Mini Banking System

1. Abstract

This project implements a Mini Banking System using Java with Object-Oriented Programming (OOP) principles and HashMap. It simulates essential banking operations such as creating accounts, depositing money, withdrawing money, checking balances, and displaying all accounts. Accounts are stored in a HashMap with account numbers as keys, ensuring efficient data management.

2. Objectives

- Develop a console-based Mini Banking System in Java.
- Use OOP concepts like classes and encapsulation.
- Store accounts efficiently using HashMap.
- Provide features like Create, Deposit, Withdraw, Balance Check, and Display Accounts.
- Ensure proper validation for transactions.

3. Modules / Features

- Account Creation Module: Allows creation of new accounts with account number, name, and initial deposit.
- Deposit Module: Deposits money into an account.
- Withdraw Module: Allows withdrawal if sufficient balance exists.
- Balance Inquiry Module: Displays current account balance.
- Display Accounts Module: Lists all accounts in the system.

4. Technologies Used

- Programming Language: Java
- Data Structure: HashMap
- Paradigm: Object-Oriented Programming (OOP)
- Platform: Any Java IDE or Command Line

5. Sample Output

```
=== Mini Banking System ===
1. Create Account
2. Deposit
3. Withdraw
4. Check Balance
5. Display All Accounts
6. Exit

Enter choice: 1
Enter Account Number: 101
Enter Account Holder Name: Alice
Enter Initial Deposit: 5000
■ Account created successfully!

Enter choice: 2
Enter Account Number: 101
Enter Deposit Amount: 2000
■ Deposit successful! New Balance: 7000

Enter choice: 4
```

```
Enter Account Number: 101
■ Balance: 7000
```

6. Source Code

```
import java.util.HashMap;
import java.util.Scanner;

// Account class
class Account {
    private String accountNumber;
    private String accountHolderName;
    private double balance;

    public Account(String accountNumber, String accountHolderName, double initialBalance) {
        this.accountNumber = accountNumber;
        this.accountHolderName = accountHolderName;
        this.balance = initialBalance;
    }

    public String getAccountNumber() { return accountNumber; }
    public String getAccountHolderName() { return accountHolderName; }
    public double getBalance() { return balance; }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("■ Deposit successful! New Balance: " + balance);
        } else {
            System.out.println("■ Invalid deposit amount.");
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("■ Withdrawal successful! New Balance: " + balance);
        } else {
            System.out.println("■ Insufficient balance or invalid amount.");
        }
    }

    public void displayAccount() {
        System.out.println("Account Number: " + accountNumber +
                " | Holder: " + accountHolderName +
                " | Balance: " + balance);
    }
}

// Bank class (manages accounts using HashMap)
class Bank {
    private HashMap<String, Account> accounts = new HashMap<>();

    public void createAccount(String accNo, String name, double initialDeposit) {
        if (!accounts.containsKey(accNo)) {
            Account newAcc = new Account(accNo, name, initialDeposit);
            accounts.put(accNo, newAcc);
            System.out.println("■ Account created successfully!");
        } else {
            System.out.println("■■ Account with this number already exists.");
        }
    }

    public void deposit(String accNo, double amount) {
        Account acc = accounts.get(accNo);
        if (acc != null) acc.deposit(amount);
        else System.out.println("■ Account not found!");
    }

    public void withdraw(String accNo, double amount) {
        Account acc = accounts.get(accNo);
        if (acc != null) acc.withdraw(amount);
        else System.out.println("■ Account not found!");
    }
    public void checkBalance(String accNo) {
```

```
        Account acc = accounts.get(accNo);
        if (acc != null) System.out.println("■ Balance: " + acc.getBalance());
        else System.out.println("■ Account not found!");
    }

    public void displayAllAccounts() {
        if (accounts.isEmpty()) {
            System.out.println("■ No accounts available.");
        } else {
            System.out.println("■ All Accounts:");
            for (Account acc : accounts.values()) acc.displayAccount();
        }
    }
}

// Main class
public class MiniBankingSystem {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Bank bank = new Bank();
        int choice;

        do {
            System.out.println("\n=== Mini Banking System ===");
            System.out.println("1. Create Account");
            System.out.println("2. Deposit");
            System.out.println("3. Withdraw");
            System.out.println("4. Check Balance");
            System.out.println("5. Display All Accounts");
            System.out.println("6. Exit");
            System.out.print("Enter choice: ");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter Account Number: ");
                    String accNo = sc.next();
                    System.out.print("Enter Account Holder Name: ");
                    String name = sc.next();
                    System.out.print("Enter Initial Deposit: ");
                    double initial = sc.nextDouble();
                    bank.createAccount(accNo, name, initial);
                    break;
                case 2:
                    System.out.print("Enter Account Number: ");
                    accNo = sc.next();
                    System.out.print("Enter Deposit Amount: ");
                    double dep = sc.nextDouble();
                    bank.deposit(accNo, dep);
                    break;
                case 3:
                    System.out.print("Enter Account Number: ");
                    accNo = sc.next();
                    System.out.print("Enter Withdraw Amount: ");
                    double wit = sc.nextDouble();
                    bank.withdraw(accNo, wit);
                    break;
                case 4:
                    System.out.print("Enter Account Number: ");
                    accNo = sc.next();
                    bank.checkBalance(accNo);
                    break;
                case 5:
                    bank.displayAllAccounts();
                    break;
                case 6:
                    System.out.println("■ Exiting system. Thank you!");
                    break;
                default:
                    System.out.println("■ Invalid choice.");
            }
        } while (choice != 6);

        sc.close();
    }
}
```

7. Conclusion

The Mini Banking System demonstrates the use of OOP principles and HashMap to build a real-world simulation of banking operations. It strengthens understanding of encapsulation, object management, and HashMap data structure while providing a hands-on approach to transaction validation and data storage.