TASK-3(ATM INTERFACE)

SOURCE CODE:

package Task3;

import java.util.\*;

class BankAccount {

    private static int accountNumberCounter = 1000000000;

    private static HashSet<Integer> existingAccountNumbers = new HashSet<>();

    private double balance;

    private String accountNumber;

    private List<String> transactionHistory;

    public BankAccount(double initialBalance) {

        int newAccountNumber;

        do {

            newAccountNumber = accountNumberCounter++;

        } while (existingAccountNumbers.contains(newAccountNumber));

        existingAccountNumbers.add(newAccountNumber);

        this.accountNumber = String.valueOf(newAccountNumber);

        this.balance = initialBalance;

        this.transactionHistory = new ArrayList<>();

    }

    public double getBalance() {

        return balance;

    }

    public void deposit(double amount) {

        balance += amount;

        transactionHistory.add("Deposit: Rs " + amount);

        System.out.println("Deposit successful. New balance: Rs " + balance);

    }

    public void withdraw(double amount) {

        if (amount <= balance) {

            balance -= amount;

            transactionHistory.add("Withdrawal: Rs " + amount);

            System.out.println("Withdrawal successful. New balance: Rs " + balance);

        } else {

            System.out.println("Insufficient funds.");

        }

   }

    public List<String> getTransactionHistory() {

        return transactionHistory;

    }

}

class ATM {

    private BankAccount account;

    private int maxPinAttempts = 3;

    public ATM(BankAccount account) {

        this.account = account;

    }

    public void start() {

        Scanner scanner = new Scanner(System.in);

        int attempts = 0;

        String pin;

        while (attempts < maxPinAttempts) {

            System.out.print("Enter your PIN: ");

            pin = scanner.nextLine();

            if (pin.equals("6612")) {

                break;

            } else {

                System.out.println("Incorrect PIN. You have " + (maxPinAttempts - attempts - 1) + " attempts left.");

                attempts++;

            }

        }

        if (attempts == maxPinAttempts) {

            System.out.println("Account blocked.");

            return;

        }

        int choice=0;

        do {

            System.out.println("\nWelcome to ATM");

            System.out.println("1. Check balance");

            System.out.println("2. Withdraw");

            System.out.println("3. Deposit");

            System.out.println("4. Check transaction history");

            System.out.println("5. Exit");

            System.out.print("Enter your choice: ");

            try {

                choice = scanner.nextInt();

                scanner.nextLine();

            } catch (InputMismatchException e) {

                System.out.println("Invalid input. Please enter a number.");

                scanner.nextLine();

                continue;

            }

            switch (choice) {

                case 1:

                    checkBalance();

                    break;

                case 2:

                    withdraw();

                    break;

                case 3:

                    deposit();

                    break;

                case 4:

                    checkTransactionHistory();

                    break;

                case 5:

                    System.out.println("Thank you for using the ATM.");

                    break;

                default:

                    System.out.println("Invalid choice.");

            }

        } while (choice != 5);

    }

    private void checkBalance() {

        System.out.println("Your balance is: Rs " + account.getBalance());

    }

    private void withdraw() {

        Scanner scanner = new Scanner(System.in);

        if(account.getBalance()>1000){

         System.out.print("Enter withdrawal amount: Rs ");

         double amount;

         try {

            amount = scanner.nextDouble();

            if (amount <= 0) {

                throw new IllegalArgumentException("Withdrawal amount must be positive.");

            }

            account.withdraw(amount);

         } catch (InputMismatchException e) {

            System.out.println("Invalid input. Please enter a valid amount.");

         } catch (IllegalArgumentException e) {

            System.out.println(e.getMessage());

         }

        }

        else{

            System.out.println("Insufficient Funds.");

        }

    }

    private void deposit() {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter deposit amount: Rs ");

        double amount;

        try {

            amount = scanner.nextDouble();

            if (amount <= 0) {

                throw new IllegalArgumentException("Deposit amount must be positive.");

            }

            account.deposit(amount);

        } catch (InputMismatchException e) {

            System.out.println("Invalid input. Please enter a valid amount.");

        } catch (IllegalArgumentException e) {

            System.out.println(e.getMessage());

        }

    }

    private void checkTransactionHistory() {

        List<String> history = account.getTransactionHistory();

        if (history.isEmpty()) {

            System.out.println("No transaction history available.");

        } else {

            System.out.println("Transaction History:");

            for (String transaction : history) {

                System.out.println(transaction);

            }

        }

    }

}

public class Atmm {

    public static void main(String args[]) {

        BankAccount account = new BankAccount(1000);

        ATM atm = new ATM(account);

        atm.start();

    }

}

OUTPUT:

