLAPUR

Samrat Chowk, Solapur

C E R T I F I C A T E

This is to certify that the Following students of Computer Department

1. Pavan Vishnu Damji - 2513

2 Venkatesh Sudhir Soma - 2508

3. Prajwal Yallappa Sanade - 2509

has satisfactorily completed micro-project titled **Online Payment Gateway Simulation.**

in course **"Java Programming"(314317)** as prescribed by Maharashtra State Board Of Technical Education, Mumbai. For the **Fourth** semester (K-Scheme) of Diploma in Computer Technology in Academic Year 2024-25

Date: / / 2025

Staff Incharge	Head of Dept	Principal
(Mrs.G.K. Ghodke)	(Mr.Patil M.C)	(Mr.Bhavtankar A.A.) S.E.S. Polytechnic, Solapur

- **Synopsis :**

This Java program is designed as a simple payment gateway simulation, where users can interact with a mock payment system to perform transactions. It utilizes a graphical user interface built using Java Swing components, such as labels, text fields, buttons, and a text area for displaying status messages. The system requires the user to input a card number, password, and an OTP (one-time password) to authenticate and authorize a payment. The program also allows the user to check their current balance and validate payments by ensuring sufficient funds, correct card details, and valid OTP. Custom exceptions, like InvalidCardException, InsufficientFundsException, and NetworkTimeoutException, are used to handle various error scenarios that may arise during the transaction process. The program is designed to be user-friendly, guiding users through actions like generating OTPs, entering payment details, and clearing inputs. The overall goal of this microproject is to simulate a basic payment processing system with error handling, making it an ideal demonstration of Java's event-driven programming and exception handling capabilities.

- **Microproject – Course Outcome matrix**

Course Outcomes :

1. **Develop Java Program using classes and objects.**
2. **Develop Java Program for implementing code reusability concept.**
3. **Develop Java Program to implement multithreading and exception handling.**
4. **Develop Java Program for implementing event handling using window based application components.Develop programs using database.**
5. **Implements network programming in java.**
6. **Develop Java Program for managing database.**

Sr. No.	Microproject	CO 1	CO 2	CO 3	CO 4	CO 5	CO 6
1	Name of Microproject	✓	✓	✓	✓		

- **Code**

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.Random;

// Exception classes for custom errors
class InvalidCardException extends Exception {
   public InvalidCardException(String msg) {
      super(msg);
   }
}

class InsufficientFundsException extends Exception {
   public InsufficientFundsException(String msg) {
      super(msg);
   }
}

class NetworkTimeoutException extends Exception {
   public NetworkTimeoutException(String msg) {
      super(msg);
   }
}

public class Micro_Project extends JFrame implements ActionListener {

   JLabel cardLabel, passwordLabel, otpLabel, transferno, amountLabel;
   JTextField cardField, otpField, transferField, generateotpfield, amountField;
   JButton payButton, clearButton, checkBalanceButton, generateotp, exitButton;
   JPasswordField passwordField;
   JTextArea statusArea;
   double balance = 5000.0; // Default balance
   String generatedOTP;
   GridBagConstraints gbc = new GridBagConstraints();
```

```
// Static card number and password
private static final String VALID_CARD_NUMBER = "1234567890";
private static final String VALID_PASSWORD = "password123";

Micro_Project(String str) {
   super(str);
   Container con = getContentPane();
   setLayout(new GridBagLayout());

   // labels
   cardLabel = new JLabel("Card Number:");
   passwordLabel = new JLabel("Enter Password:");
   otpLabel = new JLabel("Confirm OTP:");
   transferno = new JLabel("Enter Card Number to Transfer:");
   amountLabel = new JLabel("Enter Amount to Pay :");

   // textfields
   cardField = new JTextField(20);
    passwordField = new JPasswordField(20);
   passwordField.setEchoChar('*');
   otpField = new JTextField(15);
   transferField = new JTextField(20);
   generateotpfield = new JTextField(20);
   amountField = new JTextField(10);

   // buttons
   payButton = new JButton("Pay");
   clearButton = new JButton("Clear");
   checkBalanceButton = new JButton("Check Balance");
   generateotp = new JButton("Generate Otp");
   exitButton = new JButton("Exit");

   statusArea = new JTextArea(10, 30);
   statusArea.setEditable(false);

   // Setting GridBagConstraints
   gbc.insets = new Insets(5, 5, 5, 5); // Space around components
   gbc.fill = GridBagConstraints.HORIZONTAL;
```

```
// Adding components with GridBagLayout
 gbc.gridx = 0;
 gbc.gridy = 0;
 con.add(cardLabel, gbc);
 gbc.gridx = 1;
 con.add(cardField, gbc);

 gbc.gridx = 0;
 gbc.gridy = 1;
 con.add(passwordLabel, gbc);
 gbc.gridx = 1;
 con.add(passwordField, gbc);

 gbc.gridx = 0;
 gbc.gridy = 2;
 con.add(generateotp, gbc);
 gbc.gridx = 1;
 con.add(generateotpfield, gbc);

 gbc.gridx = 0;
 gbc.gridy = 3;
 con.add(otpLabel, gbc);
 gbc.gridx = 1;
 con.add(otpField, gbc);

 gbc.gridx = 0;
 gbc.gridy = 4;
 con.add(amountLabel, gbc);
 gbc.gridx = 1;
 con.add(amountField, gbc);

 gbc.gridx = 0;
 gbc.gridy = 5;
 con.add(transferno, gbc);
 gbc.gridx = 1;
 con.add(transferField, gbc);

 gbc.gridx = 0;
 gbc.gridy = 6;
```

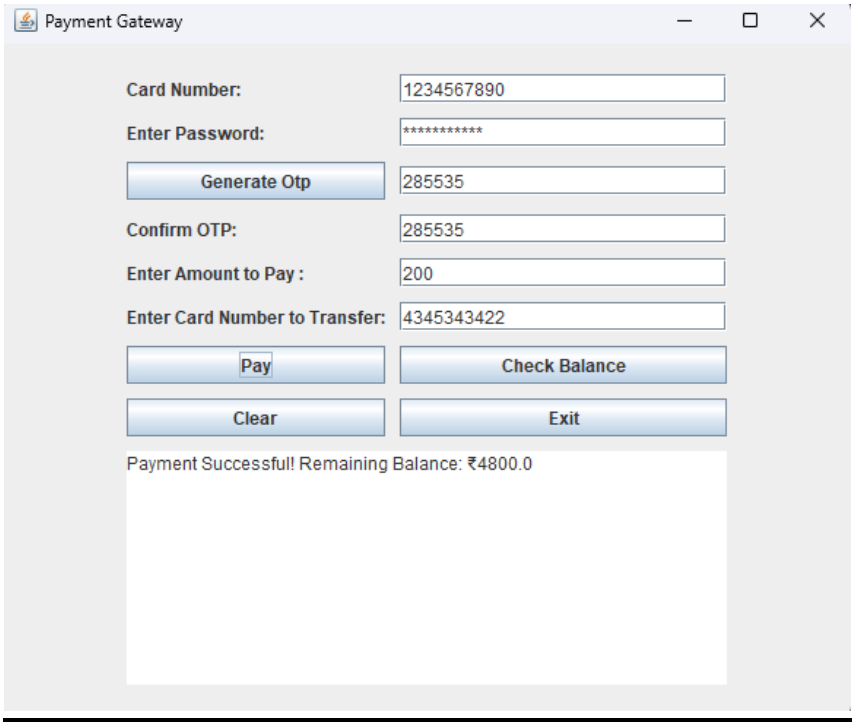
```
        con.add(payButton, gbc);
        gbc.gridx = 1;
        con.add(checkBalanceButton, gbc);

        gbc.gridx = 0;
        gbc.gridy = 7;
        con.add(clearButton, gbc);
        gbc.gridx = 1;
        con.add(exitButton, gbc);

        // Status area spanning two columns
        gbc.gridx = 0;
        gbc.gridy = 8;
        gbc.gridwidth = 2; // Span across two columns
        con.add(statusArea, gbc);

        // Action listeners
        payButton.addActionListener(this);
        clearButton.addActionListener(this);
        checkBalanceButton.addActionListener(this);
        generateotp.addActionListener(this);
        exitButton.addActionListener(this);
    }
    // Action performed for the buttons
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == payButton) {
            try {
                processPayment();
            } catch (InvalidCardException | InsufficientFundsException | NetworkTimeoutException
| NumberFormatException ex) {
                statusArea.setText("Error: " + ex.getMessage());
            }
        } else if (e.getSource() == checkBalanceButton) {
            try {
                if (!cardField.getText().equals(VALID_CARD_NUMBER) ||
!passwordField.getText().equals(VALID_PASSWORD) ||
!generateotpfield.getText().equals(otpField.getText())) {
                    throw new InvalidCardException("The Card number/Password/OTP does not
match");
```

```
            }
            else {
               statusArea.setText("Current Balance: ₹" + balance);
            }
         }catch (InvalidCardException cbe)
         {
            statusArea.setText("Error: "+cbe);
         }
      } else if (e.getSource() == generateotp) {
         generateOTP();
      } else if (e.getSource() == clearButton) {
         cardField.setText("");
         passwordField.setText("");
         otpField.setText("");
         transferField.setText("");
         generateotpfield.setText("");
         amountField.setText("");
         statusArea.setText("");
      } else if (e.getSource() == exitButton) {
         System.exit(0);
      }
   }
   // Generate OTP method
   void generateOTP() {
      Random rand = new Random();
      int otp = rand.nextInt(900000) + 100000;  // Generate a 6-digit OTP
      generatedOTP = String.valueOf(otp);
      generateotpfield.setText(generatedOTP);
      statusArea.setText("OTP Generated: " + generatedOTP);
   }
   // Process payment method
   void processPayment() throws InvalidCardException, InsufficientFundsException,
NetworkTimeoutException, NumberFormatException {
      String cardNumber = cardField.getText();
      String password = passwordField.getText();
      String otp = otpField.getText();
      String amountText = amountField.getText();

      // Check if card number and password are correct
```

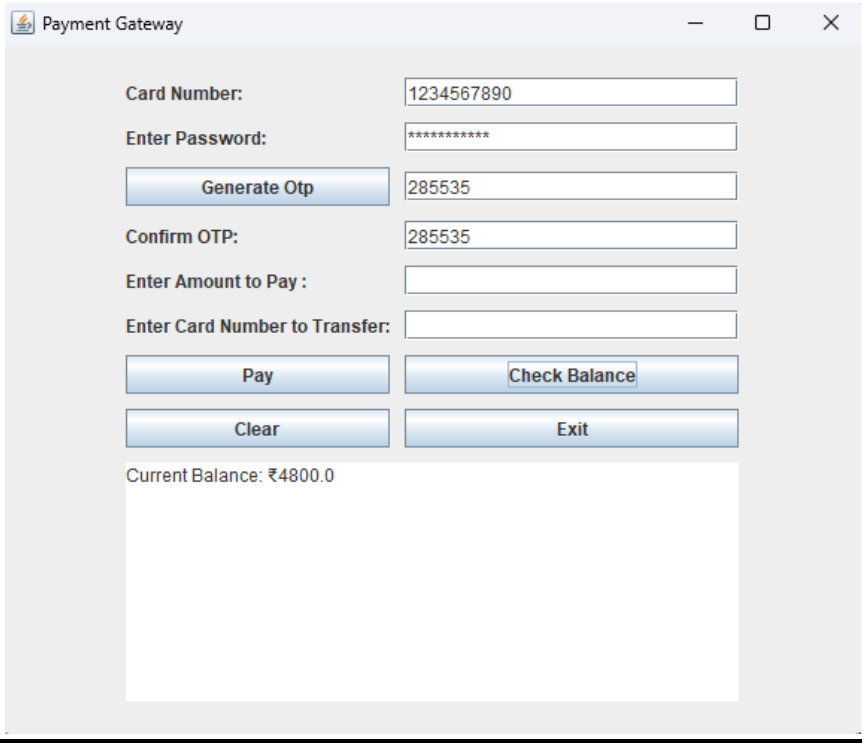
```
    if (!cardNumber.equals(VALID_CARD_NUMBER) || !password.equals(VALID_PASSWORD))
{
        throw new InvalidCardException("Invalid Card Number or Password!");
    }

    // Check if OTP is entered and valid
    if (otp.isEmpty() || !otp.equals(generatedOTP)) {
        throw new InvalidCardException("Invalid OTP!");
    }

    // Validate payment amount
    double amount;
    try {
        amount = Double.parseDouble(amountText);
        if (amount <= 0) {
            throw new NumberFormatException("Amount must be greater than zero.");
        }
    } catch (NumberFormatException ex) {
        throw new NumberFormatException("Invalid Amount! Enter a valid number.");
    }

    // Check if sufficient balance is available
    if (amount > balance) {
        throw new InsufficientFundsException("Insufficient Funds!");
    }

    // Process the payment
    balance -= amount;
    statusArea.setText("Payment Successful! Remaining Balance: ₹" + balance);
  }


  public static void main(String[] args) {
    Micro_Project f = new Micro_Project("Payment Gateway");
    f.setSize(500, 500);
    f.setVisible(true);
  }
}
```

- **Output:**

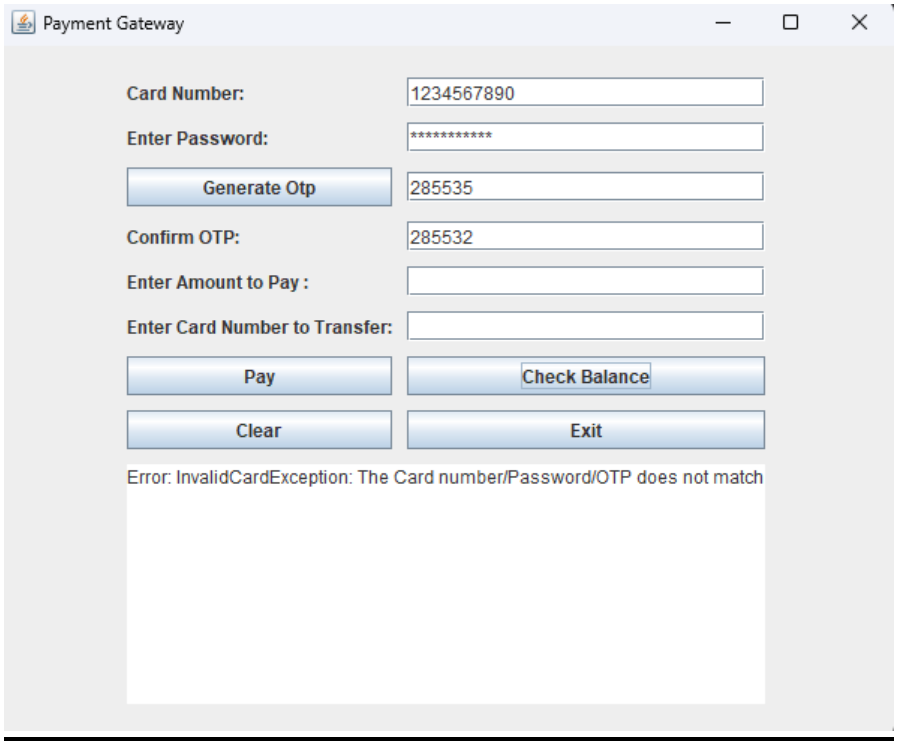
1. Payment Successful



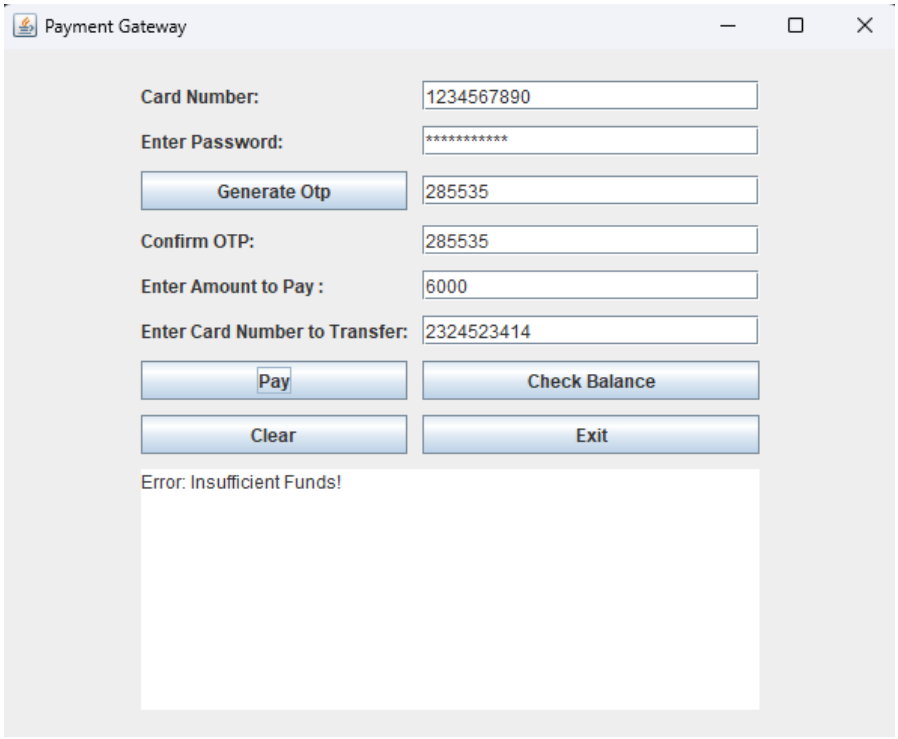
2. Current Balance



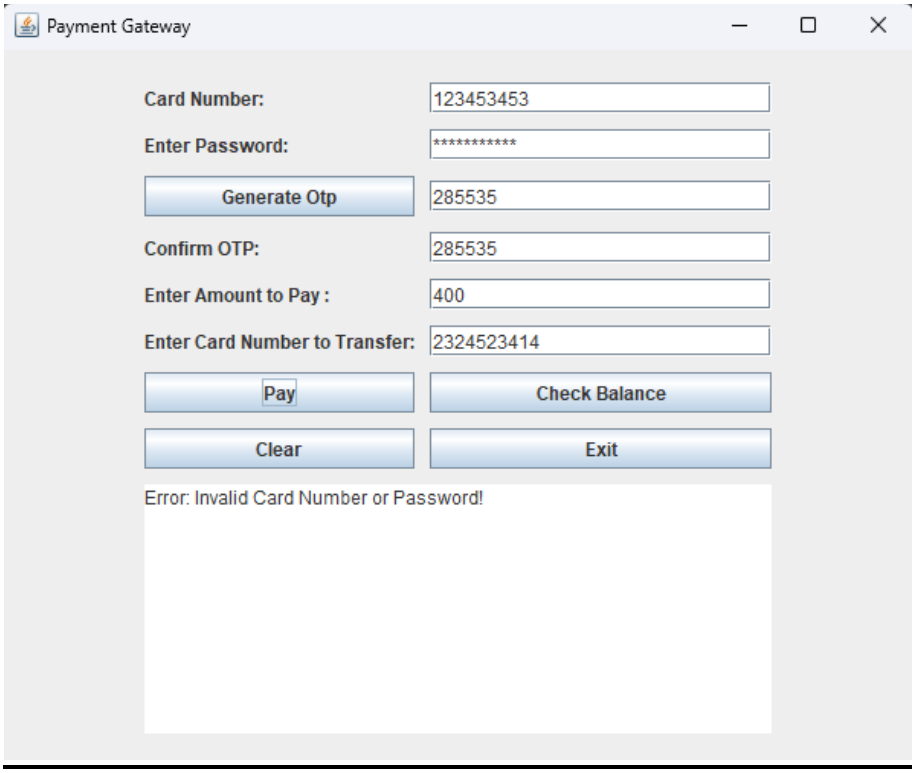
3. Invalid OTP



4. Insufficient Funds!



5. Invalid Card Number



6. Use of Clear Button