**Bank Management System - Documentation**

**Introduction**

The **Bank Management System** is a simple console-based application developed in **C++**. It provides functionalities to manage bank accounts, perform transactions, and display account details. The system follows an **object-oriented approach** and uses a **linked list** to store account details dynamically.

**Features**

* **Add Account:** Allows users to create a new bank account with an initial balance.
* **Deposit Money:** Enables users to deposit an amount into an existing account.
* **Withdraw Money:** Allows users to withdraw an amount from their account if sufficient balance is available.
* **Display All Accounts:** Lists all bank accounts with account numbers, account holders, and balances.
* **Exit System:** Terminates the program.

**Technologies Used**

* Programming Language: **C++**
* Concepts Used: **OOP (Classes, Objects, Encapsulation, Inheritance), Data Structures (Linked List)**

**Class Structure**

**1. BankAccount Class**

Represents a bank account with attributes and methods to perform transactions.

**Attributes:**

* accountNumber (string): Stores the account number.
* accountHolder (string): Stores the name of the account holder.
* balance (double): Stores the current balance of the account.

**Methods:**

* BankAccount(string, string, double): Constructor to initialize account details.
* deposit(double): Adds an amount to the balance.
* withdraw(double): Deducts an amount from the balance if sufficient funds exist.
* getBalance(): Returns the account balance.
* getAccountHolder(): Returns the account holder’s name.
* getAccountNumber(): Returns the account number.

**2. Node Struct**

* Represents a node in the linked list that holds a **BankAccount object**.

**3. LinkedList Class**

Implements a linked list to store multiple bank accounts dynamically.

**Methods:**

* insert(BankAccount): Adds a new account to the list.
* search(string): Finds an account by account number.
* display(): Displays all accounts in the system.

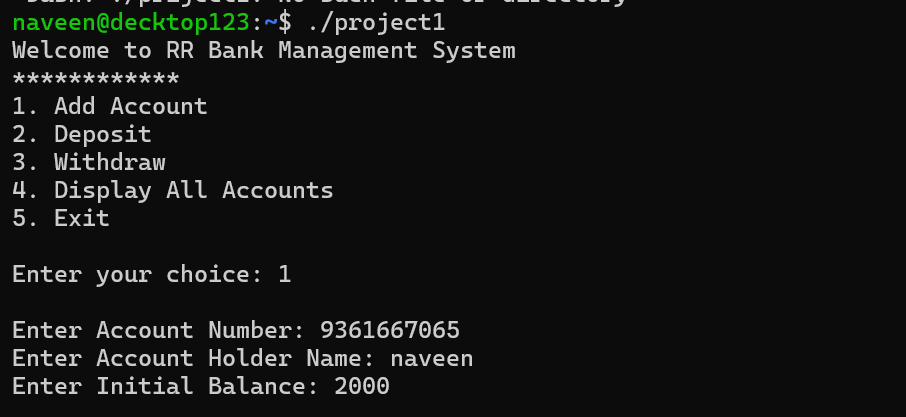
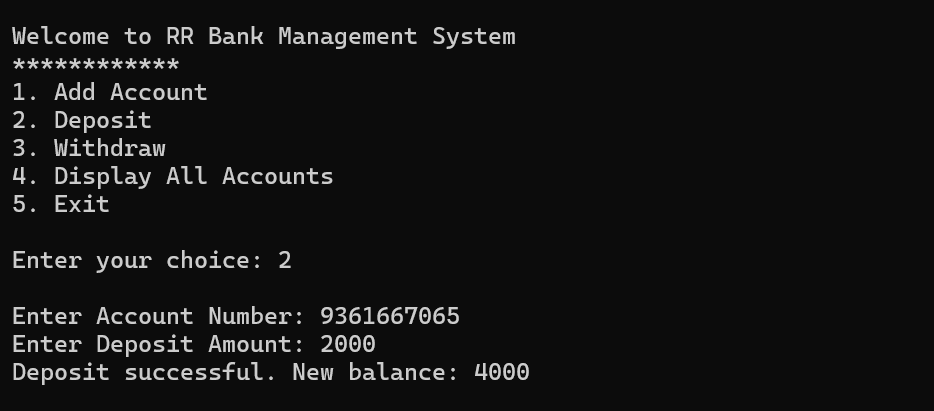
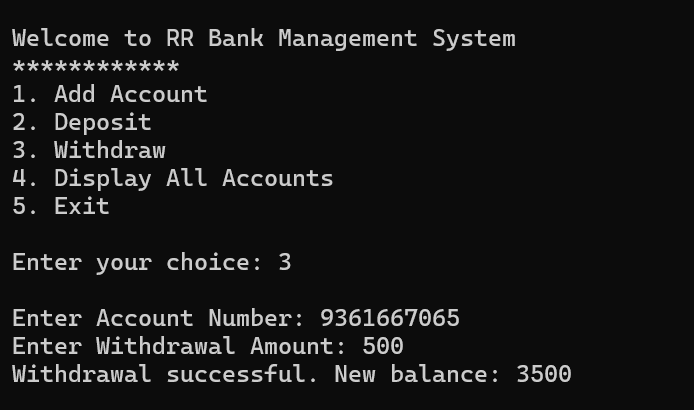
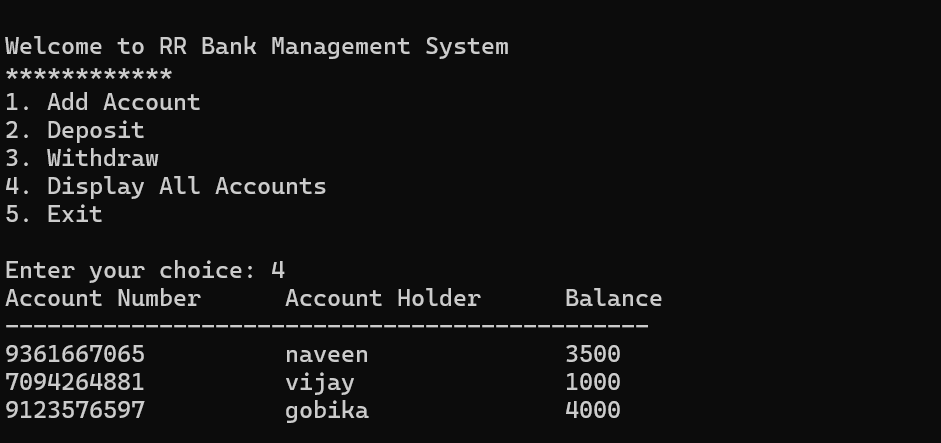
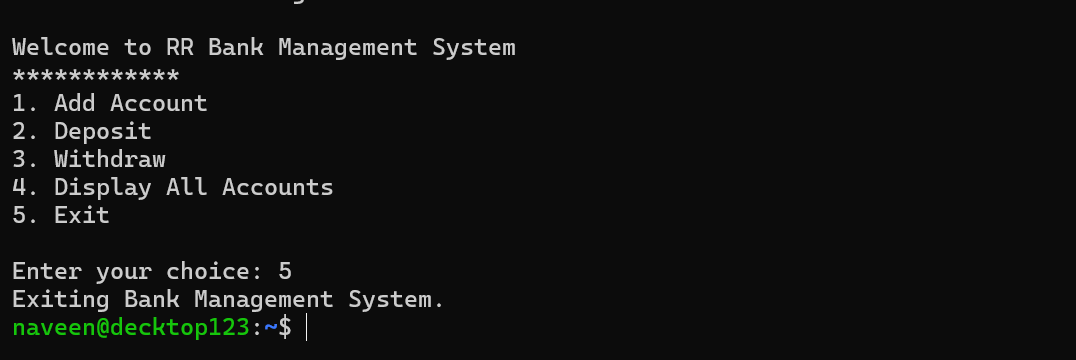
**4. Bank Class**

Manages all bank operations and interacts with the linked list.

**Methods:**

* addAccount(BankAccount): Adds a new account to the bank.
* getAccount(string): Searches and returns a pointer to a bank account.
* displayAllAccounts(): Displays all stored accounts.

**Program Flow**

1. **Main Menu:** Displays options to the user.
2. **User Choice Processing:**
   * If **Add Account** is selected, the system prompts for account details and stores them.
   * 
   * If **Deposit** or **Withdraw** is selected, it prompts for an account number and performs the respective transaction.
   * 
   * If **Display All Accounts** is selected, all stored accounts are displayed.
   * If **Exit** is selected, the program terminates.
   * 

**Example Usage**

Welcome to RR Bank Management System

\*\*\*\*\*\*\*\*\*\*\*\*

1. Add Account

2. Deposit

3. Withdraw

4. Display All Accounts

5. Exit

Enter your choice: 1

Enter Account Number: 1001

Enter Account Holder Name: John Doe

Enter Initial Balance: 5000

**Future Enhancements**

* Implement **file handling** to store account details persistently.
* Add a **login system** for security.
* Introduce **interest calculations** for savings accounts.
* Develop a **GUI version** using frameworks like Qt or Tkinter.

**Conclusion**

This project is a **basic bank management system** designed to demonstrate **object-oriented programming, linked lists, and real-world banking operations** in C++. It provides a good foundation for further enhancements and expansion into a fully functional banking application.