Library Management System

Group No. = 4

Presented By: - Ankitraj Jha, Sejal Singh, Parthiv Kumar, Sairaj Jadhav

—---

Introduction to the Case Study:-

This case study is based on the development of a Library Management System using C++.
 Libraries handle many books every day, and managing them manually can be slow and confusing.
 This system helps to manage book records like adding new books, issuing books, returning books, and searching books easily.
 The project is designed as a console-based application and focuses on clarity, correctness, and ease of use.

Case Background (Abstract):-

Managing a library manually requires maintaining registers for book details, issue records, and returns.
 This process is time-consuming and may lead to errors such as duplicate entries or incorrect issue status.

The objective of this case study is to design and implement a Library Management System in C++ that automates these tasks.
 The system provides an efficient way to store book information and perform basic library operations using programming concepts.

Case Study Design:-

The system is designed to work in a menu-driven format.
 Each operation of the library is divided into separate modules such as adding books, issuing books, returning books, and searching books.

“An array of `Book` structures is used to store book records (max 100 books).”
This is more accurate for your code.
 The design ensures that each book has a unique ID and a clear issue status.

Library Management System

Methods & Algorithms Technology Applied in the Problem Statement The following methods and technologies are applied:

- C++ Programming Language for implementation
- Structure to store book details
- Linear Search Algorithm to find books by ID or keyword
- Conditional Statements to check book status
- Loops to display and manage records
- Input Validation to handle invalid user input

These methods ensure the system works efficiently and correctly.

Case Study Implementation Details and Snapshots

The implementation consists of multiple functions, each performing a specific task:

- addBook() – Adds a new book to the library
- issueBook() – Issues a book if available
- returnBook() – Returns an issued book
- searchBook() – Searches books by title or author
- deleteBook() – Deletes a book record
- displayBooks() – Displays all books with status

The program starts execution from the main() function, where the menu is displayed repeatedly until the user exits.

Code Snippets:

1. Structure Declaration (Book Record)

This structure stores all book information.

Library Management System

```
#include<iostream>
#include<string>
using namespace std;

struct Book {
    int id;
    string title;
    string author;
    bool issued;
};

Book library[100];
int totalBooks = 0;
```

2. Add Book Function

Adds a new book into the library.

```
void addBook() {
    cout << "Enter Book ID: ";
    cin >> library[totalBooks].id;

    cin.ignore();
    cout << "Enter Title: ";
    getline(cin, library[totalBooks].title);

    cout << "Enter Author: ";
    getline(cin, library[totalBooks].author);

    library[totalBooks].issued = false;
    totalBooks++;

    cout << "Book added successfully!\n";
}
```

3. Issue Book Function

Issues a book if available

```
void issueBook(int id) {
    for(int i=0;i<totalBooks;i++) {
        if(library[i].id == id && !library[i].issued) {
            library[i].issued = true;
            cout<<"Book Issued Successfully\n";
            return;
        }
    }
    cout<<"Book not available\n";
}
```

4. Return Book Function

Marks an issued book as returned.

```
void returnBook(int id) {
    for(int i=0;i<totalBooks;i++) {
        if(library[i].id == id && library[i].issued) {
            library[i].issued = false;
            cout<<"Book Returned Successfully\n";
            return;
        }
    }
    cout<<"Invalid Book ID\n";
}
```

5. Search Book Function

Searches a book using ID.

```
void searchBook(int id) {
    for(int i=0;i<totalBooks;i++) {
        if(library[i].id == id) {
            cout<<"Title: "<<library[i].title<<endl;
            cout<<"Author: "<<library[i].author<<endl;
            cout<<"Status: "
                <<(library[i].issued ? "Issued":"Available")<<endl;
            return;
        }
    }
    cout<<"Book not found\n";
}
```

6. Display Books Function

Shows all book records.

```
void displayBooks() {
    for(int i=0;i<totalBooks;i++) {
        cout<<"ID: "<<library[i].id<<endl;
        cout<<"Title: "<<library[i].title<<endl;
        cout<<"Author: "<<library[i].author<<endl;
        cout<<"Status: "
            <<(library[i].issued ? "Issued":"Available")<<endl;
        cout<<"------------------"<<endl;
    }
}
```

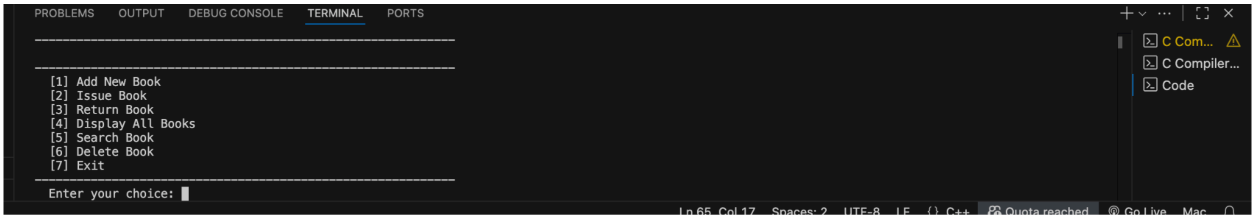
7. Main Function (Menu Driven Program)

Program execution starts here.

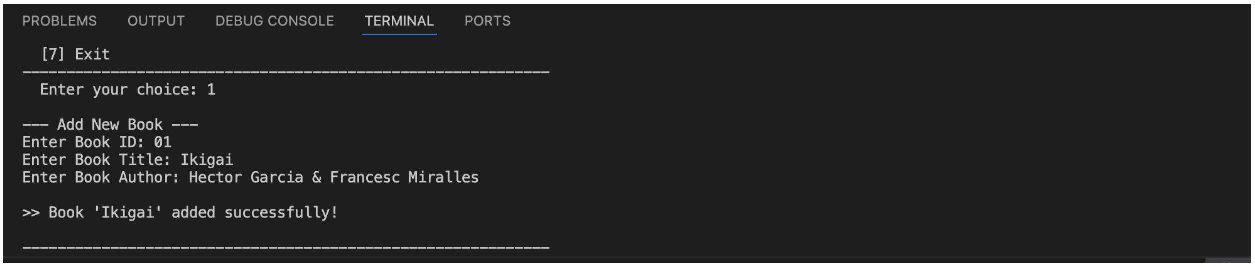
```
int main() {
    int choice, id;

    do {
        cout<<"\n1.Add Book\n2.Issue Book\n3.Return Book\n4.Search Book\n5.Display Books\n6.Exit\n";
        cin>>choice;

        switch(choice) {
            case 1: addBook(); break;
            case 2: cout<<"Enter ID: "; cin>>id; issueBook(id); break;
            case 3: cout<<"Enter ID: "; cin>>id; returnBook(id); break;
            case 4: cout<<"Enter ID: "; cin>>id; searchBook(id); break;
            case 5: displayBooks(); break;
        }
    } while(choice!=6);

    return 0;
}
```

Snapshots:

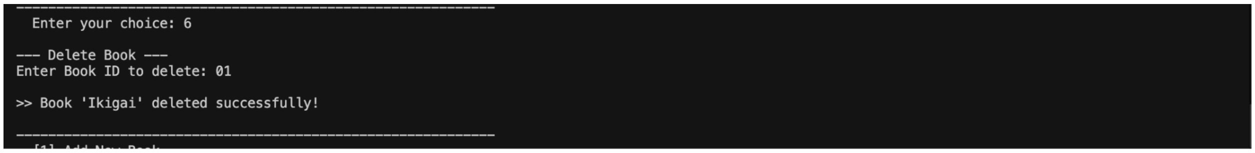
Library Management System



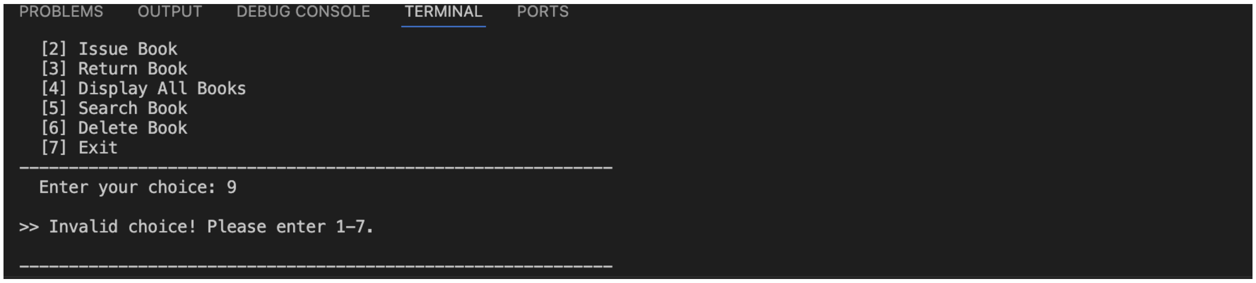
```
------------------------------------------------------------

------------------------------------------------------------
  [1] Add New Book
  [2] Issue Book
  [3] Return Book
  [4] Display All Books
  [5] Search Book
  [6] Delete Book
  [7] Exit
------------------------------------------------------------
  Enter your choice: █
```



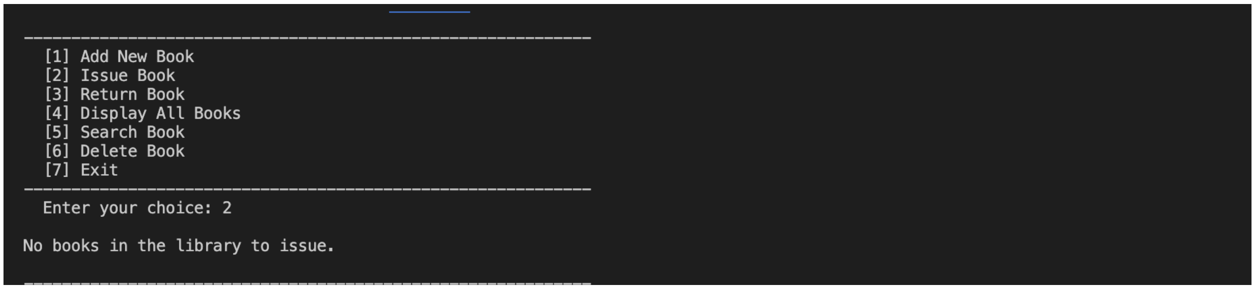
```
  [7] Exit
------------------------------------------------------------
  Enter your choice: 1

--- Add New Book ---
Enter Book ID: 01
Enter Book Title: Ikigai
Enter Book Author: Hector Garcia & Francesc Miralles

>> Book 'Ikigai' added successfully!

------------------------------------------------------------
```



```
------------------------------------------------------------
  Enter your choice: 6

--- Delete Book ---
Enter Book ID to delete: 01

>> Book 'Ikigai' deleted successfully!

------------------------------------------------------------
  [1] Add New Book
```



```
  [2] Issue Book
  [3] Return Book
  [4] Display All Books
  [5] Search Book
  [6] Delete Book
  [7] Exit
------------------------------------------------------------
  Enter your choice: 9

>> Invalid choice! Please enter 1-7.

------------------------------------------------------------
```



```
------------------------------------------------------------
  [1] Add New Book
  [2] Issue Book
  [3] Return Book
  [4] Display All Books
  [5] Search Book
  [6] Delete Book
  [7] Exit
------------------------------------------------------------
  Enter your choice: 2

No books in the library to issue.

------------------------------------------------------------
```

🔹 Limitations & Future Improvements:

The current system stores a maximum of 100 books and does not use file handling for permanent data storage. In future versions, file handling and database integration can be added to make the system more scalable and practical for large libraries.

Case Study Results and Conclusion

The Library Management System successfully performs all intended operations.
It allows users to manage books efficiently and avoids common manual errors.

The case study demonstrates how basic C++ concepts can be applied to solve real-world problems.
The system is suitable for small libraries and academic learning purposes.

References:-

1. Bjarne Stroustrup, *The C++ Programming Language*
2. E. Balagurusamy, *Object Oriented Programming with C++*
3. cppreference.com – C++ Standard Library Documentation
4. W3School