RIPHAH INTERNATIONAL UNIVERSITY

Object Oriented Programming

Spring 2023

Bank Record System

Project Team

Name of Students	Sap ID	Program	Valid Email Address
Manahil Habib	47876	BSSE	47876@students.riphah.edu.pk
Nagarash Fateh	44815	BSSE	44815@students,riphah.edu.pk

Date of Submission

6/4/2023

Table of Contents

Artifact # 1
Project Proposal

Introduction of the Project

Project Title: Bank Record System

Introduction:

The Bank Record System is a web-based application designed to store the information of account type, account opening form, deposit, withdrawal, and searching the transaction information, transaction report, individual account opening form, and group account. Information about interest rates, statistical summaries of account types, and transaction reports are all shown in the records. This aids in offering the user a versatile solution. When compared to the current system, this one has a lower mistake probability. The system will be developed using Java.

Problem Statement:

Create a Bank Record System to manage all bank records using java.

Proposed Solution:

We will code Bank Record System in Java through Inheritance, Polymorphism, Abstraction and Encapsulation using Object Oriented Programming.

Scope of the Project:

In order to keep track of everything in the bank, the banking system needs to maintain a few records, thus a software program is needed to make the task easier. For instance, the banking system is responsible for maintaining the value of INR and other currencies on a global scale.

Modules Description:

1. Admin
2. Manager
3. Client

Modules Description:

- **Admin:**
    1. Client Details (Search, Update and Delete)
    2. Registration (Manager and Client)
    3. Display Records
- **Manager:**
    1. Registration (Name, ID, Passcode)
    2. Create Accounts

3. Input Records (Create, Update, and Delete)

- **Client:**
  1. Registration (Name, ID, Passcode)
  2. View Account
  3. Contact Us
  4. Feedback

Inheritance

Definition:

Inheritance is the procedure or mechanism of acquiring all the properties and behavior of one class to another, i.e., acquiring the properties and behavior of a child class from the parent class.

Explanation:

In a bank record system, inheritance can be utilized to create different types of bank accounts that share common functionalities. For instance, a "SavingsAccount" class and a "CheckingAccount" class can inherit from a common "BankAccount" class, which may contain common methods like deposit, withdraw, and getBalance, and then add their own specialized methods and data members.

Polymorphism

Definition:

The word polymorphism means having multiple forms. Polymorphism enables objects of different classes to be treated as if they were of the same class.

Explanation:

In a bank record system, polymorphism can be applied to allow different types of bank accounts to be processed uniformly. For example, a "BankAccount" interface or an abstract class can define common methods like deposit, withdraw, and getBalance, which can be implemented differently in different account types. This way, different account objects can be treated uniformly, regardless of their specific types.

Abstraction

Definition:

Data Abstraction is technique whose feature provides us the capability of differentiating essential details that need to be displayed to the user.

Explanation:

In a bank record system, abstraction can be utilized to create abstract classes or interfaces that define common methods and properties without providing any implementation. Concrete classes that inherit from these abstract classes or implement these interfaces can provide the actual implementation. This way, the complexities of the underlying system can be abstracted, and a simplified interface can be exposed to the users of the bank record system.

Encapsulation

Definition:

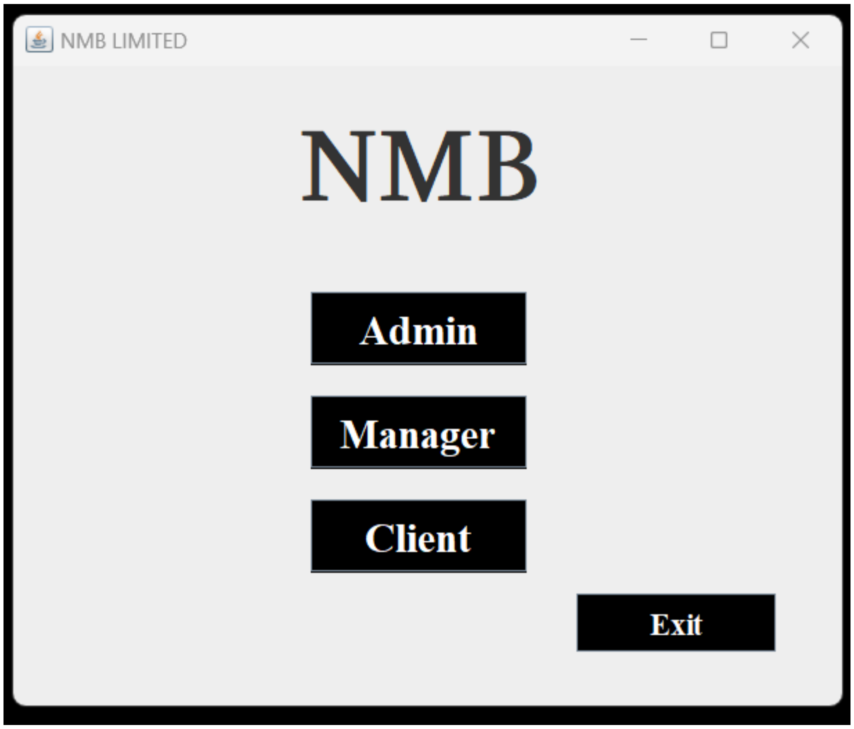
Encapsulation gives us the ability to make variables of a class keep hidden from all other classes of that program or namespace.

Explanation:

In a bank record system, encapsulation can be implemented by using classes and objects to encapsulate data related to bank accounts, transactions, and customers. For example, a class "BankAccount" can have private data members such as account number, balance, and customer information, which can be accessed through methods like deposit, withdraw, and getBalance.

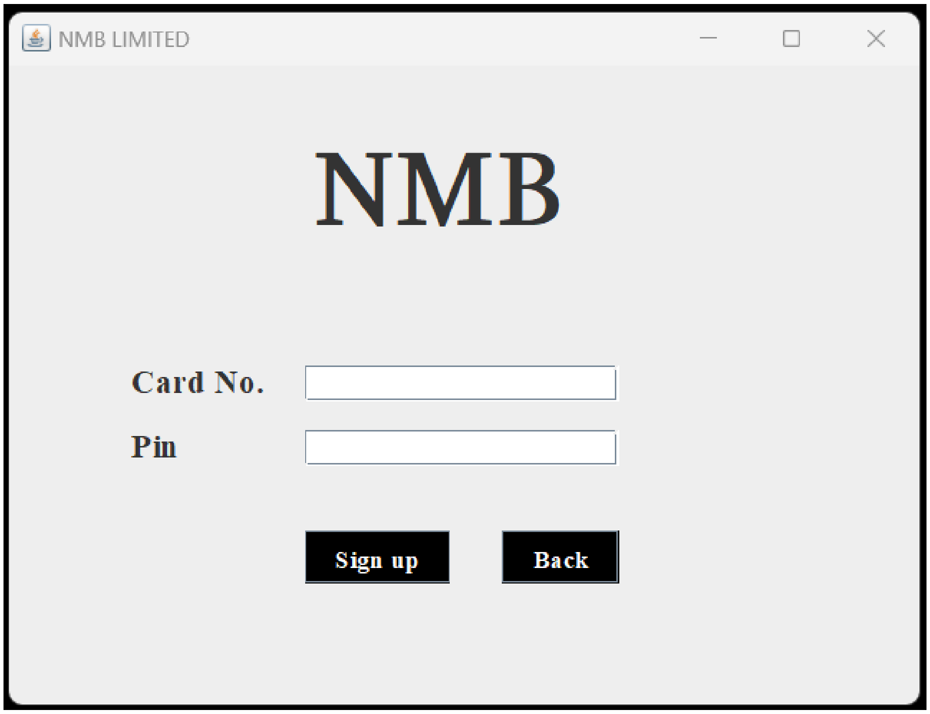
Artifact # 2
Screens

- Main Menu

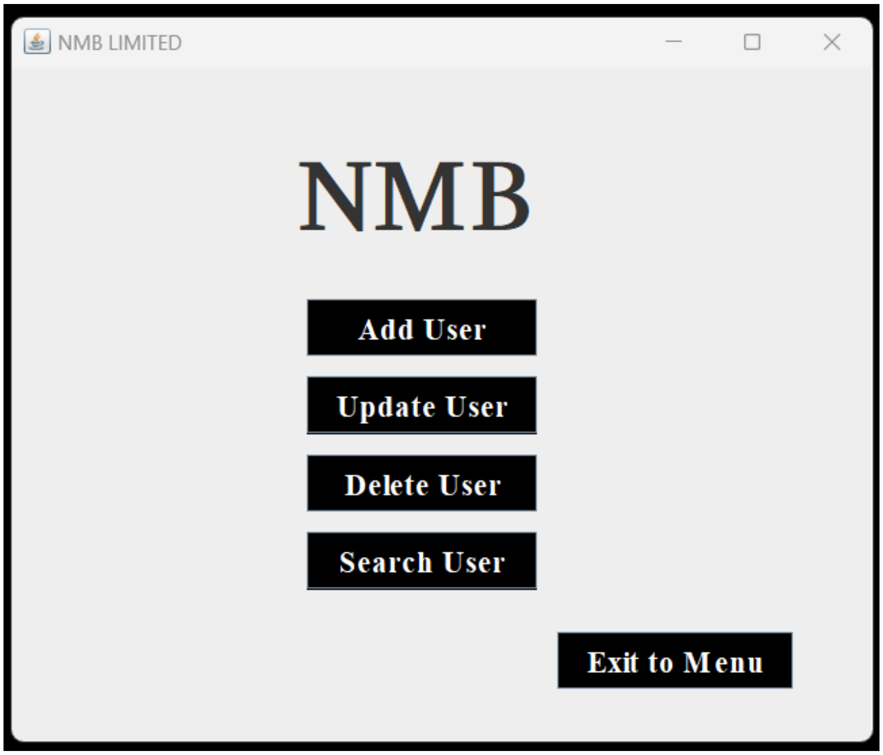


- Admin

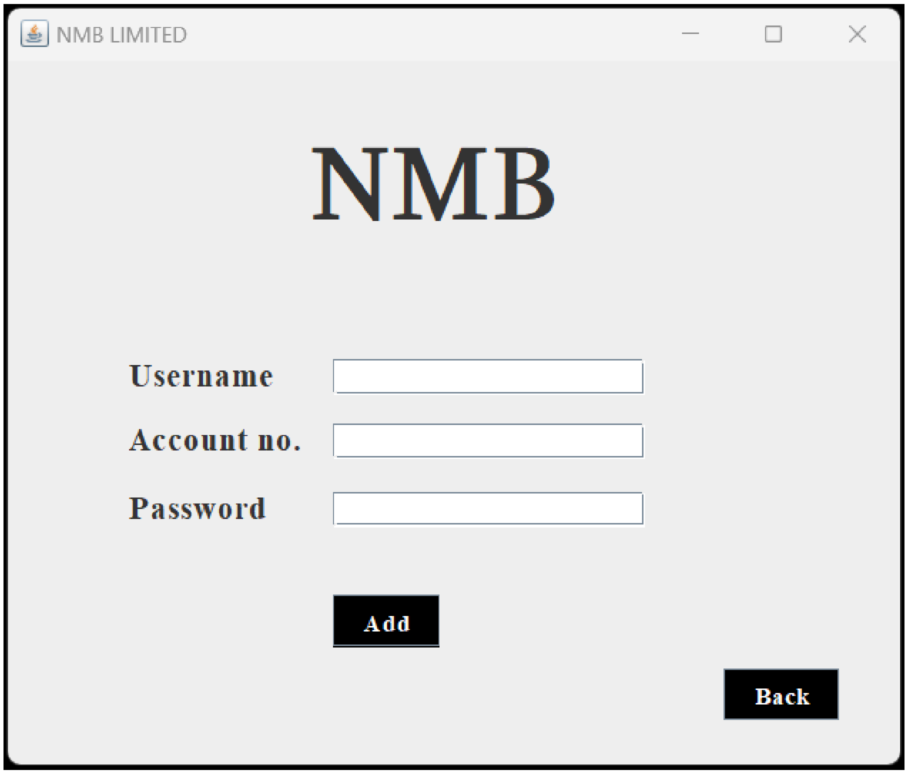
Screen 01 (Reisteration)



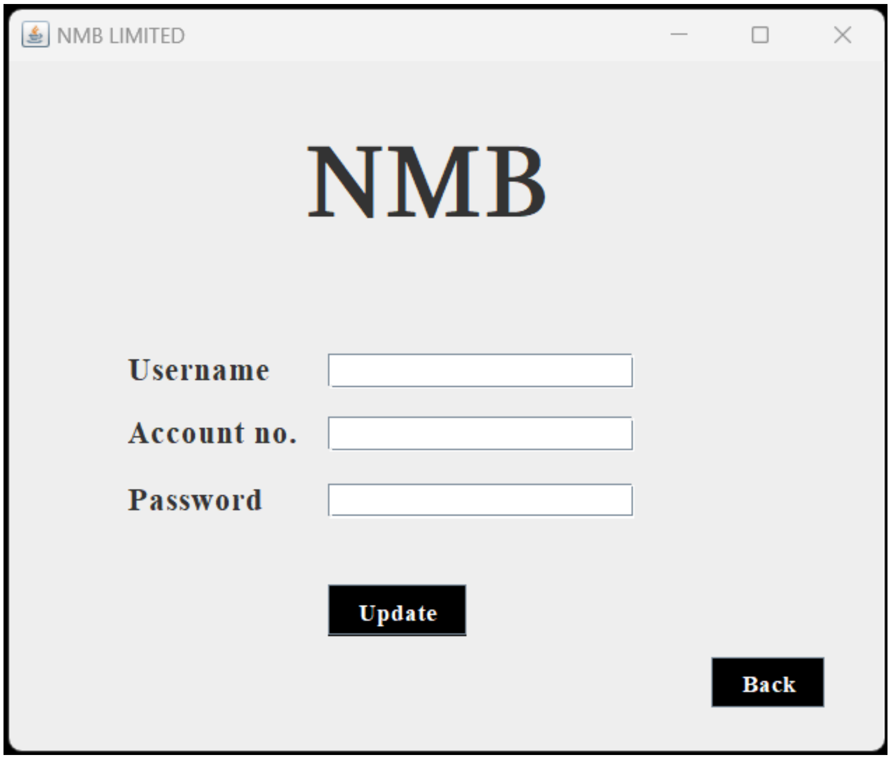
Screen 02 (Admin Menu)



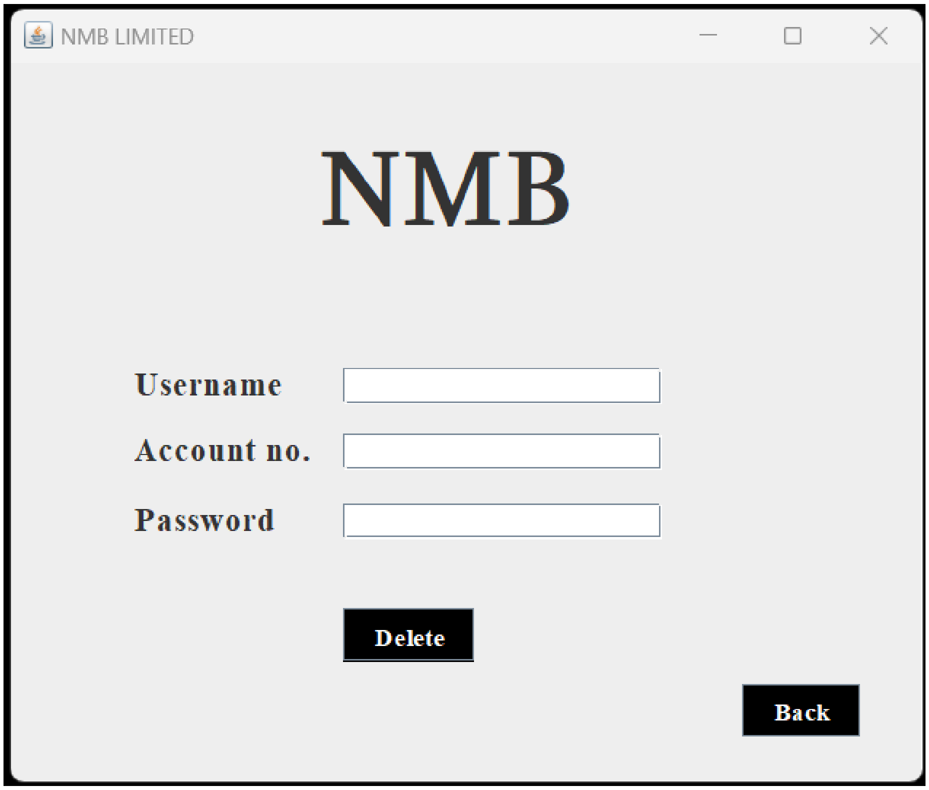
Screen 03 (Add User)



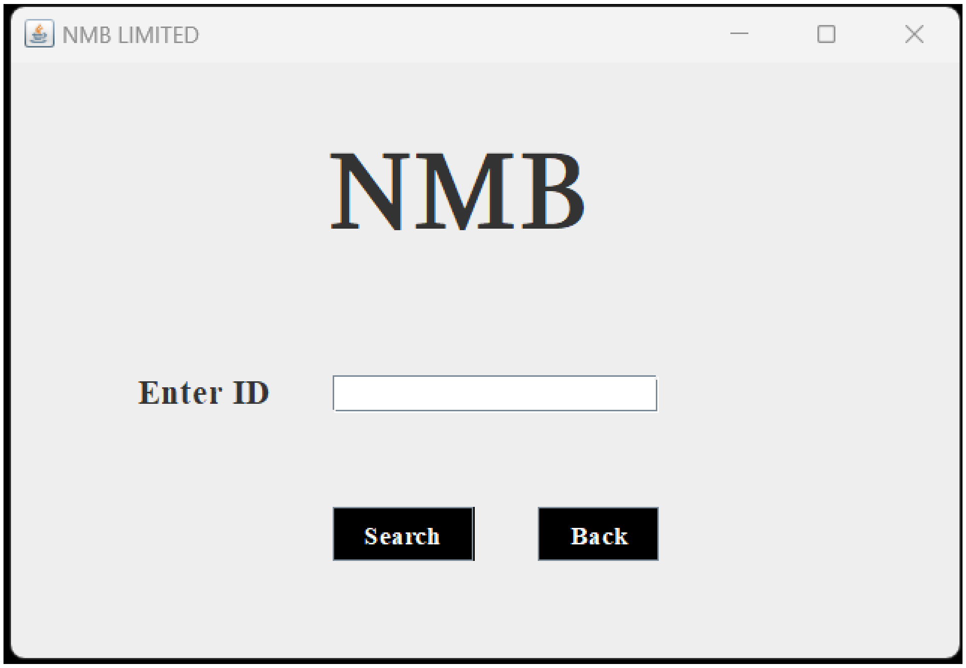
Screen 04 (Update User)



Screen 05 (Delete User)

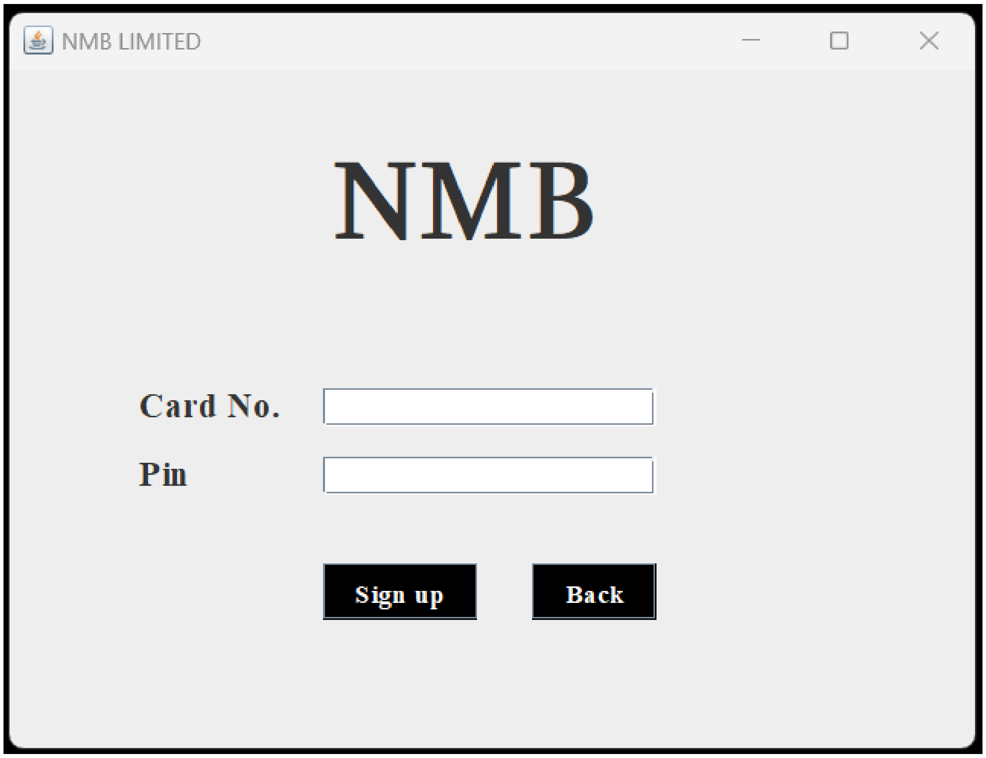


Screen 06 (Search User)

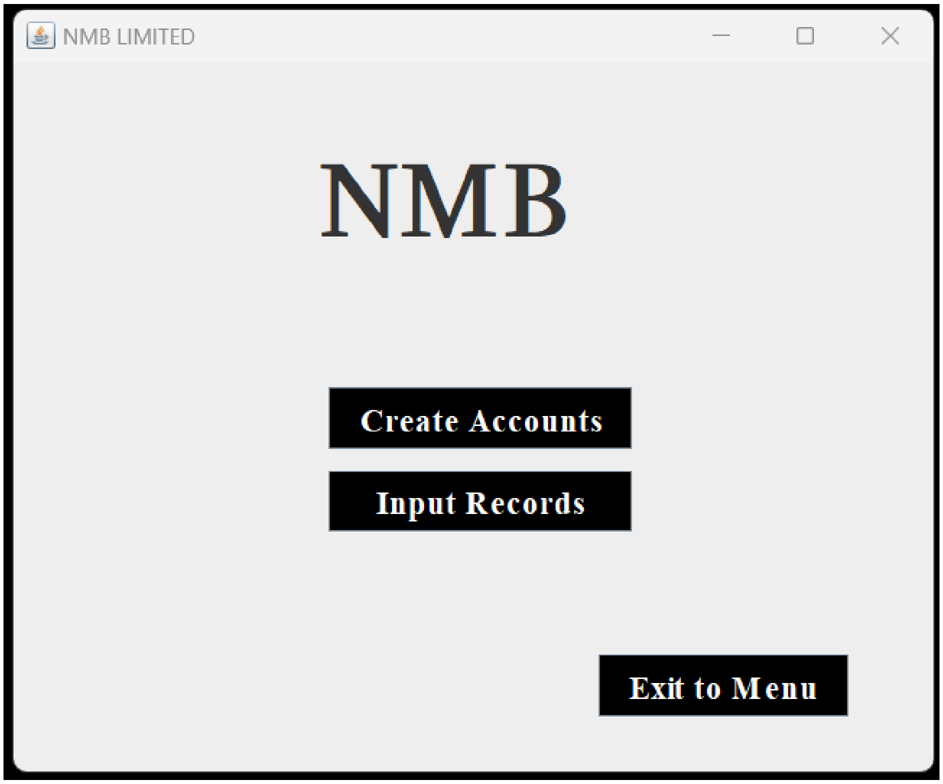


- Manager

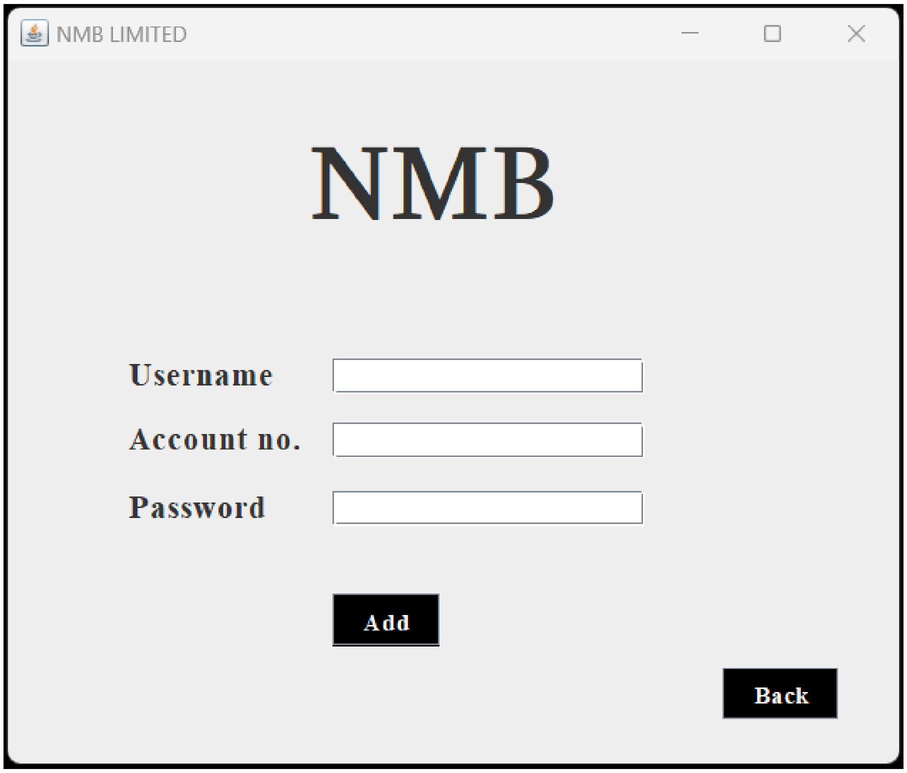
Screen 01 (Registeration)



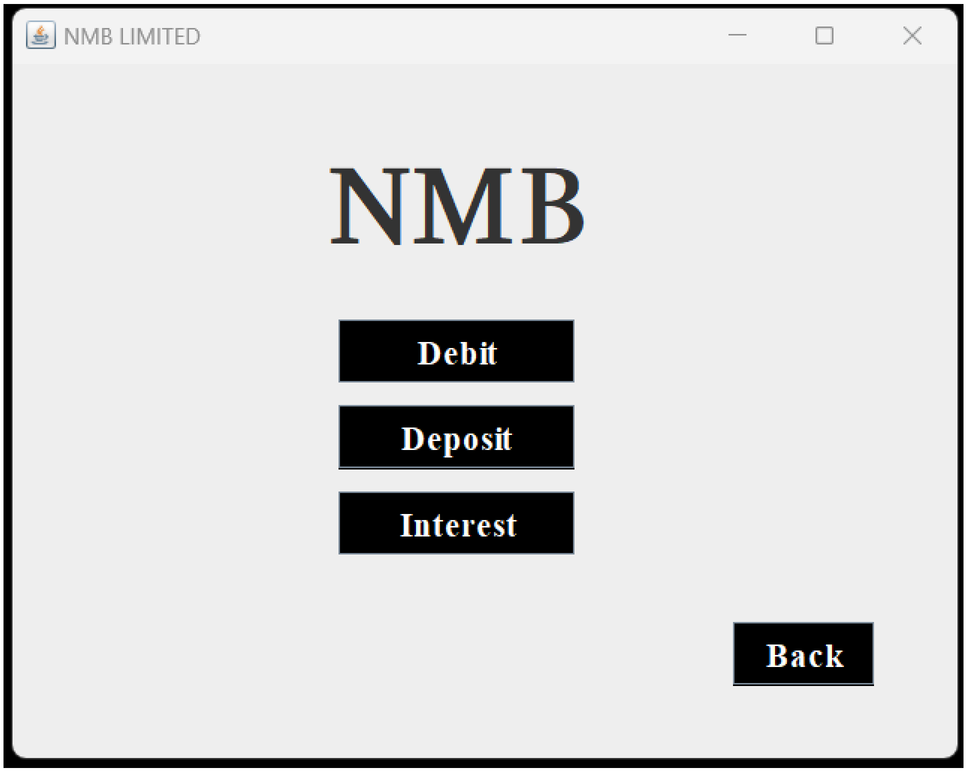
Screen 02 (Manager Menu)



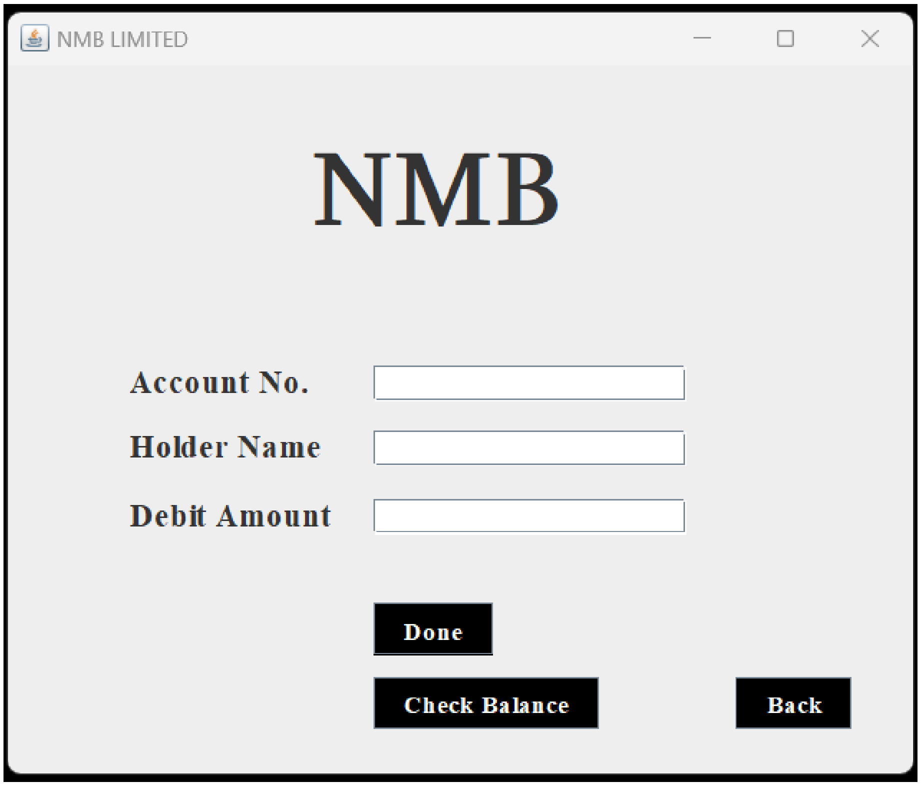
Screen 03 (Create Accounts)



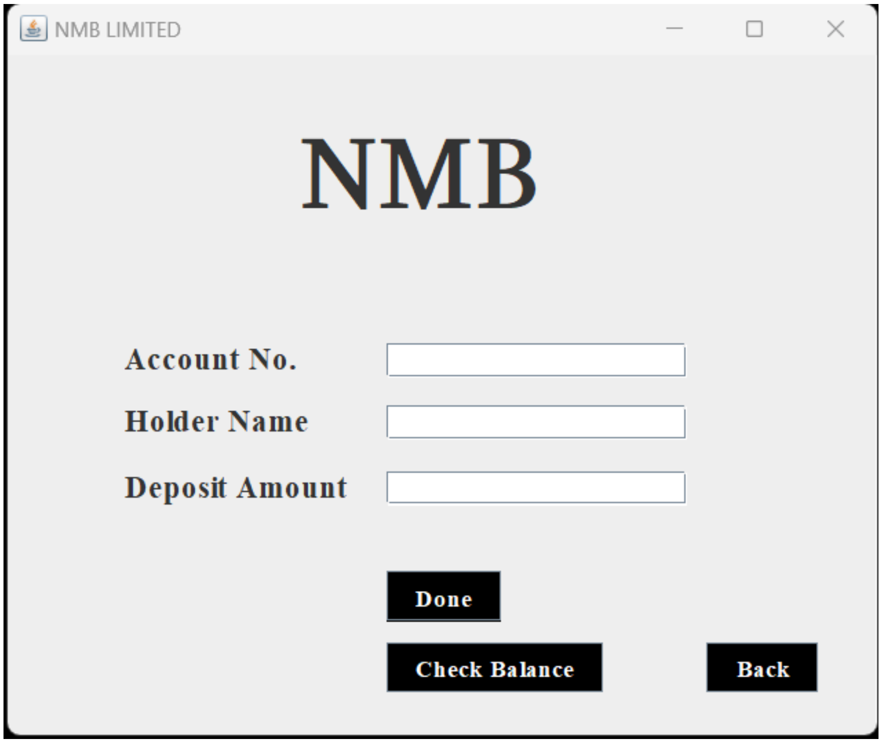
Screen 04 (Input Records)



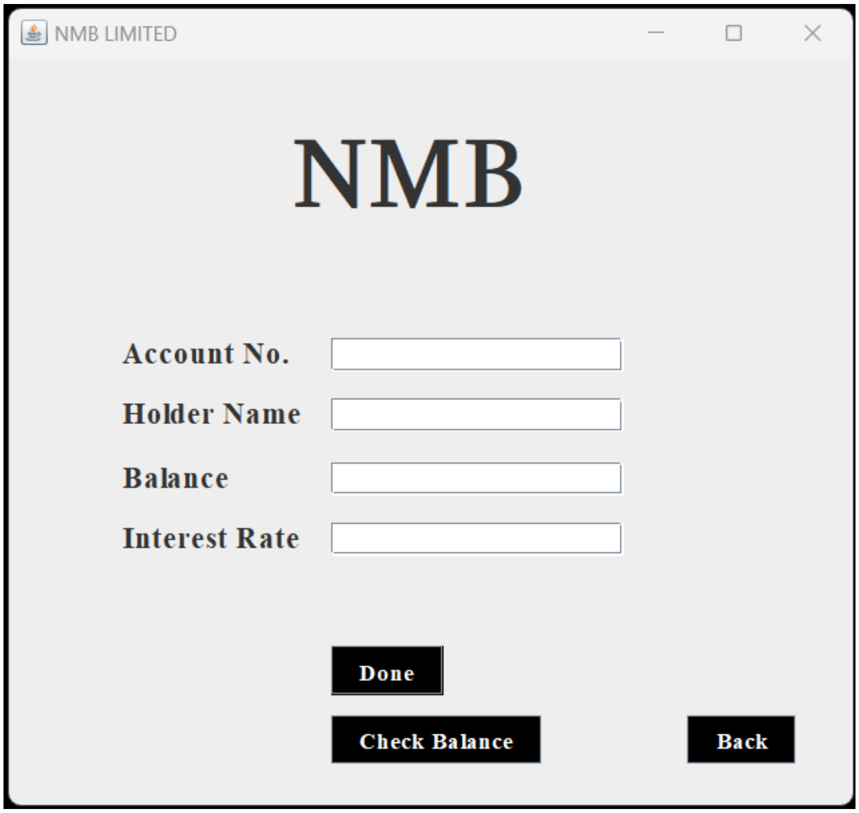
Screen 04 (Debit)



Screen 05 (Deposit)

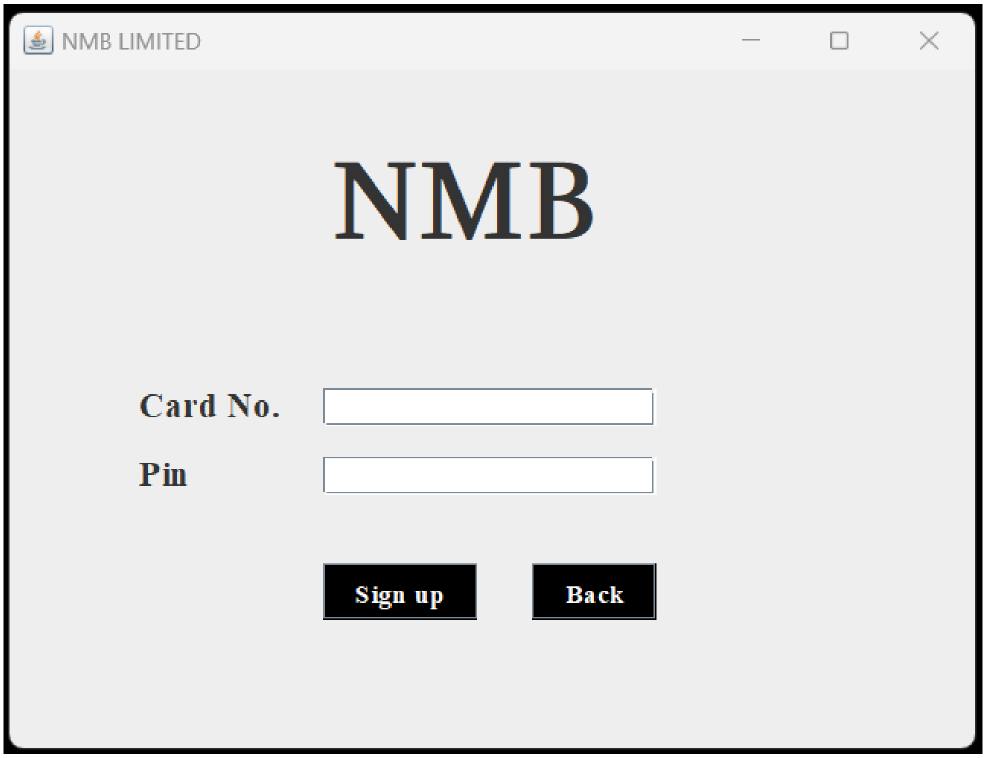


Screen 06 (Interest)

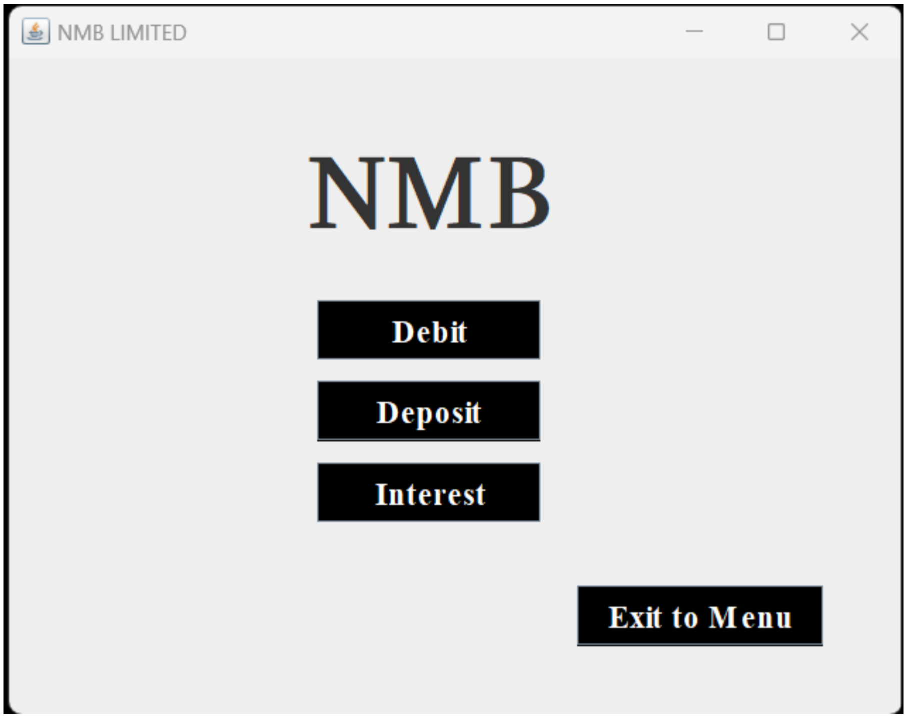


- Client

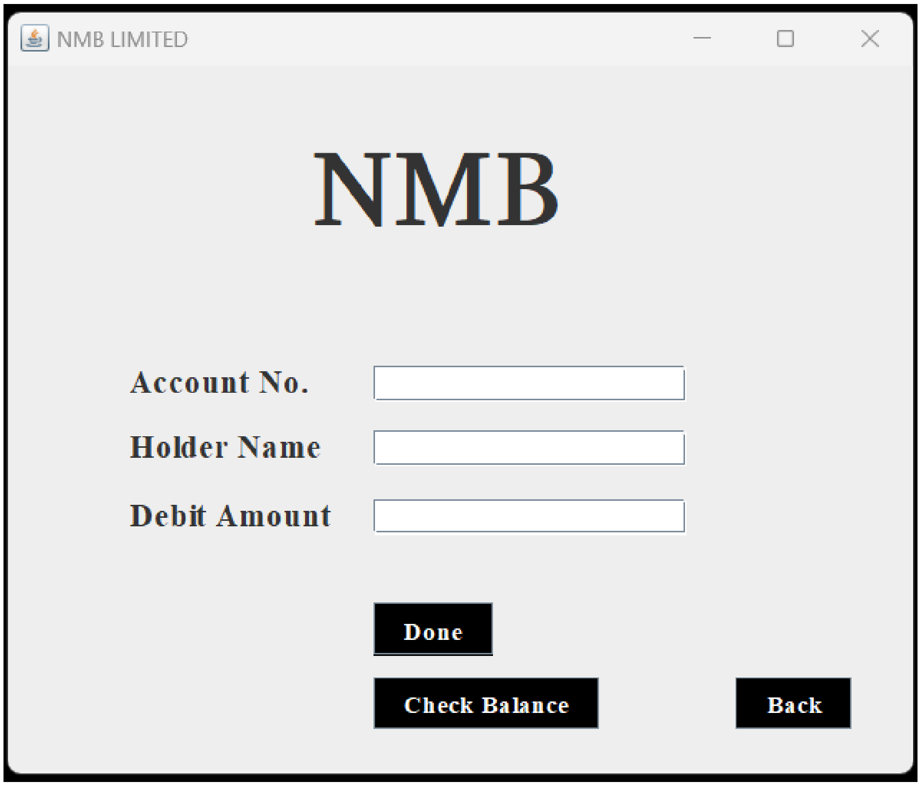
Screen 01 (Registeration)



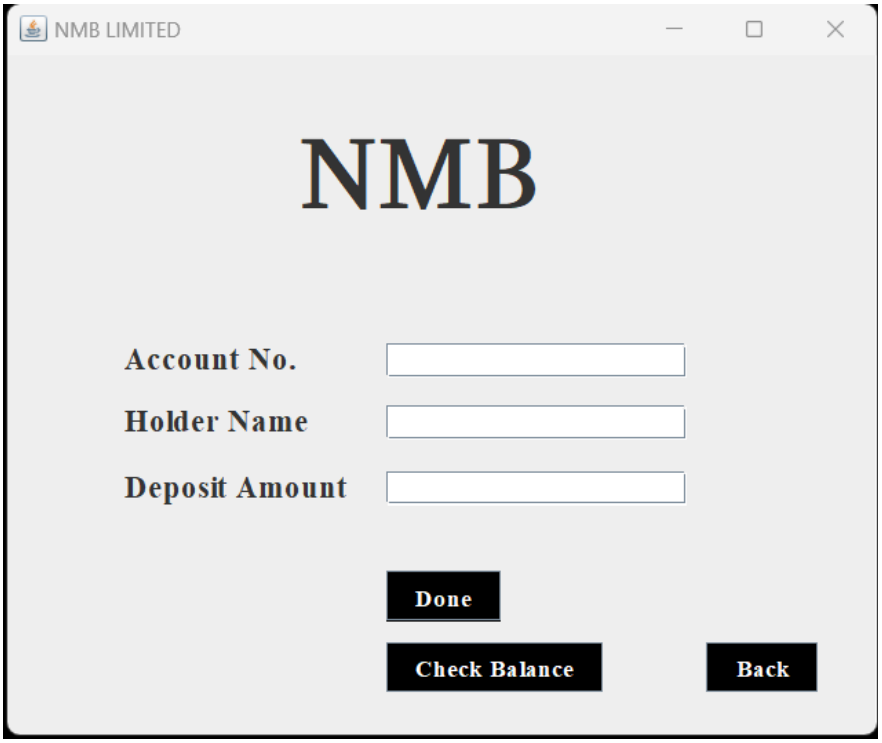
Screen 02 (Client Menu)



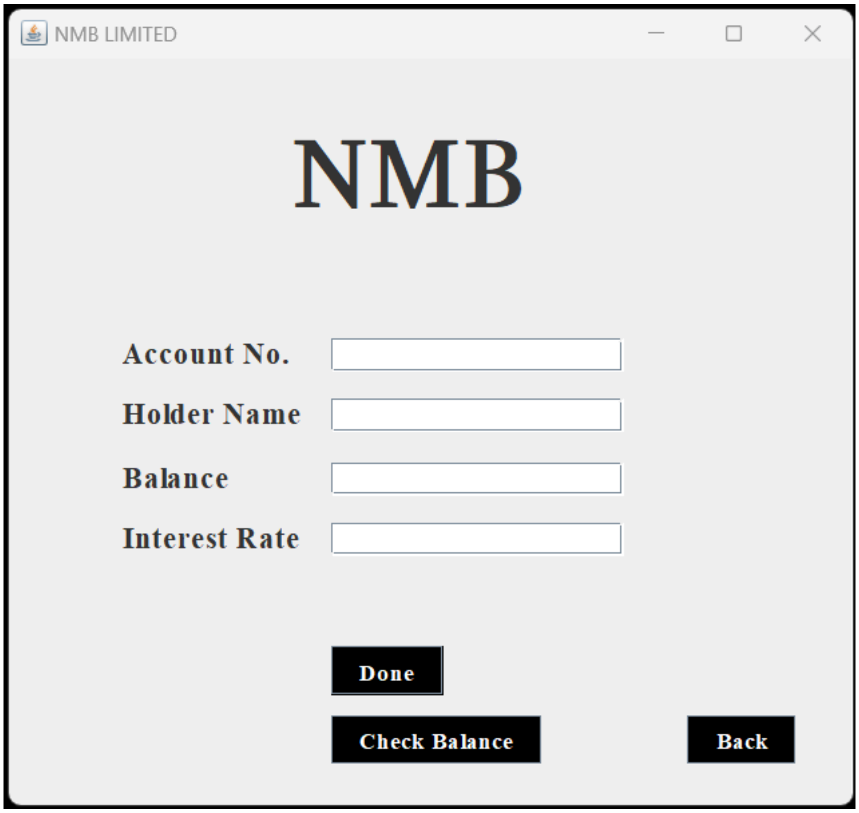
Screen 03 (Debit)



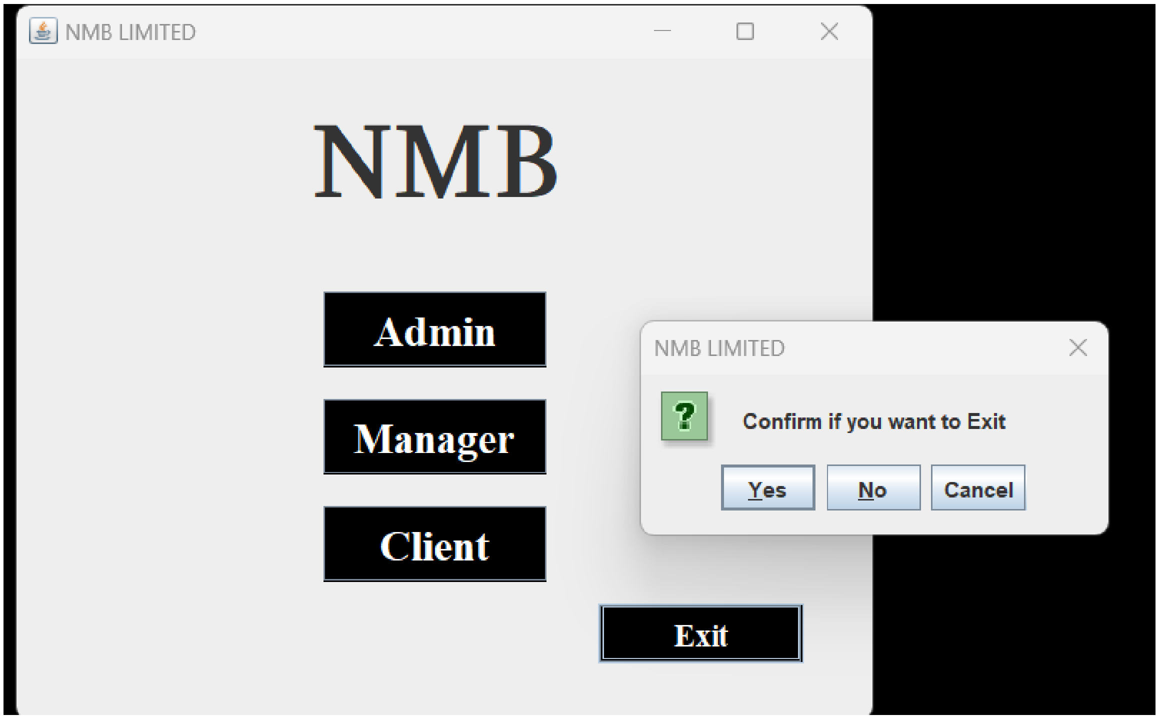
Screen 04 (Deposit)



Screen 05 (Interest)



- Exiting from GUI



Confirm if you want to Exit