**JAVA PROJECT**

**BANKING APPLICATION**

**PROGRAM**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.util.ArrayList;

// Transaction class to store individual transactions

class Transaction {

private String type;

private double amount;

public Transaction(String type, double amount) {

this.type = type;

this.amount = amount;

}

public String getType() {

return type;

}

public double getAmount() {

return amount;

}

@Override

public String toString() {

return type + ": $" + amount;

}

}

// Account class to handle deposit, withdrawal, balance, and transaction history

class Account {

private double balance;

private double totalDeposits;

private double totalWithdrawals;

private ArrayList<Transaction> transactionHistory;

public Account() {

balance = 0.0;

totalDeposits = 0.0;

totalWithdrawals = 0.0;

transactionHistory = new ArrayList<>();

}

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

totalDeposits += amount;

transactionHistory.add(new Transaction("Deposit", amount));

}

}

public void withdraw(double amount) {

if (amount > 0 && amount <= balance) {

balance -= amount;

totalWithdrawals += amount;

transactionHistory.add(new Transaction("Withdrawal", amount));

}

}

public double getBalance() {

return balance;

}

public double getTotalDeposits() {

return totalDeposits;

}

public double getTotalWithdrawals() {

return totalWithdrawals;

}

public String getTransactionHistory() {

if (transactionHistory.isEmpty()) {

return "No transactions yet.";

}

StringBuilder history = new StringBuilder();

for (Transaction t : transactionHistory) {

history.append(t.toString()).append("\n");

}

return history.toString();

}

public String getTransactionSummary() {

return "Total Deposits: $" + String.format("%.2f", totalDeposits) + "\n" +

"Total Withdrawals: $" + String.format("%.2f", totalWithdrawals) + "\n" +

"Final Balance: $" + String.format("%.2f", balance);

}

}

// GUI class to create the frontend using Swing

public class BankingAppGUI {

private Account account = new Account();

// GUI Components

private JFrame frame;

private JTextField amountField;

private JTextArea transactionArea;

private JLabel balanceLabel;

public BankingAppGUI() {

// Create the main frame

frame = new JFrame("Banking Application");

frame.setSize(400, 400);

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

frame.setLayout(new BorderLayout());

// Panel for input and buttons

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(4, 2, 10, 10));

// Amount input field

JLabel amountLabel = new JLabel("Enter Amount:");

amountField = new JTextField();

panel.add(amountLabel);

panel.add(amountField);

// Deposit button

JButton depositButton = new JButton("Deposit");

depositButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

depositAction();

}

});

panel.add(depositButton);

// Withdraw button

JButton withdrawButton = new JButton("Withdraw");

withdrawButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

withdrawAction();

}

});

panel.add(withdrawButton);

// Transaction summary button

JButton summaryButton = new JButton("Transaction Summary");

summaryButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

showTransactionSummary();

}

});

panel.add(summaryButton);

// Add panel to the frame

frame.add(panel, BorderLayout.NORTH);

// Transaction history area

transactionArea = new JTextArea(10, 30);

transactionArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(transactionArea);

frame.add(scrollPane, BorderLayout.CENTER);

// Balance label

balanceLabel = new JLabel("Balance: $0.00");

frame.add(balanceLabel, BorderLayout.SOUTH);

// Display the window

frame.setVisible(true);

}

// Handle deposit action

private void depositAction() {

try {

double amount = Double.parseDouble(amountField.getText());

account.deposit(amount);

updateUI();

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(frame, "Please enter a valid number.");

}

}

// Handle withdraw action

private void withdrawAction() {

try {

double amount = Double.parseDouble(amountField.getText());

if (amount > account.getBalance()) {

JOptionPane.showMessageDialog(frame, "Insufficient balance.");

} else {

account.withdraw(amount);

updateUI();

}

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(frame, "Please enter a valid number.");

}

}

// Show transaction summary in a dialog

private void showTransactionSummary() {

String summary = account.getTransactionSummary();

JOptionPane.showMessageDialog(frame, summary, "Transaction Summary", JOptionPane.INFORMATION\_MESSAGE);

}

// Update the transaction history and balance on the UI

private void updateUI() {

transactionArea.setText(account.getTransactionHistory());

balanceLabel.setText("Balance: $" + String.format("%.2f", account.getBalance()));

amountField.setText(""); // Clear input field

}

public static void main(String[] args) {

// Run the application

SwingUtilities.invokeLater(new Runnable() {

@Override

public void run() {

new BankingAppGUI();

}

});

}

}