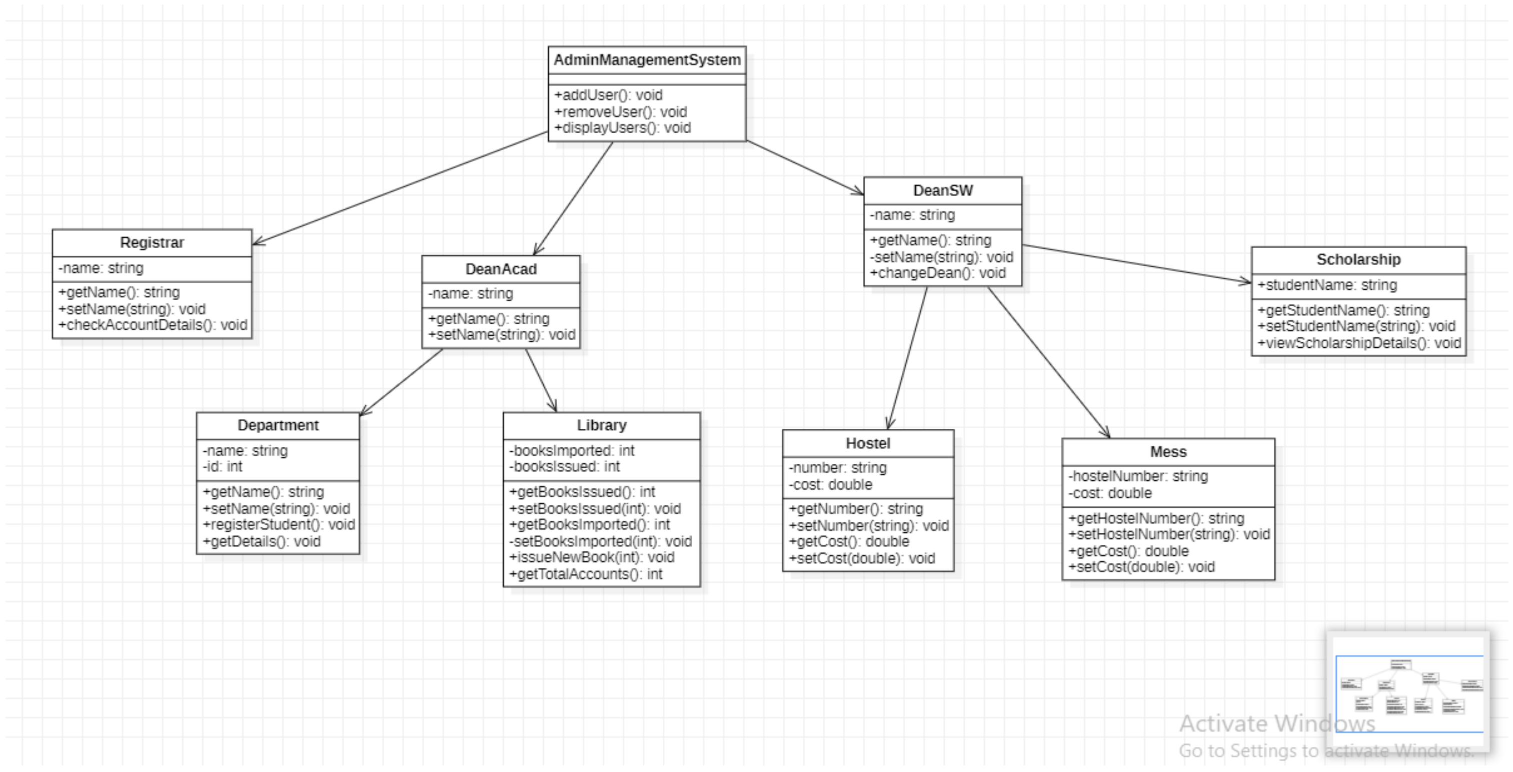
StarUml



Code

```
#include <bits/stdc++.h>
using namespace std;

class AdminManagementSystem
{
public:
   virtual void addUser(string username, string password, int id, string bankAccount) = 0;
   virtual void removeUser(int id) = 0;
   virtual void displayUsers() = 0;
   virtual ~AdminManagementSystem() {}
};

class Registrar : public AdminManagementSystem
{
```

```
private:
  struct User
  {
    int id;
    string username;
    string bankAccount;
  };

  vector<User> users;

public:
  void addUser(string username, string password, int id, string bankAccount) override
  {
    users.push_back({id, username, bankAccount});
    cout << "User '" << username << "' added to Registrar system." << endl;
  }

  void removeUser(int id) override
  {
    auto it = find_if(users.begin(), users.end(), [id](const User &user)
                      { return user.id == id; });

    if (it != users.end())
    {
      users.erase(it);
      cout << "User with ID '" << id << "' removed from Registrar system." << endl;
    }
    else
    {
      cout << "User with ID '" << id << "' not found in Registrar system." << endl;
    }
  }

  void displayUsers() override
  {
    cout << "Users in Registrar system:" << endl;
    for (const User &user : users)
    {
      cout << "ID: " << user.id << ", Username: " << user.username << ", Bank Account: " <<
user.bankAccount << endl;
    }
  }
};

class DeanAcad
```

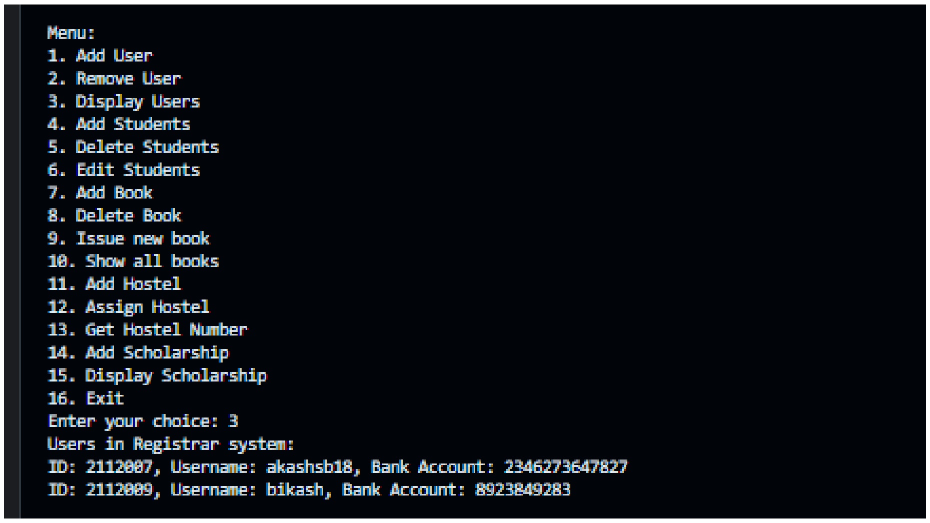
```
{
public:
    virtual void addStudent(string name, int scholarId, string phone, int hostelNum) = 0;
    virtual void deleteStudent(int scholarId) = 0;
    virtual void editStudent(int scholarId, string newName, string newPhone, int hostelNum) = 0;
    virtual ~DeanAcad() {}
};

class Department : public DeanAcad
{
private:
    struct Student
    {
        int scholarId;
        string name;
        string phone;
        int hostelNum;
    };

    vector<Student> students;

public:
    void addStudent(string name, int scholarId, string phone, int hostelNum) override
    {
        students.push_back({scholarId, name, phone, hostelNum});
        cout << "Student '" << name << "' with scholar ID '" << scholarId << "Hostel Number" << hostelNum
<< "' added to the department." << endl;
    }

    void deleteStudent(int scholarId) override
    {
        auto it = find_if(students.begin(), students.end(), [scholarId](const Student &student)
                          { return student.scholarId == scholarId; });

        if (it != students.end())
        {
            students.erase(it);
            cout << "Student with scholar ID '" << scholarId << "' deleted from the department." << endl;
        }
        else
        {
            cout << "Student with scholar ID '" << scholarId << "' not found in the department." << endl;
        }
    }
    void editStudent(int scholarId, string newName, string newPhone, int newhostelNum) override
    {
        auto it = find_if(students.begin(), students.end(), [scholarId](const Student &student)
                          { return student.scholarId == scholarId; });

        if (it != students.end())
        {
            it->name = newName;
            it->phone = newPhone;
            it->hostelNum = newhostelNum;
            cout << "Student with scholar ID '" << scholarId << "' edited successfully." << endl;
        }
        else
        {
            cout << "Student with scholar ID '" << scholarId << "' not found in the department." << endl;
        }
    }
};

class Book
{
private:
    string name;
    string author;

public:
    Book(string _name, string _author) : name(_name), author(_author) {}
    string getName() const { return name; }
    string getAuthor() const { return author; }
};

class Library
{
private:
    int booksIssued;
    int booksImported;
    vector<Book> books;

public:
    Library() : booksIssued(0), booksImported(0) {}

    int getBooksIssued() const
    {
        return booksIssued;
    }
```

```
void setBooksIssued(int numBooksIssued)
{
   booksIssued = numBooksIssued;
}

int getBooksImported() const
{
   return booksImported;
}

void setBooksImported(int numBooksImported)
{
   booksImported = numBooksImported;
}

void addBook(string name, string author)
{
   booksImported++;
   Book newBook(name, author);
   books.push_back(newBook);
   cout << "New book added successfully.\n";
}

void deleteBook()
{
   if (booksImported > 0)
   {
      booksImported--;
      cout << "Book deleted successfully.\n";
   }
   else
   {
      cout << "No books available to delete.\n";
   }
}
void displayBooks() const
{
   cout << "List of Books:\n";
   for (const auto &book : books)
   {
      cout << "Title: " << book.getName() << ", Author: " << book.getAuthor() << endl;
   }
}
void issueBook(int rollNumber)
```

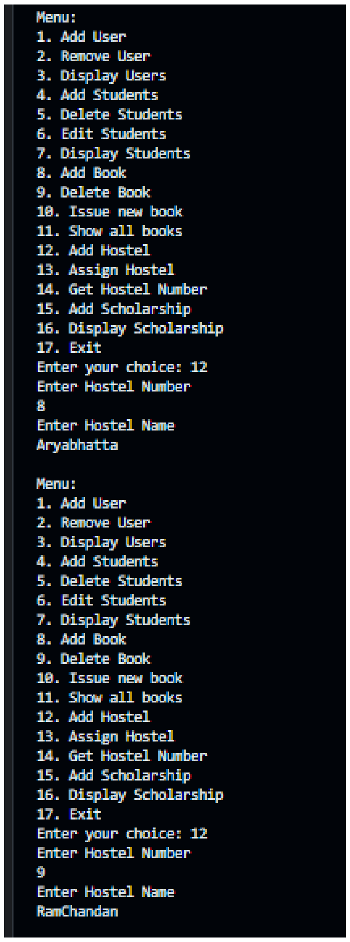
```
    {
        if (booksImported > 0)
        {
            booksIssued++;
            booksImported--;
            cout << "Book issued successfully to roll number " << rollNumber << ".\n";
        }
        else
        {
            cout << "No books available to issue.\n";
        }
    }

    int getTotalBooks() const
    {
        return books.size();
    }
};

class Hostel
{
private:
    int hostelNumber;
    string hostelName;
    unordered_map<int, int> studentHostelMap;

public:
    Hostel() : hostelNumber(0), hostelName("") {}

    void addHostel(int number, const string &name)
    {
        hostelNumber = number;
        hostelName = name;
    }

    void assignHostel(int scholarId, int hostelNum)
    {
        studentHostelMap[scholarId] = hostelNum;
        cout << "Scholar ID :" << scholarId << " "
             << "is assigned to Hostel Number " << hostelNum << endl;
    }

    int getHostelNumber(int scholarId) const
    {
        auto it = studentHostelMap.find(scholarId);
        if (it != studentHostelMap.end())
        {
            return it->second;
        }
        else
        {
            return -1;
        }
    }
};

class Event
{
private:
    string eventName;
    string eventDate;
    vector<string> attendees;

public:
    Event(string name, string date) : eventName(name), eventDate(date) {}

    void addAttendee(const string &attendee)
    {
        attendees.push_back(attendee);
        cout << "Attendee '" << attendee << "' added to the event '" << eventName << "'." << endl;
    }

    void removeAttendee(const string &attendee)
    {
        auto it = find(attendees.begin(), attendees.end(), attendee);
        if (it != attendees.end())
        {
            attendees.erase(it);
            cout << "Attendee '" << attendee << "' removed from the event '" << eventName << "'." << endl;
        }
        else
        {
            cout << "Attendee '" << attendee << "' is not registered for the event '" << eventName << "'." <<
endl;
        }
    }

    void displayAttendees() const
    {
        cout << "Attendees for the event '" << eventName << "':" << endl;
        for (const auto &attendee : attendees)
        {
            cout << "- " << attendee << endl;
        }
    }
};

class Scholarship
{
private:
    string name;
    double amount;
    string eligibilityCriteria;

public:
    Scholarship(string _name, double _amount, string _eligibilityCriteria)
        : name(_name), amount(_amount), eligibilityCriteria(_eligibilityCriteria) {}

    string getName() const { return name; }
    double getAmount() const { return amount; }
    string getEligibilityCriteria() const { return eligibilityCriteria; }

    void displayDetails() const
    {
        cout << "Scholarship Name: " << name << endl;
        cout << "Amount: " << amount << endl;
        cout << "Eligibility Criteria: " << eligibilityCriteria << endl;
    }
};

void displayMenu()
{
    cout << "\nMenu:\n";
    cout << "1. Add User\n";
    cout << "2. Remove User\n";
    cout << "3. Display Users\n";
    cout << "4. Add Students\n";
    cout << "5. Delete Students\n";
    cout << "6. Edit Students\n";
    cout << "7. Add Book\n";
    cout << "8. Delete Book\n";
    cout << "9. Issue new book\n";
    cout << "10. Show all books\n";
    cout << "11. Add Hostel\n";
    cout << "12. Assign Hostel\n";
    cout << "13. Get Hostel Number\n";
    cout << "14. Add Scholarship\n";
    cout << "15. Display Scholarship\n";
    cout << "16. Exit\n";
    cout << "Enter your choice: ";
}

int main()
{
    Registrar registrar;
    Department department;
    Library library;
    Hostel hostel;

    int choice, hostelNum;
    string username, password, bankAccount;
    string name, phone, newName, newPhone;
    string book, author, hostelName;
    vector<Scholarship> scholarships;
    int scholarId;

    int id;

    do
    {
        displayMenu();
        cin >> choice;

        switch (choice)
        {
        case 1:
            cout << "Enter username: ";
            cin >> username;
            cout << "Enter password: ";
            cin >> password;
            cout << "Enter ID: ";
            cin >> id;
            cout << "Enter bank account number: ";
            cin >> bankAccount;
            registrar.addUser(username, password, id, bankAccount);
            break;
        case 2:
            cout << "Enter ID to remove: ";
            cin >> id;
            registrar.removeUser(id);
            break;
        case 3:
            registrar.displayUsers();
            break;
        case 4:
            cout << "Enter student name: ";
            cin >> name;
            cout << "Enter scholar ID: ";
            cin >> scholarId;
            cout << "Enter phone number: ";
            cin >> phone;
            cout << "Enter Hostel Number" << endl;
            cin >> hostelNum;
            department.addStudent(name, scholarId, phone, hostelNum);
            break;
        case 5:
            cout << "Enter scholar ID to delete: ";
            cin >> scholarId;
            department.deleteStudent(scholarId);
            break;
        case 6:
            cout << "Enter scholar ID to edit: ";
            cin >> scholarId;
            cout << "Enter new name: ";
            cin >> newName;
            cout << "Enter new phone number: ";
            cin >> newPhone;
            cout << "Enter Hostel Number" << endl;
            cin >> hostelNum;
            department.editStudent(scholarId, newName, newPhone, hostelNum);
            break;
        case 7:
            cout << "Enter Book Name" << endl;
            cin >> name;
            cout << "Enter author Name" << endl;
            cin >> author;
            library.addBook(name, author);
            break;
        case 8:
            library.deleteBook();
            break;
        case 9:
            int roll;
            cout << "Enter Scholar ID" << endl;
            cin >> roll;
        library.issueBook(roll);
        break;
case 10:
    library.displayBooks();
    break;
case 11:
    cout << "Enter Hostel Number" << endl;
    cin >> hostelNum;
    cout << "Enter Hostel Name" << endl;
    cin >> hostelName;
    hostel.addHostel(hostelNum, hostelName);
    break;
case 12:
    cout << "Enter Scholar ID" << endl;
    cin >> scholarId;
    cout << "Enter Hostel Number" << endl;
    cin >> hostelNum;
    hostel.assignHostel(scholarId, hostelNum);
    break;
case 13:
    cout << "Enter Scholar ID" << endl;
    hostel.getHostelNumber(scholarId);
    break;
case 14:
{
    string scholarshipName, eligibilityCriteria; // Initialize inside the case
    double scholarshipAmount;
    cout << "Enter Scholarship Name: ";
    cin >> scholarshipName;
    cout << "Enter Scholarship Amount: ";
    cin >> scholarshipAmount;
    cout << "Enter Eligibility Criteria: ";
    cin.ignore(); // Clear buffer
    getline(cin, eligibilityCriteria);
    scholarships.push_back(Scholarship(scholarshipName, scholarshipAmount, eligibilityCriteria));
    cout << "Scholarship added successfully.\n";
    break;
}

case 15:
    cout << "Displaying Scholarships:\n";
    for (const auto &scholarship : scholarships)
    {
        scholarship.displayDetails();
        cout << endl;
```

```
            }
            break;

        case 16:
            cout << "Exiting program.\n";
            break;
        default:
            cout << "Invalid choice. Please try again.\n";
        }
    } while (choice != 16);

    return 0;}
```

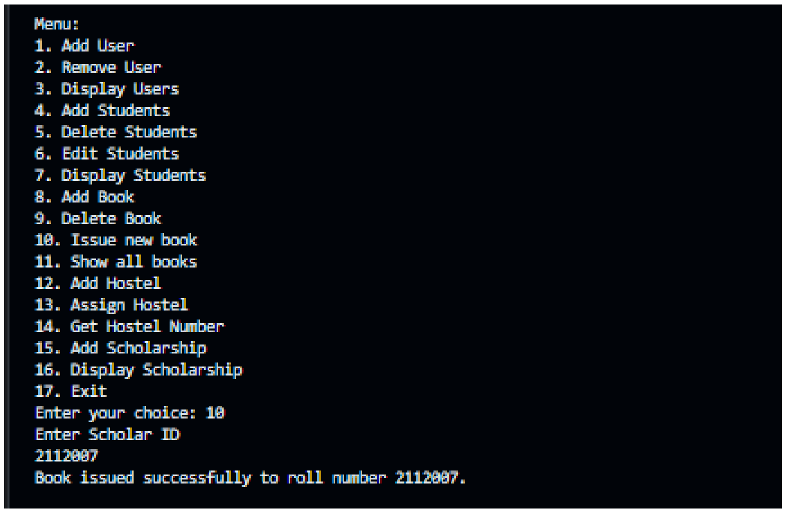
OUTPUT



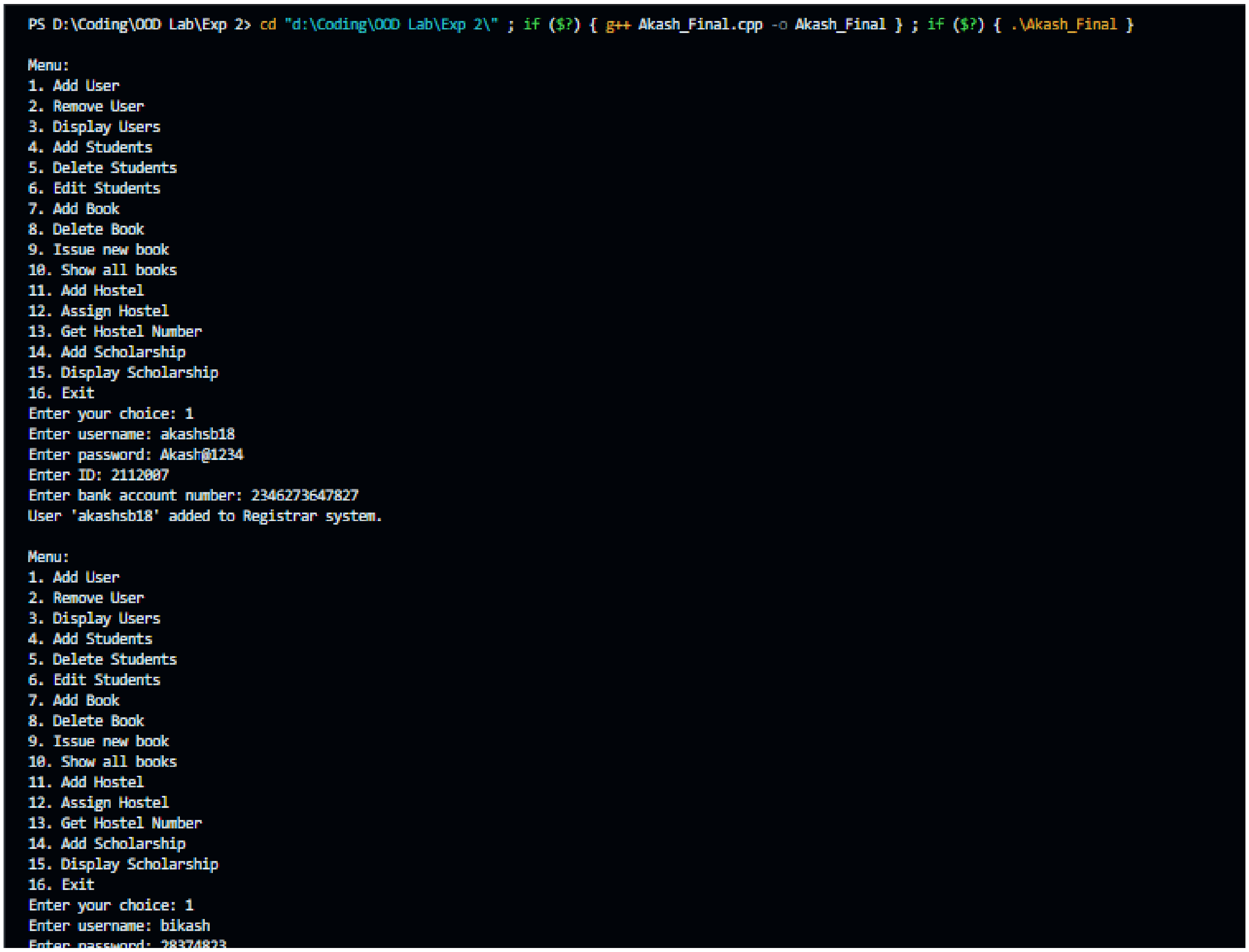
```
Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Add Book
8. Delete Book
9. Issue new book
10. Show all books
11. Add Hostel
12. Assign Hostel
13. Get Hostel Number
14. Add Scholarship
15. Display Scholarship
16. Exit
Enter your choice: 3
Users in Registrar system:
ID: 2112007, Username: akashsb18, Bank Account: 2346273647827
ID: 2112009, Username: bikash, Bank Account: 8923849283
```



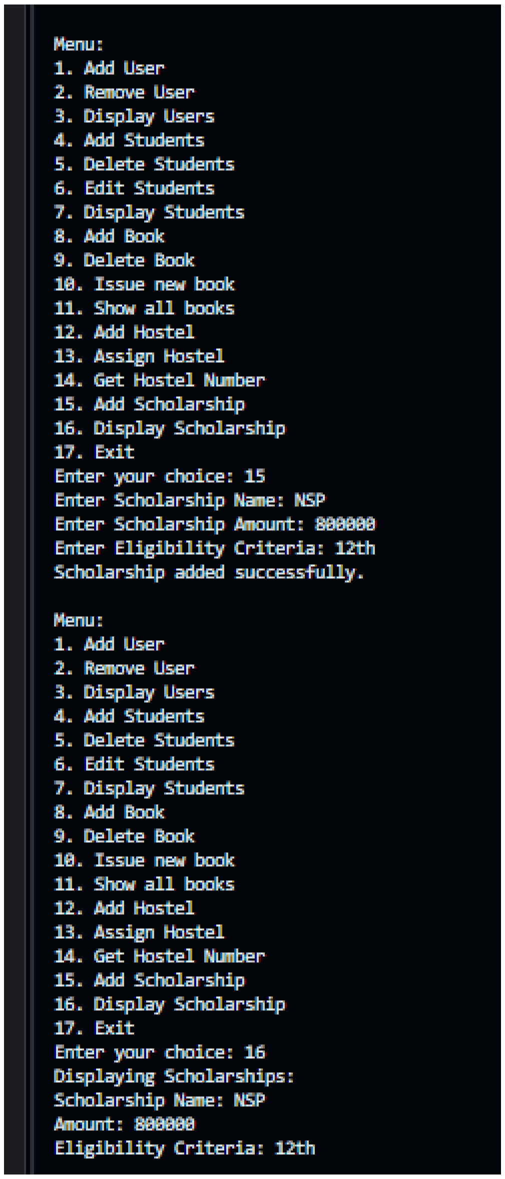
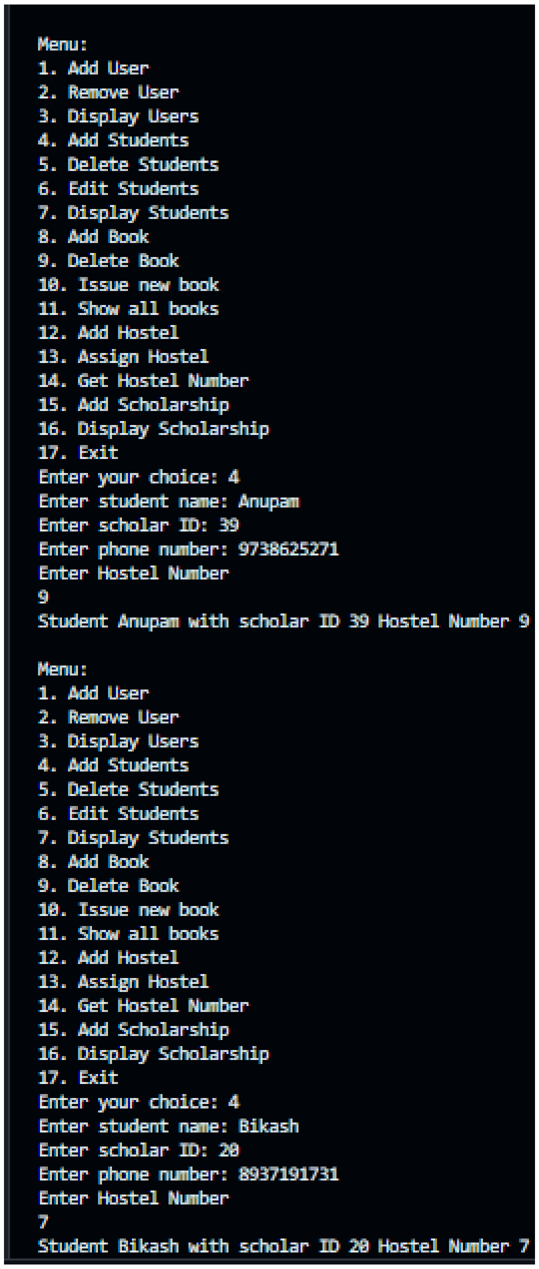
```
Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Display Students
8. Add Book
9. Delete Book
10. Issue new book
11. Show all books
12. Add Hostel
13. Assign Hostel
14. Get Hostel Number
15. Add Scholarship
16. Display Scholarship
17. Exit
Enter your choice: 12
Enter Hostel Number
8
Enter Hostel Name
Aryabhatta

Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Display Students
8. Add Book
9. Delete Book
10. Issue new book
11. Show all books
12. Add Hostel
13. Assign Hostel
14. Get Hostel Number
15. Add Scholarship
16. Display Scholarship
17. Exit
Enter your choice: 12
Enter Hostel Number
9
Enter Hostel Name
RamChandan
```



```
Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Display Students
8. Add Book
9. Delete Book
10. Issue new book
11. Show all books
12. Add Hostel
13. Assign Hostel
14. Get Hostel Number
15. Add Scholarship
16. Display Scholarship
17. Exit
Enter your choice: 10
Enter Scholar ID
2112007
Book issued successfully to roll number 2112007.
```



```
PS D:\Coding\OOD Lab\Exp 2> cd "d:\Coding\OOD Lab\Exp 2\" ; if ($?) { g++ Akash_Final.cpp -o Akash_Final } ; if ($?) { .\Akash_Final }

Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Add Book
8. Delete Book
9. Issue new book
10. Show all books
11. Add Hostel
12. Assign Hostel
13. Get Hostel Number
14. Add Scholarship
15. Display Scholarship
16. Exit
Enter your choice: 1
Enter username: akashsb18
Enter password: Akash@1234
Enter ID: 2112007
Enter bank account number: 2346273647827
User 'akashsb18' added to Registrar system.

Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Add Book
8. Delete Book
9. Issue new book
10. Show all books
11. Add Hostel
12. Assign Hostel
13. Get Hostel Number
14. Add Scholarship
15. Display Scholarship
16. Exit
Enter your choice: 1
Enter username: bikash
Enter password: 28374823
```


```
Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Display Students
8. Add Book
9. Delete Book
10. Issue new book
11. Show all books
12. Add Hostel
13. Assign Hostel
14. Get Hostel Number
15. Add Scholarship
16. Display Scholarship
17. Exit
Enter your choice: 15
Enter Scholarship Name: NSP
Enter Scholarship Amount: 800000
Enter Eligibility Criteria: 12th
Scholarship added successfully.

Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Display Students
8. Add Book
9. Delete Book
10. Issue new book
11. Show all books
12. Add Hostel
13. Assign Hostel
14. Get Hostel Number
15. Add Scholarship
16. Display Scholarship
17. Exit
Enter your choice: 16
Displaying Scholarships:
Scholarship Name: NSP
Amount: 800000
Eligibility Criteria: 12th
```

```
Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Display Students
8. Add Book
9. Delete Book
10. Issue new book
11. Show all books
12. Add Hostel
13. Assign Hostel
14. Get Hostel Number
15. Add Scholarship
16. Display Scholarship
17. Exit
Enter your choice: 4
Enter student name: Anupam
Enter scholar ID: 39
Enter phone number: 9738625271
Enter Hostel Number
9
Student Anupam with scholar ID 39 Hostel Number 9

Menu:
1. Add User
2. Remove User
3. Display Users
4. Add Students
5. Delete Students
6. Edit Students
7. Display Students
8. Add Book
9. Delete Book
10. Issue new book
11. Show all books
12. Add Hostel
13. Assign Hostel
14. Get Hostel Number
15. Add Scholarship
16. Display Scholarship
17. Exit
Enter your choice: 4
Enter student name: Bikash
Enter scholar ID: 20
Enter phone number: 8937191731
Enter Hostel Number
7
Student Bikash with scholar ID 20 Hostel Number 7
```