Simple Library Management System

Board Infinity- Data Structure and Algorithm

A training report

Submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology

(Computer Science and Engineering)

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

From 05/06/2023 to 10/07/2023

SUBMITTED BY

Name of student: Praveen Kumar Tiwari

Registration Number: 12104582

Student Declaration

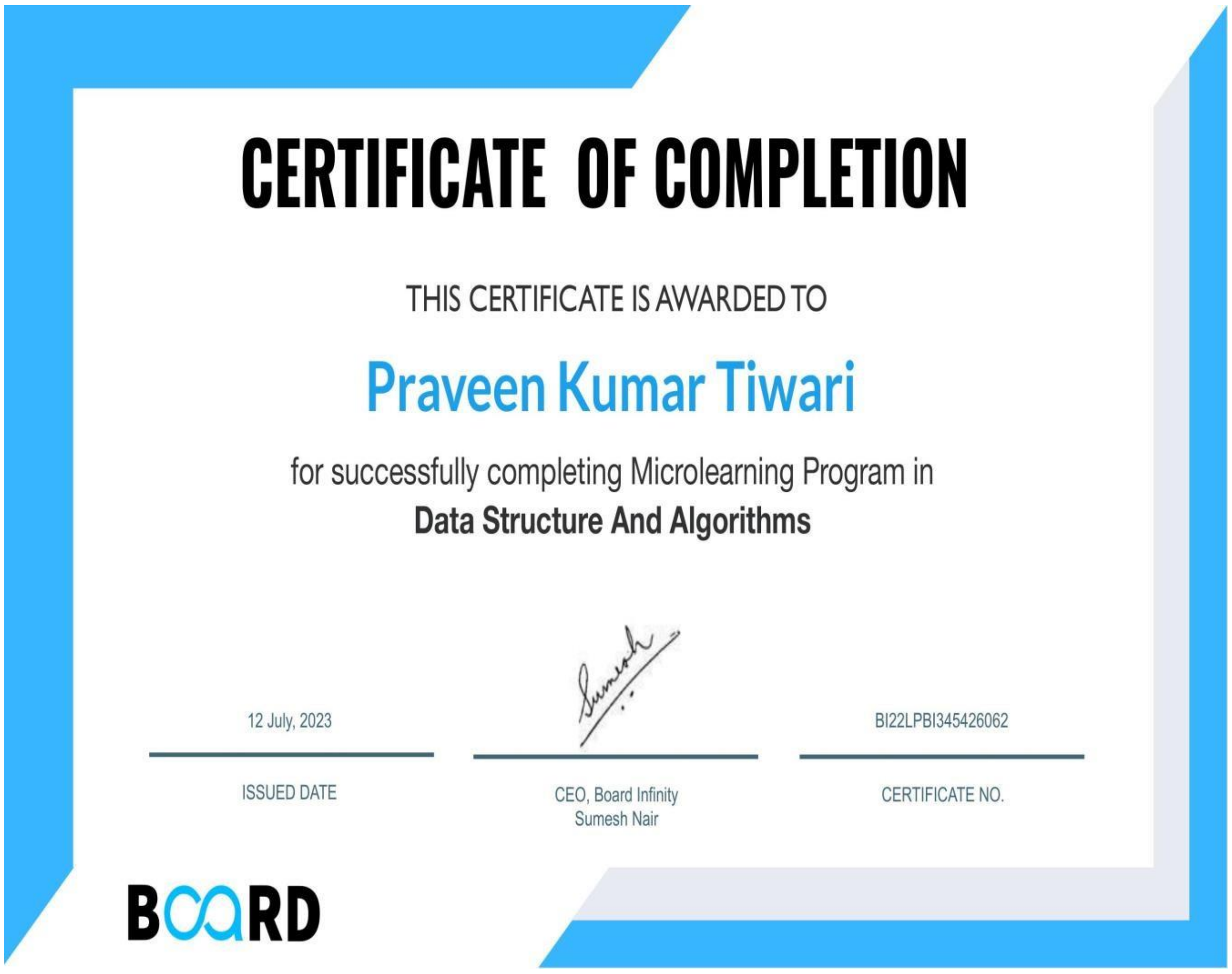
I, **Praveen Kumar Tiwari, 12104582**, hereby declare that the work done by me on "**Simple Library management System**" from **June 2023 to July 2023**, is a record of original work for the partial fulfillment of the requirements for the award of the degree, **Bachelor of Technology in Computer Science and Engineering.**

Name of the Student (Registration Number): Praveen Kumar Tiwari (12104582)

Dated: 12/08/2023

Table of Contents

<u>Training certificate from Board Infinity</u>



CERTIFICATE OF COMPLETION

THIS CERTIFICATE IS AWARDED TO

Praveen Kumar Tiwari

for successfully completing Microlearning Program in

Data Structure And Algorithms

12 July, 2023	[illegible]	BI22LPBI345426062
ISSUED DATE	CEO, Board Infinity Sumesh Nair	CERTIFICATE NO.

BOARD

Completion Certificate

By Board Infinitiy

INTRODUCTION

The Simple Library Management System project aims to develop a user-friendly software application using C++. The system is designed to assist librarians in efficiently managing books and tracking their availability. It offers various features such as adding new books, searching for books, issuing books to students, returning books, listing all books, and deleting books from the system.

This project provides an excellent opportunity for students to apply their knowledge of data structures, including arrays, linked lists, stacks, and queues, in a practical scenario. Additionally, students will gain hands-on experience in implementing essential algorithms for searching and sorting, such as binary search, linear search, quick sort, and merge sort.

Features:

1. Add New Books:

The librarian can easily add new books to the system by providing details like unique ID, title, author, and status (available or issued). The book data can be stored using an array or linked list data structure.

2. Search for a Book:

The system allows the librarian to search for a book by its title or ID. A search algorithm, such as binary search for sorted books or linear search for unsorted books, can be implemented to find the desired book. If found, the system displays the book's details.

3. Issue a Book:

When a book is issued to a student, the system updates its status from available to issued. It also stores the relevant details of the student to whom the book has been issued. To manage book issues efficiently, a stack or queue data structure can be utilized.

4. Return a Book:

When a student returns a book, the system updates its status from issued to available. The system also removes the details of the student to whom the book was issued, ensuring accurate book tracking.

5. List All Books:

The librarian can view a comprehensive list of all books in the library. To present the books in a sorted manner, a sorting algorithm like quick sort or merge sort can be implemented based on criteria such as ID or title.

6. Delete a Book:

The librarian has the ability to delete a book from the system. If a linked list data structure is employed for storing books, removing a node from the linked list will accomplish this task.

"With the advancement of technology, it is imperative to exalt all the systems into a user-friendly manner. The Library Management system (LMS) acts as a tool to transform traditional libraries into digital libraries". In traditional libraries, the students/user has to search for books which are hassle process and there is no proper maintenance of database about issues/fines. The overall progress of work is slow and it is impossible to generate a fast report. The librarians have to work allotted for arranging, sorting books in the book sells. At the same time, they have to check and monitor the lend/borrow book details with its fine. It is a tedious process to work simultaneously in different sectors. LMS will assist the librarians to work easily. The LMS supports the librarians to encounter all the issues concurrently. The users need not stand in a queue for a long period to return/borrow a book from the library. The single PC contains all the data's in it. The librarians have to assess the system and provide an entry in it. Through LMS the librarian can find the book in the bookshelves. The LMS is designed with the basic features such as librarian can add/view/update/delete books and students' details in it. Once he/she ingress into the system they can modify any data's in the database. The complete model is developed in Dot net technology, the C# language is used to build the front end application whereas the SQL server is exploiting as database. The authorized person can only access the LMS system, they have to log in with their user id and password. As aforementioned that the LMS is designed in a user-friendly manner, so the admin can smoothly activate the system without expert advice. Every data is storing and retrieving from the SQL database so it is highly

secure. Therefore our system contributes its new approach towards the digital library setup.

With the advancement of technology, it is imperative to exalt all the systems into a user-friendly manner. The Library Management system (LMS) acts as a tool to transform traditional libraries into digital libraries. In traditional libraries, the students/user has to search for books which are hassle process and there is no proper maintenance of database about issues/fines. The overall progress of work is slow and it is impossible to generate a fast report. The librarians have to work allotted for arranging, sorting books in the book sells. At the same time, they have to check and monitor the lend/borrow book details with its fine. It is a tedious process to work simultaneously in different sectors. LMS will assist the librarians to work easily. The LMS supports the librarians to encounter all the issues concurrently. The users need not stand in a queue for a long period to return/borrow a book from the library. The single PC contains all the data's in it. The librarians have to assess the system and provide an entry in it. Through LMS the librarian can find the book in the bookshelves. The LMS is designed with the basic features such as librarian can add/view/update/delete books and students' details in it. Once he/she ingress into the system they can modify any data's in the database. The complete model is developed in Dot net technology, the C++ language is used to build the front end application whereas the SQL server is exploiting as database. The authorized person can only access the LMS system, they have to log in with their user id and password. As aforementioned that the LMS is designed in a user-friendly manner, so the admin can smoothly activate the system without expert advice. Every data is storing and retrieving from the SQL database so it is highly secure. Thus our system contributes its new approach towards the digital library setup.

A library is a place where a huge collection of books and resources are available which can be accessible by the users. It acts as a brain for the institutions. It enhances the dissemination of knowledge and spiritual civilization among the students. The tons of books and research works are captivating the students to improvise their
knowledge in all perspectives. It guides the students to promote their views differently. This knowledge optimizes the student to achieve a better result in academic as well as personal skill development. Improvisation in technology causes the demand for developing a way to enhance the traditional library set up to digital one. Numerous tedious processes reduce the efficiency of the library. For example, it always needs manual support to do any activities in the traditional library. The count and details of books are scribbled in the paper for reference. Each data is fetched in the notebook for future citations. To examine any data then they have to refer the notebooks. At the same time while distributing the books to the students they have to enter into the notebook where they need to represent the book id, distribution and renewal date, and student id. The librarians/staff have to assign a tag for each book and provide an id for it. They have to align and arrange the books on the shelves and marked it. Missing or theft of the book builds a serious issue and confusion to the

librarians. While collecting the book from the students they have to verify the penalties of the books. Therefore it causes a monotonous among the staff. Consequently, it builds an uninteresting among the student due to the slow progress of
the staff. To evoke the library into the technological era, we presented a system called Library Management system (LMS). It is an automatic system that reduces the work burden of the staff/librarians through a single click. It will manage, organize and
oriented the library task. The LMS supports the librarian to add/view/delete/update details from the library stock. Here we integrate all the library data into the SQL server. Preliminarily the librarian has to add studen and book details into the database. After that he/she can view/delete/update those details through the Library Management system. On account of this, the user can access the library at any time. The librarians can assist the data without any confusion. Each data are retrieved from the database. if he/she access any user details then it shows username, id, book details,
and penalty details. They no need to write it on paper for any references. By editing the data they can change the parameter in it. In spite of working on the manual, the librarian can feel easy to handle the automatic system. It has more additional features such as librarian can maintain library records, student's history of penalties and issues. It always tracks the count of the book in the library and issued book details. This causes a flexible service for librarians and students. It is a user-friendly interface, so basic computer knowledge is enough to access the LMS. The system is a customizable and user-configurable one which causes it to use in different organizations. We represent the LMS with Admin module. We built the LMS in .Net Technology which is considered as the one of the upcoming technology in IT industries. By the integration of all the modules, it will be presented on the desktop of your computer.
are stored together and maintained properly. It allows the user to create their database
as per the requirement. The database gets manipulated by the programs which provide
an interface between the databases. The database management system (DBMS) receives the command from the administrator based on the instruction it changes the data in the database. This instruction may load, retrieve or modify the existing database. It is better to assign a DBMS as a centralized one which helps multiple users to access the database in a controlled manner at a different location. Based on the scheme of DBMS, the system can assign a view mode for each user like some people can see only some data and authorized one can see all the data existing in the database. It offers both logical and physical data independence. The Open database

connectivity (ODBC) provides an application programming interface that allows the client-side program to call the DBMS on the server-side.

Expected Learning Outcomes:

This project offers several learning outcomes for students, including:

1. Understanding of Data Structures:

Students will gain a practical understanding of various data structures, such as arrays, linked lists, stacks, and queues. They will learn how to select and implement the appropriate data structure for storing and managing book data efficiently.

2. Algorithm Implementation:

By implementing search algorithms (e.g., binary search, linear search) and sorting algorithms (e.g., quick sort, merge sort), students will enhance their algorithmic skills. They will understand the importance of selecting the right algorithm for specific scenarios and learn to analyze their efficiency.

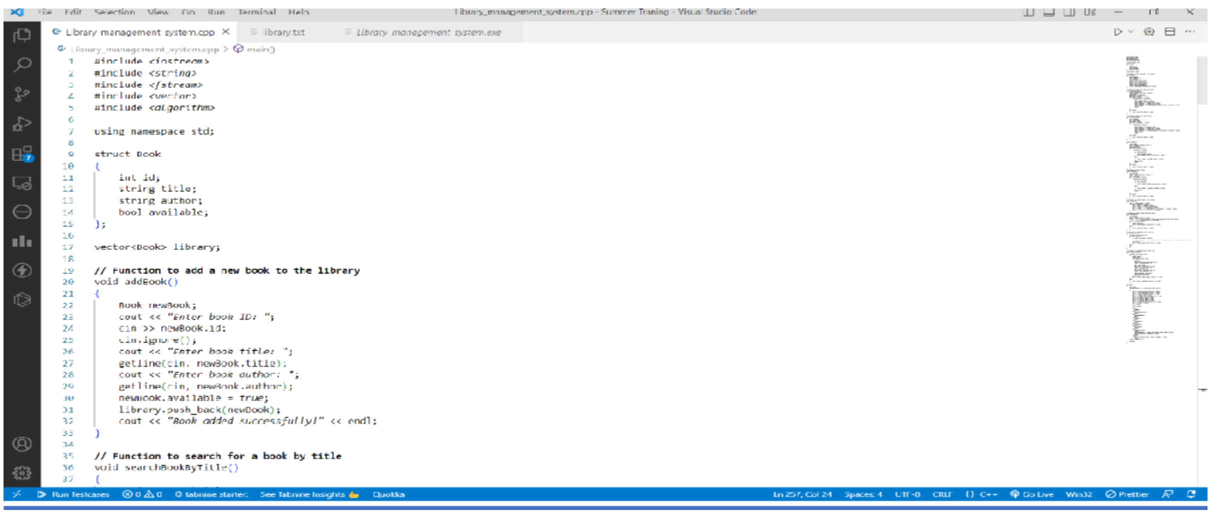
3. Real-World Application:

Building a library management system provides students with practical experience in developing software for real-world applications. They will gain insights into the challenges and considerations involved in creating systems that meet specific requirements and user needs.

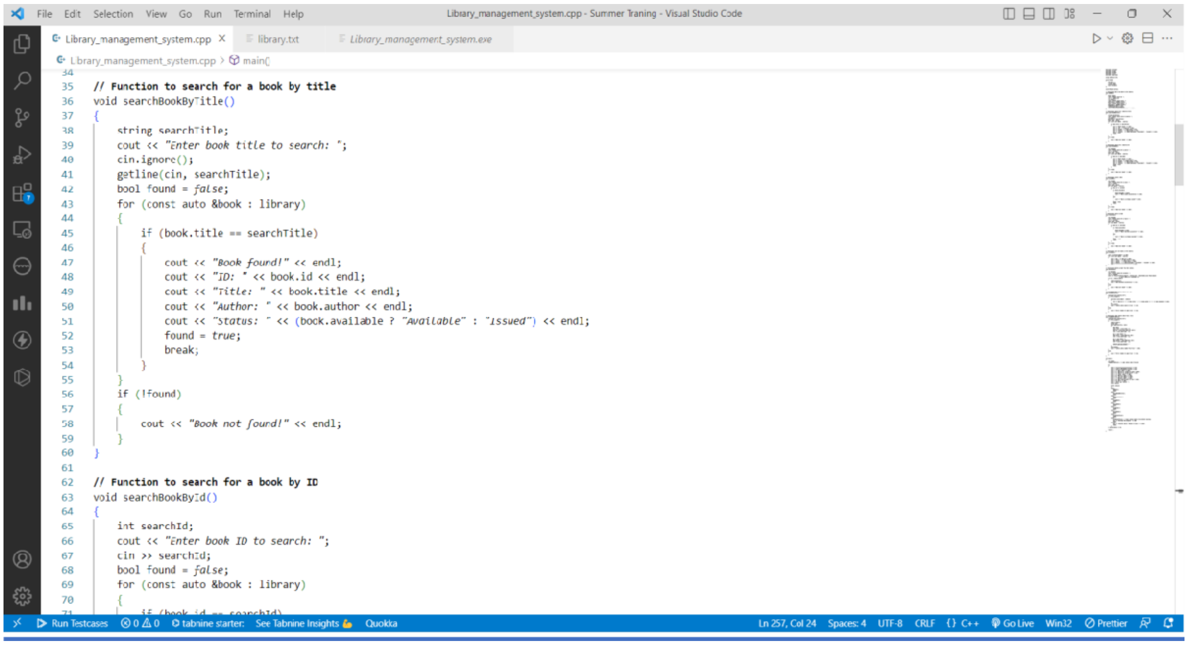
4. File Handling Techniques:

The project introduces students to file handling techniques, allowing them to store library data persistently. Students will learn how to load data from a file into the system and save the library data for future use.

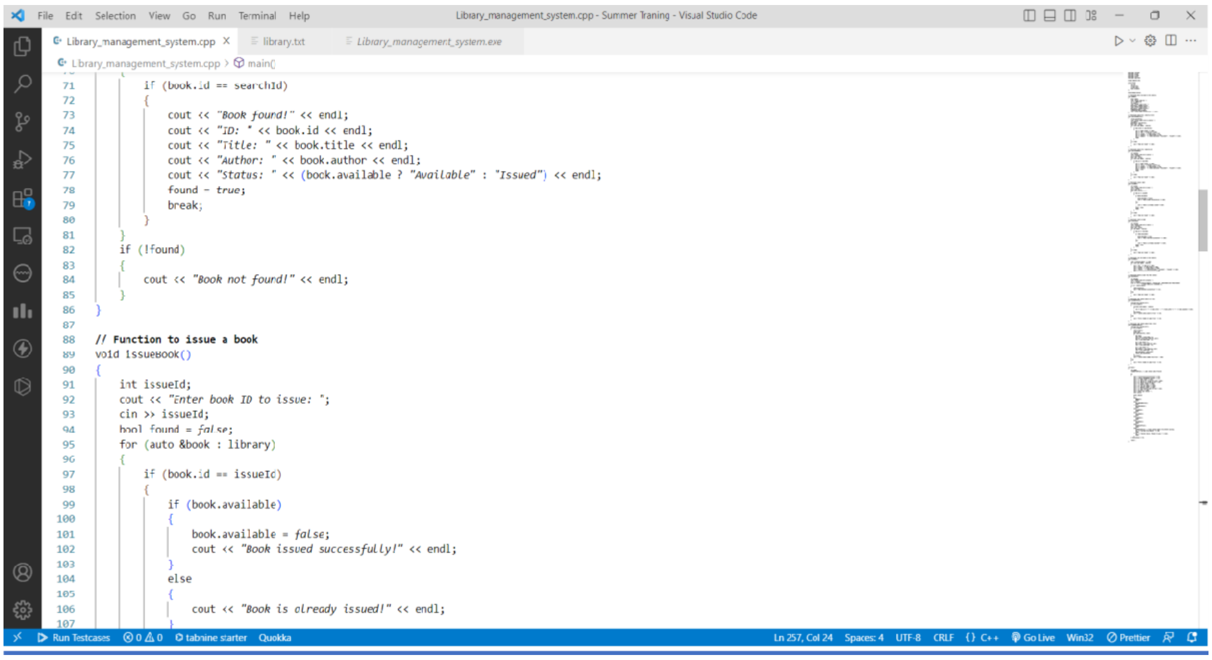
IMPLIMENTATION



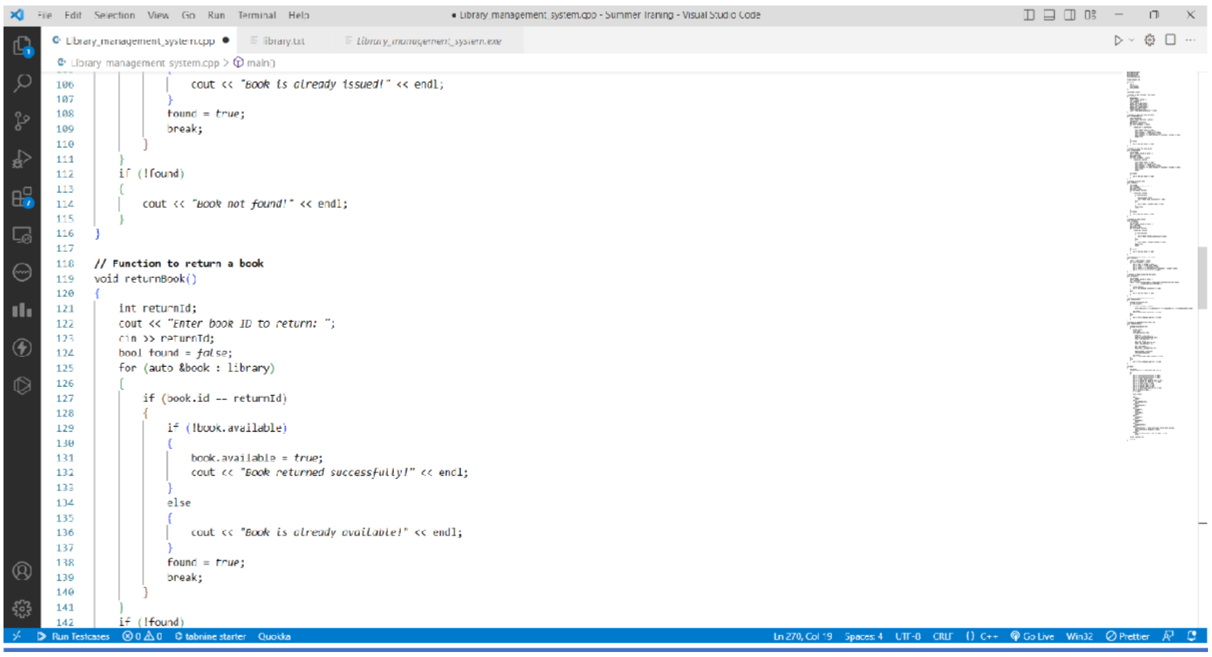
```cpp
#include <iostream>
#include <string>
#include <fstream>
#include <vector>
#include <algorithm>

using namespace std;

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

vector<Book> library;

// Function to add a new book to the library
void addBook()
{
    Book newBook;
    cout << "Enter book ID: ";
    cin >> newBook.id;
    cin.ignore();
    cout << "Enter book title: ";
    getline(cin, newBook.title);
    cout << "Enter book author: ";
    getline(cin, newBook.author);
    newBook.available = true;
    library.push_back(newBook);
    cout << "Book added successfully!" << endl;
}

// Function to search for a book by title
void searchBookByTitle()
{
```



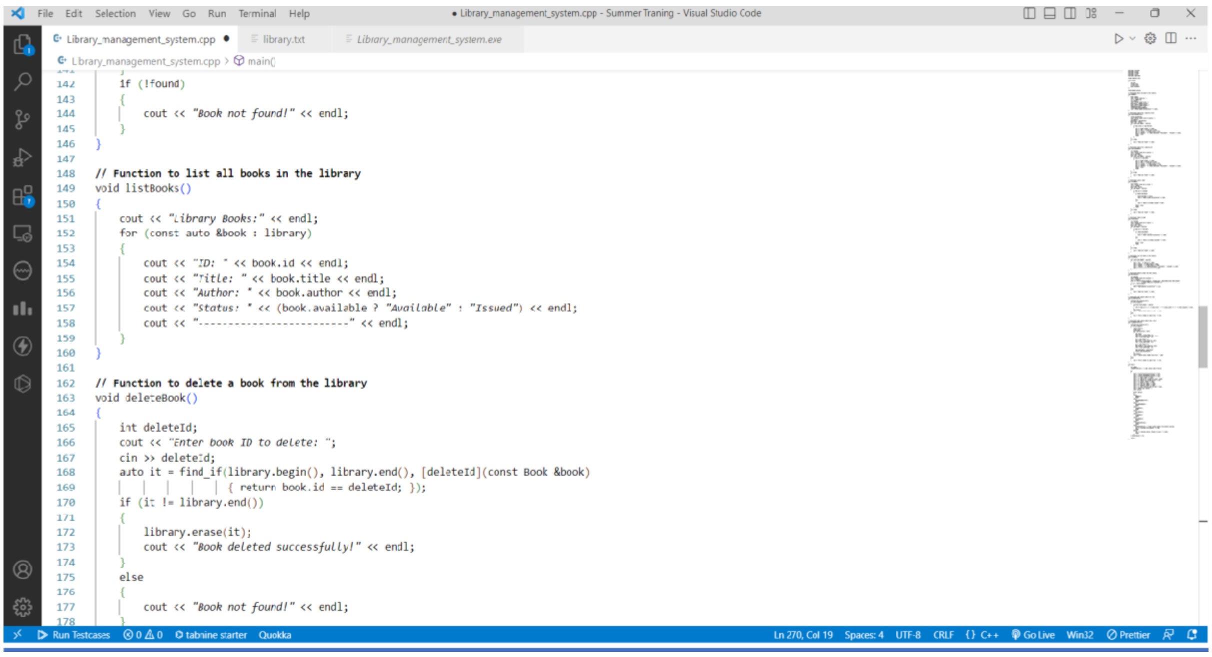
```cpp
// Function to search for a book by title
void searchBookByTitle()
{
    string searchTitle;
    cout << "Enter book title to search: ";
    cin.ignore();
    getline(cin, searchTitle);
    bool found = false;
    for (const auto &book : library)
    {
        if (book.title == searchTitle)
        {
            cout << "Book found!" << endl;
            cout << "ID: " << book.id << endl;
            cout << "Title: " << book.title << endl;
            cout << "Author: " << book.author << endl;
            cout << "Status: " << (book.available ? "Available" : "Issued") << endl;
            found = true;
            break;
        }
    }
    if (!found)
    {
        cout << "Book not found!" << endl;
    }
}

// Function to search for a book by ID
void searchBookById()
{
    int searchId;
    cout << "Enter book ID to search: ";
    cin >> searchId;
    bool found = false;
    for (const auto &book : library)
    {
        if (book.id == searchId)
```


File Edit Selection View Go Run Terminal Help
Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
Run Testcases
tabnine starter
Quokka
Ln 257, Col 24 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

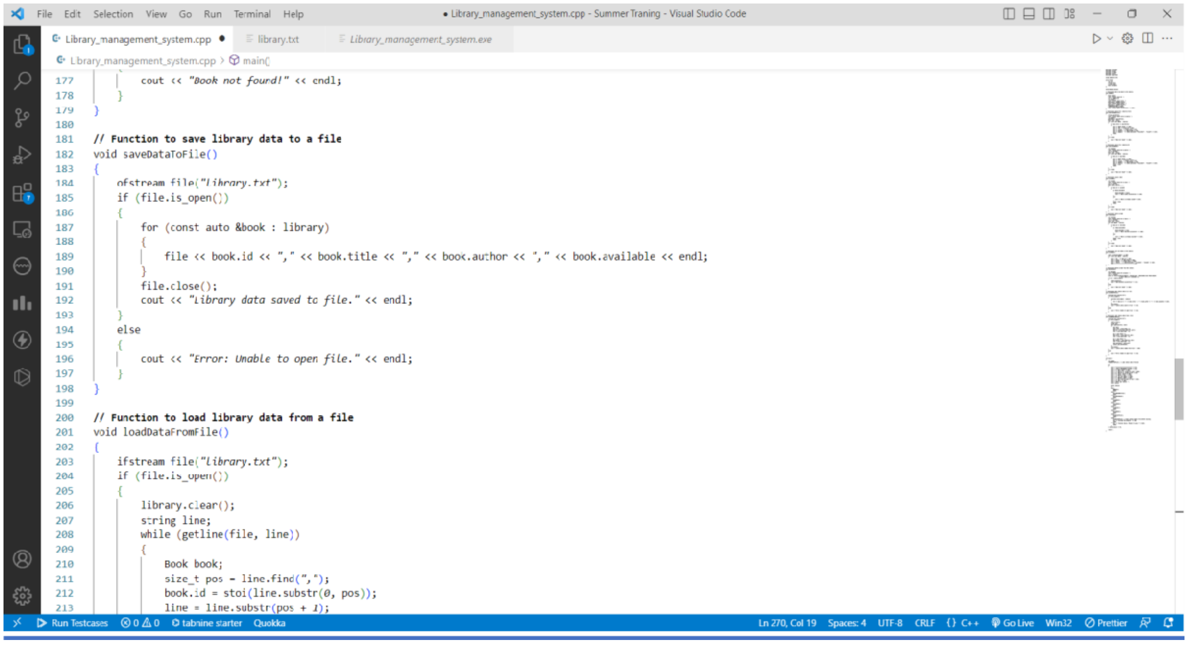
```cpp
            if (book.id == searchId)
            {
                cout << "Book found!" << endl;
                cout << "ID: " << book.id << endl;
                cout << "Title: " << book.title << endl;
                cout << "Author: " << book.author << endl;
                cout << "Status: " << (book.available ? "Available" : "Issued") << endl;
                found = true;
                break;
            }
        }
        if (!found)
        {
            cout << "Book not found!" << endl;
        }
}

// Function to issue a book
void issueBook()
{
    int issueId;
    cout << "Enter book ID to issue: ";
    cin >> issueId;
    bool found = false;
    for (auto &book : library)
    {
        if (book.id == issueId)
        {
            if (book.available)
            {
                book.available = false;
                cout << "Book issued successfully!" << endl;
            }
            else
            {
                cout << "Book is already issued!" << endl;
```


File Edit Selection View Go Run Terminal Help
Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

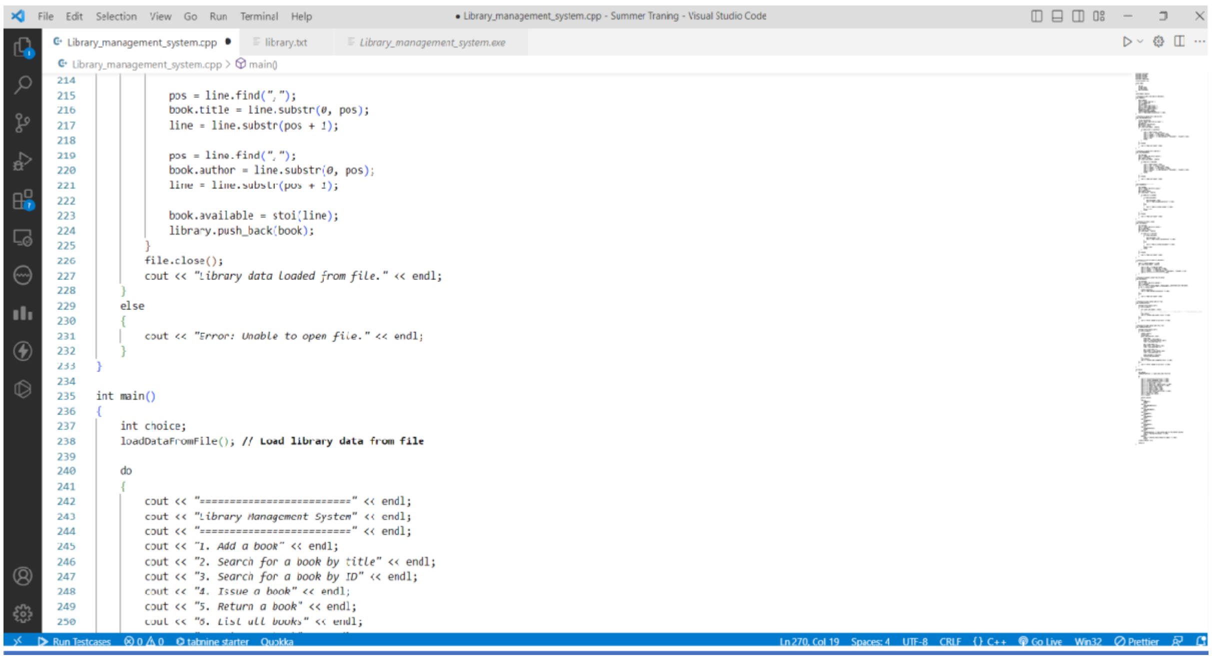
```cpp
                cout << "Book is already issued!" << endl;
            }
            found = true;
            break;
        }
    }
    if (!found)
    {
        cout << "Book not found!" << endl;
    }
}

// Function to return a book
void returnBook()
{
    int returnId;
    cout << "Enter book ID to return: ";
    cin >> returnId;
    bool found = false;
    for (auto &book : library)
    {
        if (book.id == returnId)
        {
            if (!book.available)
            {
                book.available = true;
                cout << "Book returned successfully!" << endl;
            }
            else
            {
                cout << "Book is already available!" << endl;
            }
            found = true;
            break;
        }
    }
    if (!found)
```


File Edit Selection View Go Run Terminal Help
• Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

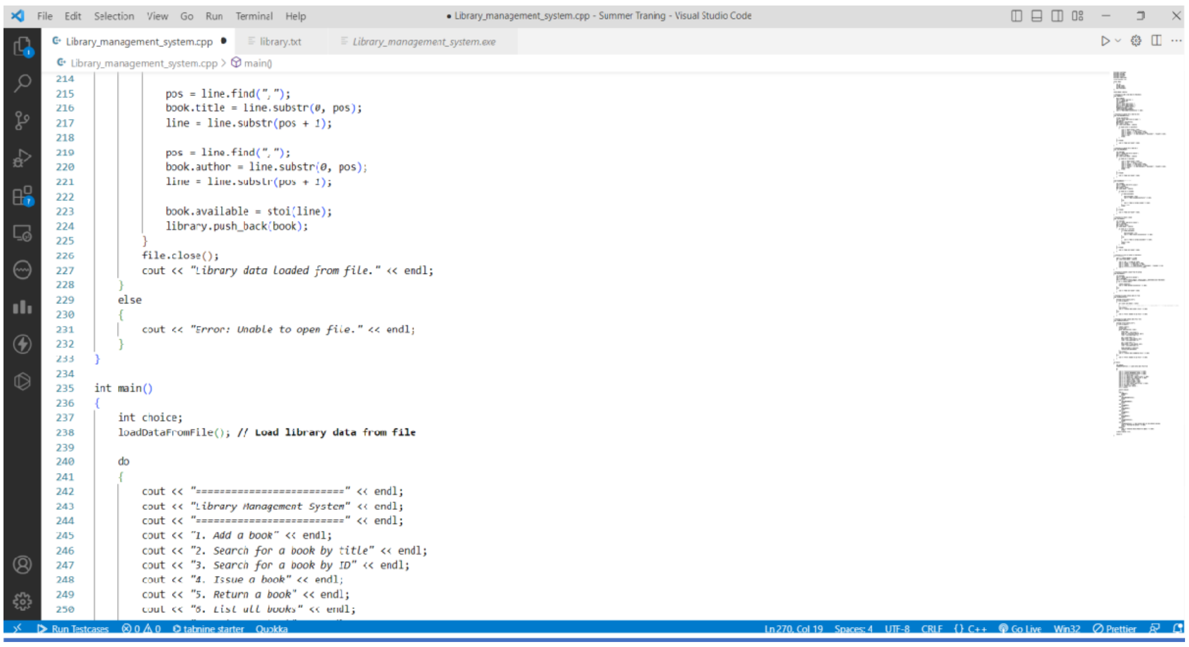
```cpp
    if (!found)
    {
        cout << "Book not found!" << endl;
    }
}

// Function to list all books in the library
void listBooks()
{
    cout << "Library Books:" << endl;
    for (const auto &book : library)
    {
        cout << "ID: " << book.id << endl;
        cout << "Title: " << book.title << endl;
        cout << "Author: " << book.author << endl;
        cout << "Status: " << (book.available ? "Available" : "Issued") << endl;
        cout << "------------------------" << endl;
    }
}

// Function to delete a book from the library
void deleteBook()
{
    int deleteId;
    cout << "Enter book ID to delete: ";
    cin >> deleteId;
    auto it = find_if(library.begin(), library.end(), [deleteId](const Book &book)
                      { return book.id == deleteId; });
    if (it != library.end())
    {
        library.erase(it);
        cout << "Book deleted successfully!" << endl;
    }
    else
    {
        cout << "Book not found!" << endl;
    }
```


File Edit Selection View Go Run Terminal Help
• Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

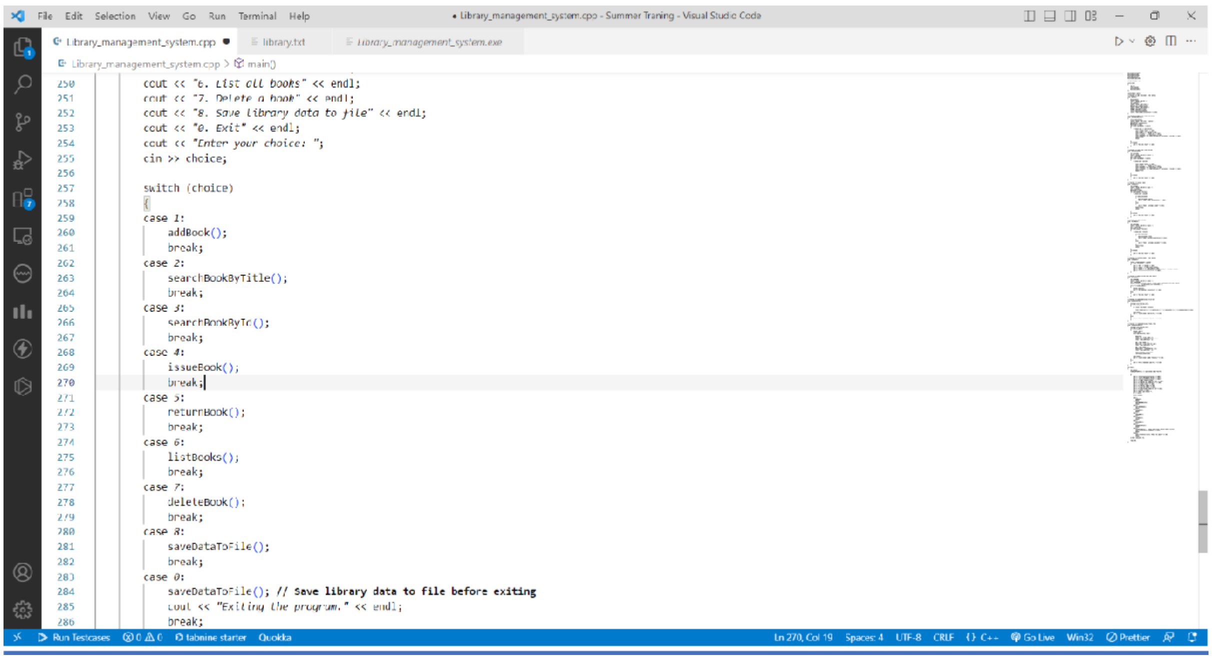
```cpp
        cout << "Book not found!" << endl;
    }
}

// Function to save library data to a file
void saveDataToFile()
{
    ofstream file("library.txt");
    if (file.is_open())
    {
        for (const auto &book : library)
        {
            file << book.id << "," << book.title << "," << book.author << "," << book.available << endl;
        }
        file.close();
        cout << "Library data saved to file." << endl;
    }
    else
    {
        cout << "Error: Unable to open file." << endl;
    }
}

// Function to load library data from a file
void loadDataFromFile()
{
    ifstream file("library.txt");
    if (file.is_open())
    {
        library.clear();
        string line;
        while (getline(file, line))
        {
            Book book;
            size_t pos = line.find(",");
            book.id = stoi(line.substr(0, pos));
            line = line.substr(pos + 1);
```


File Edit Selection View Go Run Terminal Help
• Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

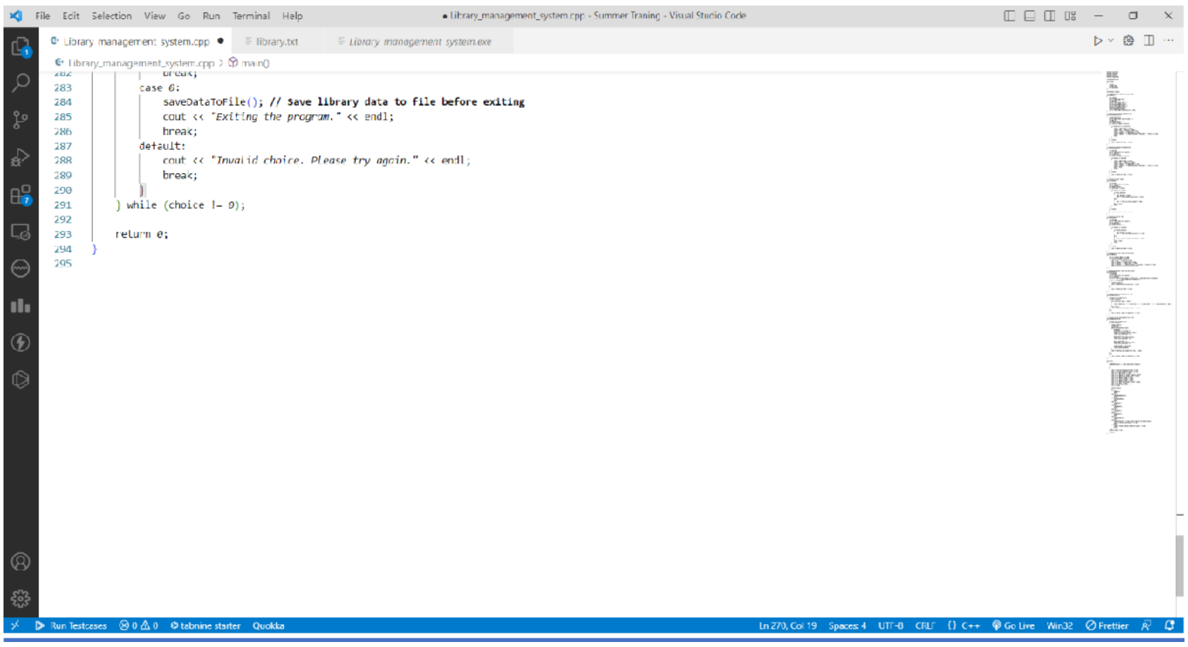
```
            pos = line.find(",");
            book.title = line.substr(0, pos);
            line = line.substr(pos + 1);

            pos = line.find(",");
            book.author = line.substr(0, pos);
            line = line.substr(pos + 1);

            book.available = stoi(line);
            library.push_back(book);
        }
        file.close();
        cout << "Library data loaded from file." << endl;
    }
    else
    {
        cout << "Error: Unable to open file." << endl;
    }
}

int main()
{
    int choice;
    loadDataFromFile(); // Load library data from file

    do
    {
        cout << "========================" << endl;
        cout << "Library Management System" << endl;
        cout << "========================" << endl;
        cout << "1. Add a book" << endl;
        cout << "2. Search for a book by title" << endl;
        cout << "3. Search for a book by ID" << endl;
        cout << "4. Issue a book" << endl;
        cout << "5. Return a book" << endl;
        cout << "6. List all books" << endl;
```



```
            pos = line.find(",");
            book.title = line.substr(0, pos);
            line = line.substr(pos + 1);

            pos = line.find(",");
            book.author = line.substr(0, pos);
            line = line.substr(pos + 1);

            book.available = stoi(line);
            library.push_back(book);
        }
        file.close();
        cout << "Library data loaded from file." << endl;
    }
    else
    {
        cout << "Error: Unable to open file." << endl;
    }
}

int main()
{
    int choice;
    loadDataFromFile(); // Load library data from file

    do
    {
        cout << "========================" << endl;
        cout << "Library Management System" << endl;
        cout << "========================" << endl;
        cout << "1. Add a book" << endl;
        cout << "2. Search for a book by title" << endl;
        cout << "3. Search for a book by ID" << endl;
        cout << "4. Issue a book" << endl;
        cout << "5. Return a book" << endl;
        cout << "6. List all books" << endl;
```


File Edit Selection View Go Run Terminal Help
• Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

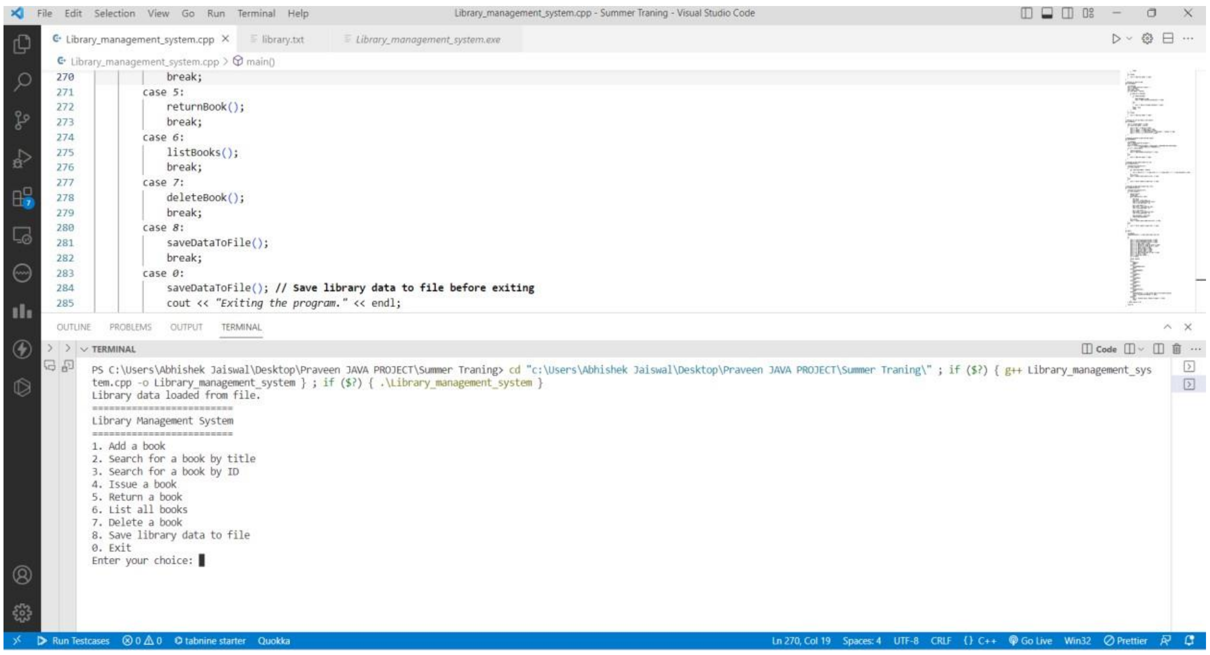
```
        cout << "6. List all books" << endl;
        cout << "7. Delete a book" << endl;
        cout << "8. Save library data to file" << endl;
        cout << "0. Exit" << endl;
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice)
        {
        case 1:
            addBook();
            break;
        case 2:
            searchBookByTitle();
            break;
        case 3:
            searchBookById();
            break;
        case 4:
            issueBook();
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
            break;
        case 7:
            deleteBook();
            break;
        case 8:
            saveDataToFile();
            break;
        case 0:
            saveDataToFile(); // Save library data to file before exiting
            cout << "Exiting the program." << endl;
            break;
```


File Edit Selection View Go Run Terminal Help
• Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

```
            break;
        case 0:
            saveDataToFile(); // Save library data to file before exiting
            cout << "Exiting the program." << endl;
            break;
        default:
            cout << "Invalid choice. Please try again." << endl;
            break;
        }
    } while (choice != 0);

    return 0;
}
```

OUTPUT

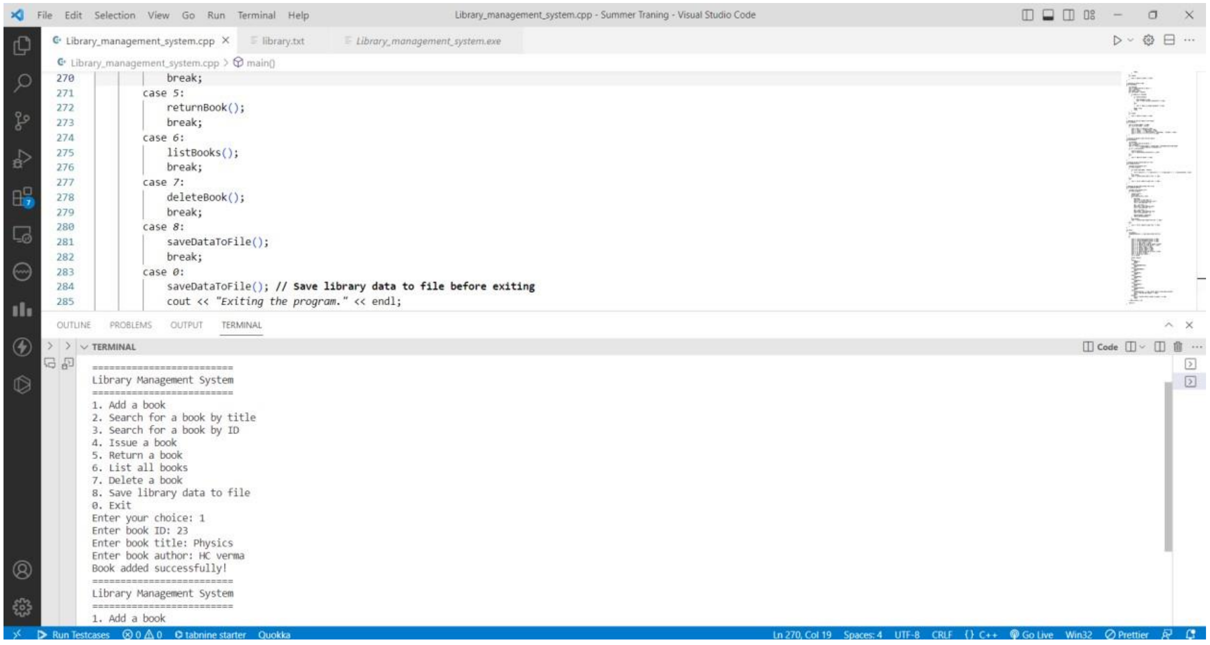
Main Menu



```
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
            break;
        case 7:
            deleteBook();
            break;
        case 8:
            saveDataToFile();
            break;
        case 0:
            saveDataToFile(); // Save library data to file before exiting
            cout << "Exiting the program." << endl;
```

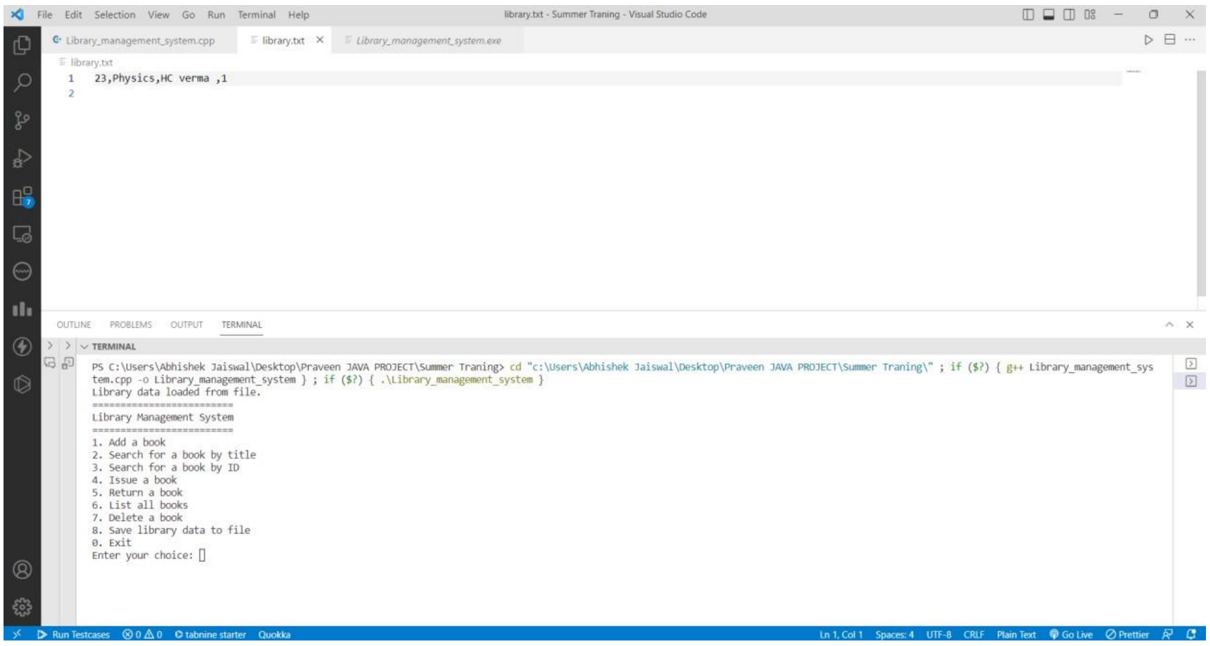
```
PS C:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning> cd "c:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning\" ; if ($?) { g++ Library_management_sys
tem.cpp -o Library_management_system } ; if ($?) { .\Library_management_system }
Library data loaded from file.
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 
```

Adding a new book feature



```
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
            break;
        case 7:
            deleteBook();
            break;
        case 8:
            saveDataToFile();
            break;
        case 0:
            saveDataToFile(); // Save library data to file before exiting
            cout << "Exiting the program." << endl;
```

```
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 1
Enter book ID: 23
Enter book title: Physics
Enter book author: HC verma
Book added successfully!
========================
Library Management System
========================
1. Add a book
```

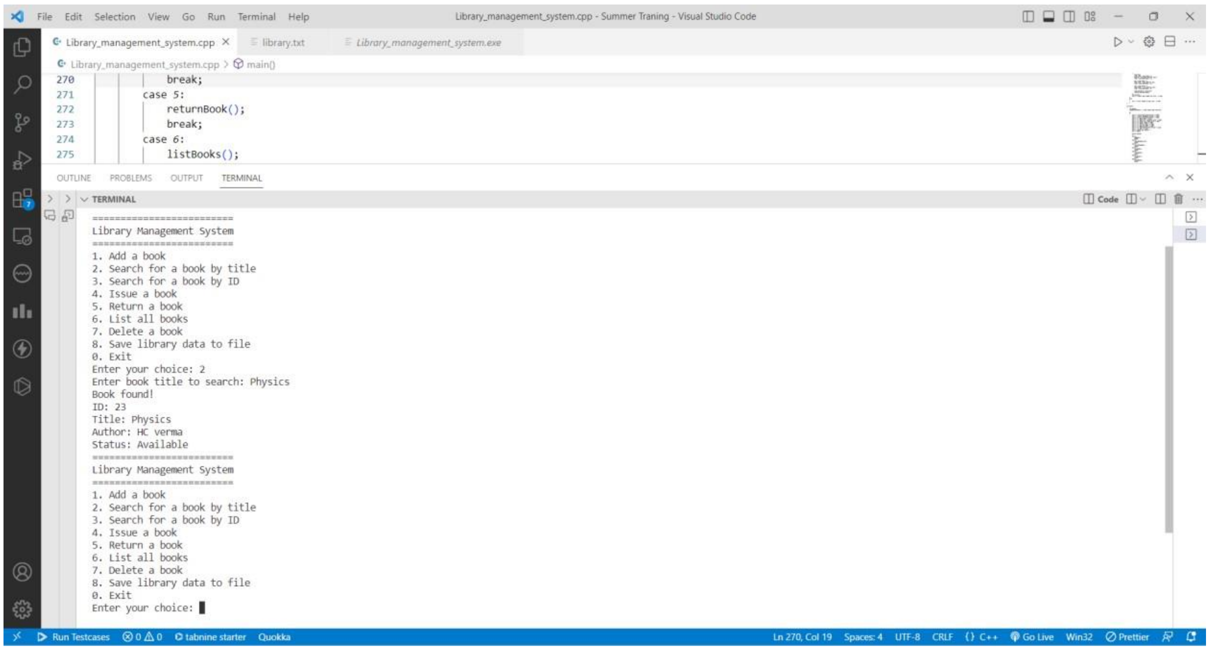


```
23,Physics,HC verma ,1
```

```
PS C:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning> cd "c:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning\" ; if ($?) { g++ Library_management_sys
tem.cpp -o Library_management_system } ; if ($?) { .\Library_management_system }
Library data loaded from file.
=========================
Library Management System
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice:
```

Searching for the book feature

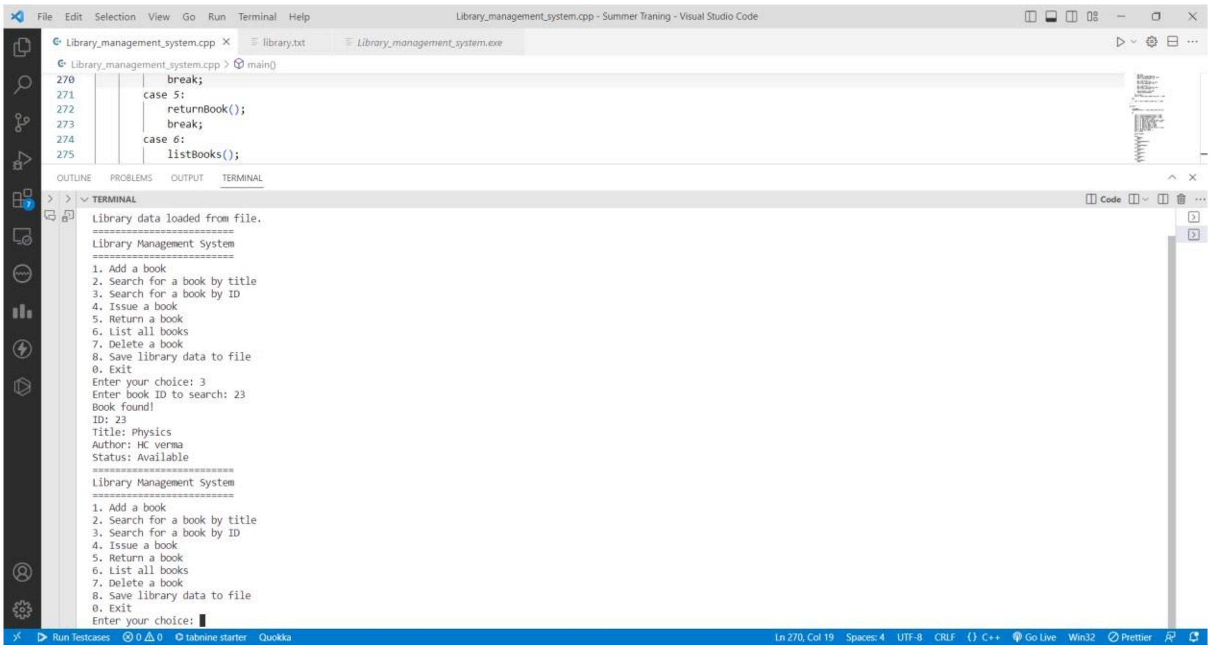
Searching by Title



```
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
```

```
=========================
Library Management System
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 2
Enter book title to search: Physics
Book found!
ID: 23
Title: Physics
Author: HC verma
Status: Available
=========================
Library Management System
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice:
```

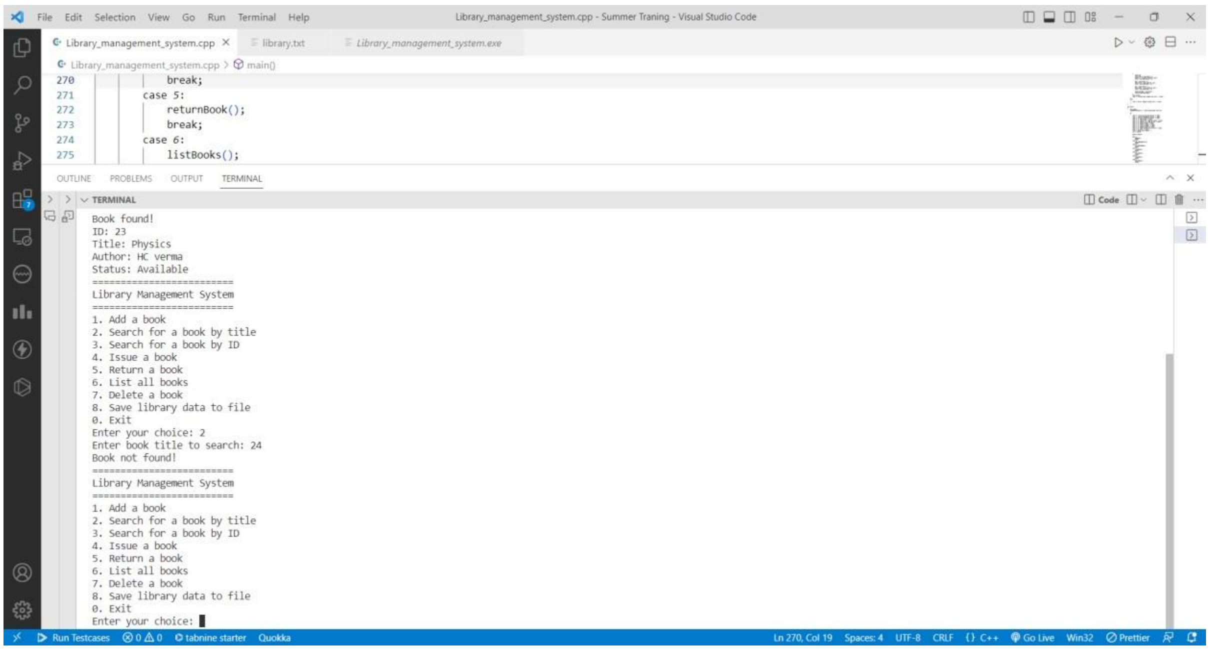
Searching by Id


File Edit Selection View Go Run Terminal Help
Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
OUTLINE PROBLEMS OUTPUT TERMINAL
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF C++ Go Live Win32 Prettier

```
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
```

```
Library data loaded from file.
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 3
Enter book ID to search: 23
Book found!
ID: 23
Title: Physics
Author: HC verma
Status: Available
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 
```

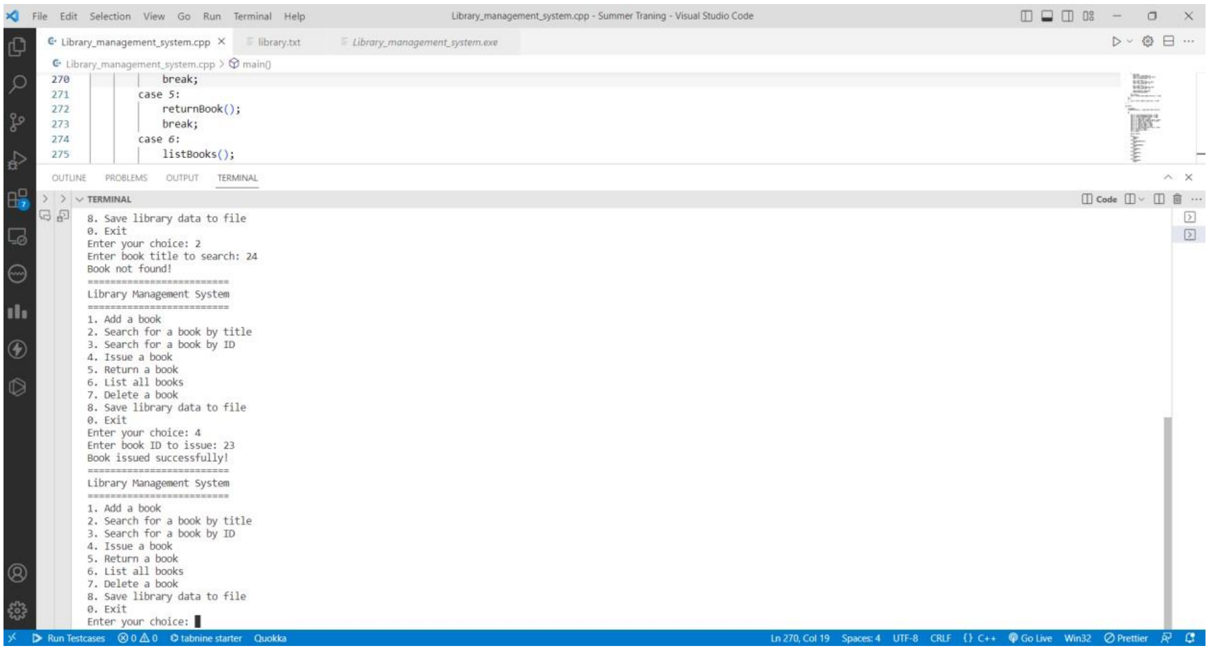
Book not found


File Edit Selection View Go Run Terminal Help
Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
OUTLINE PROBLEMS OUTPUT TERMINAL
Run Testcases
tabnine starter
Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF C++ Go Live Win32 Prettier

```
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
```

```
Book found!
ID: 23
Title: Physics
Author: HC verma
Status: Available
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 2
Enter book title to search: 24
Book not found!
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 
```

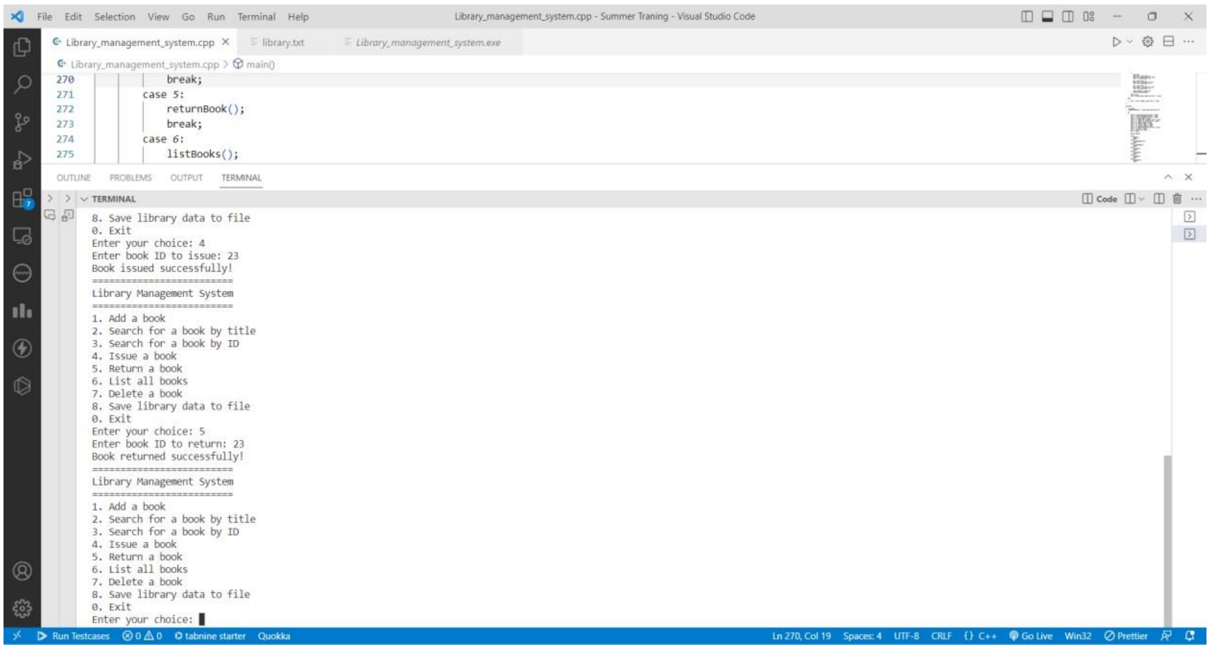
Issue the book.


File Edit Selection View Go Run Terminal Help
Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
OUTLINE PROBLEMS OUTPUT TERMINAL
TERMINAL
Code
Run Testcases ⊗ 0 ⚠ 0 tabnine starter Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

```
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
```

```
8. Save library data to file
0. Exit
Enter your choice: 2
Enter book title to search: 24
Book not found!
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 4
Enter book ID to issue: 23
Book issued successfully!
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: █
```

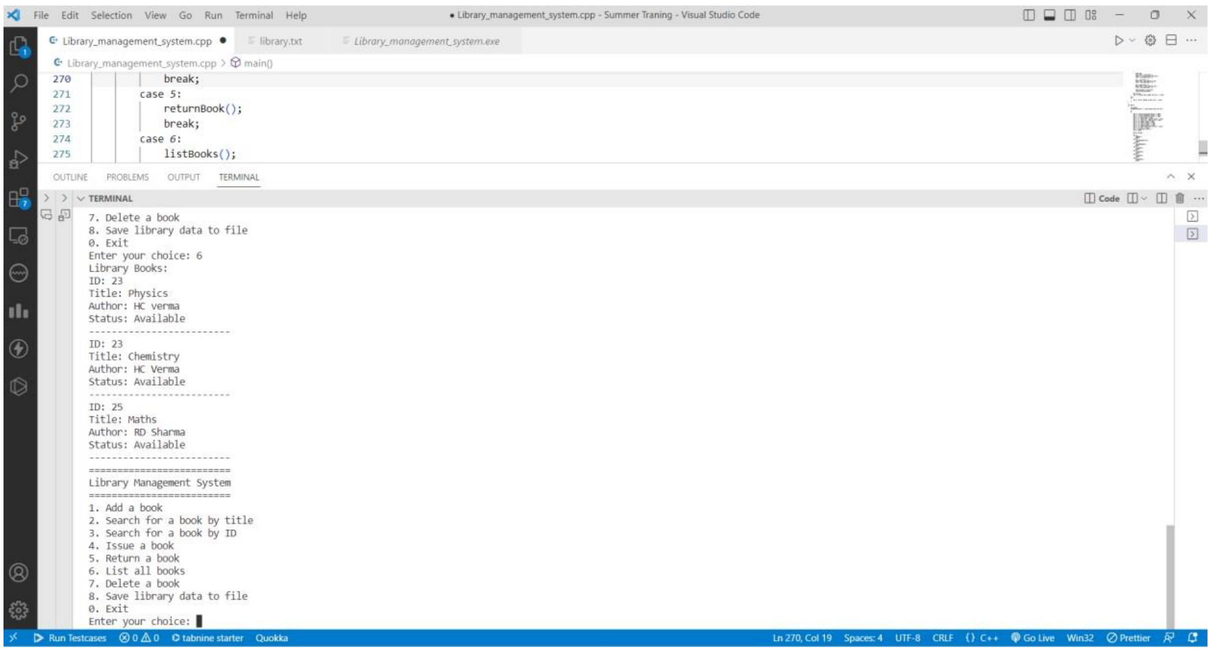
Return The book.


File Edit Selection View Go Run Terminal Help
Library_management_system.cpp - Summer Traning - Visual Studio Code
Library_management_system.cpp
library.txt
Library_management_system.exe
Library_management_system.cpp > main()
OUTLINE PROBLEMS OUTPUT TERMINAL
TERMINAL
Code
Run Testcases ⊗ 0 ⚠ 0 tabnine starter Quokka
Ln 270, Col 19 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32 Prettier

```
            break;
        case 5:
            returnBook();
            break;
        case 6:
            listBooks();
```

```
8. Save library data to file
0. Exit
Enter your choice: 4
Enter book ID to issue: 23
Book issued successfully!
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 5
Enter book ID to return: 23
Book returned successfully!
========================
Library Management System
========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: █
```

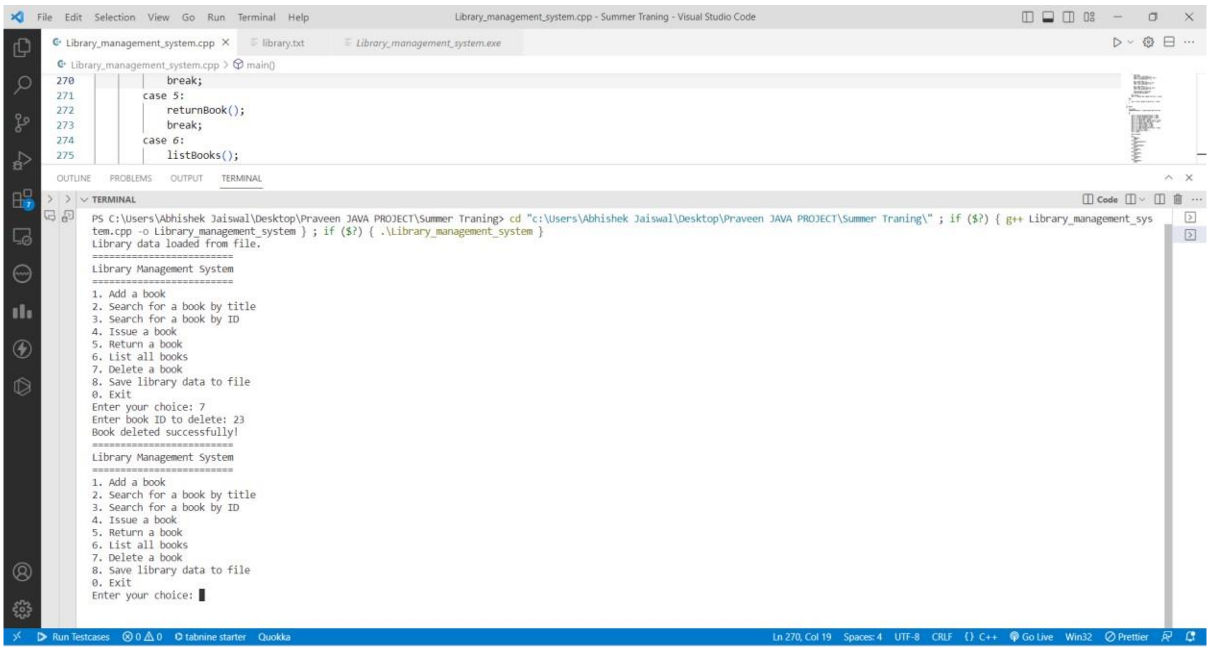
List of All the books



```cpp
                break;
            case 5:
                returnBook();
                break;
            case 6:
                listBooks();
```

```
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 6
Library Books:
ID: 23
Title: Physics
Author: HC verma
Status: Available
-------------------------
ID: 23
Title: Chemistry
Author: HC Verma
Status: Available
-------------------------
ID: 25
Title: Maths
Author: RD Sharma
Status: Available
-------------------------
=========================
Library Management System
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice:
```

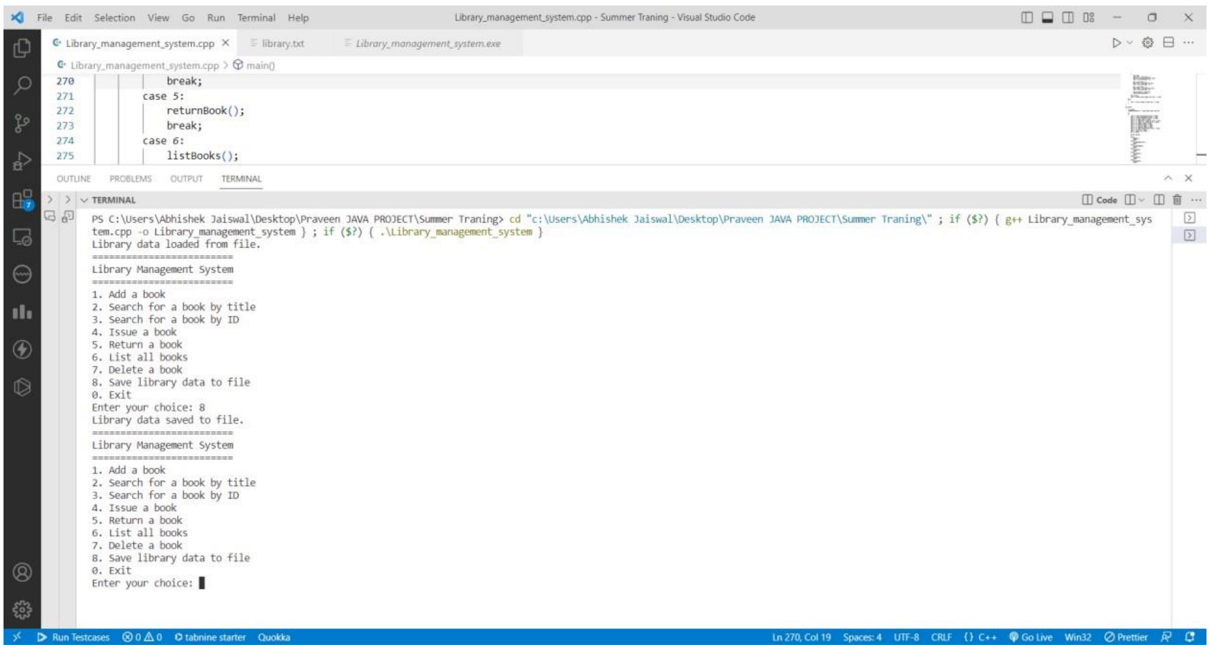
Delete the book



```cpp
                break;
            case 5:
                returnBook();
                break;
            case 6:
                listBooks();
```

```
PS C:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning> cd "c:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning\" ; if ($?) { g++ Library_management_sys
tem.cpp -o Library_management_system } ; if ($?) { .\Library_management_system }
Library data loaded from file.
=========================
Library Management System
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 7
Enter book ID to delete: 23
Book deleted successfully!
=========================
Library Management System
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice:
```

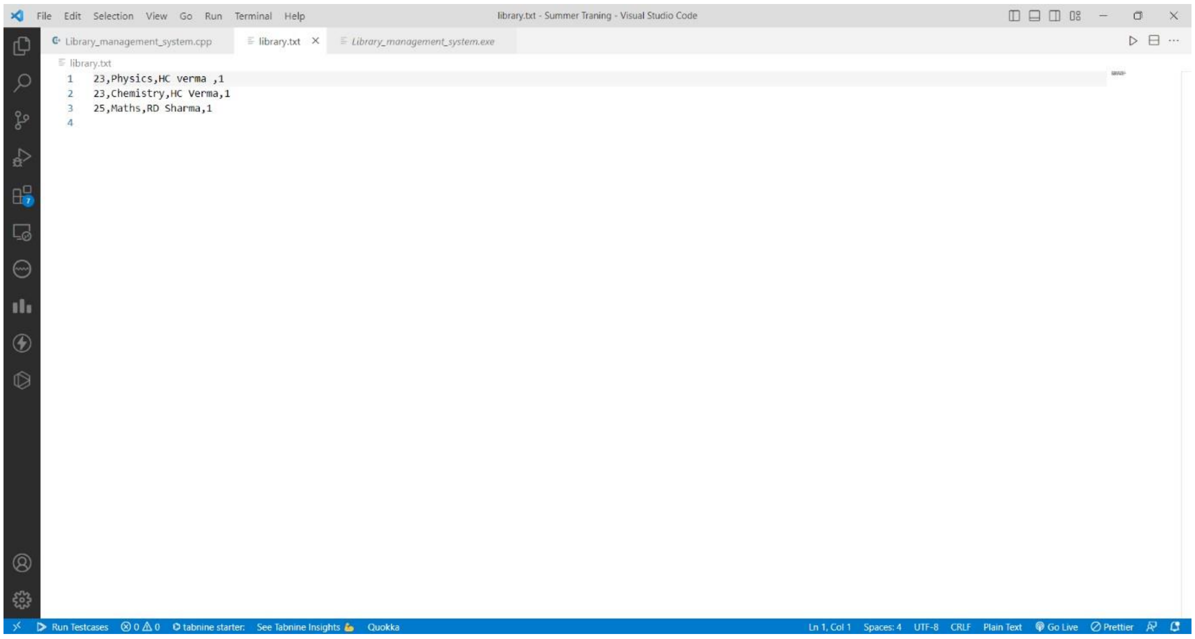
Save the library Data to file.



```
        break;
    case 5:
        returnBook();
        break;
    case 6:
        listBooks();
```

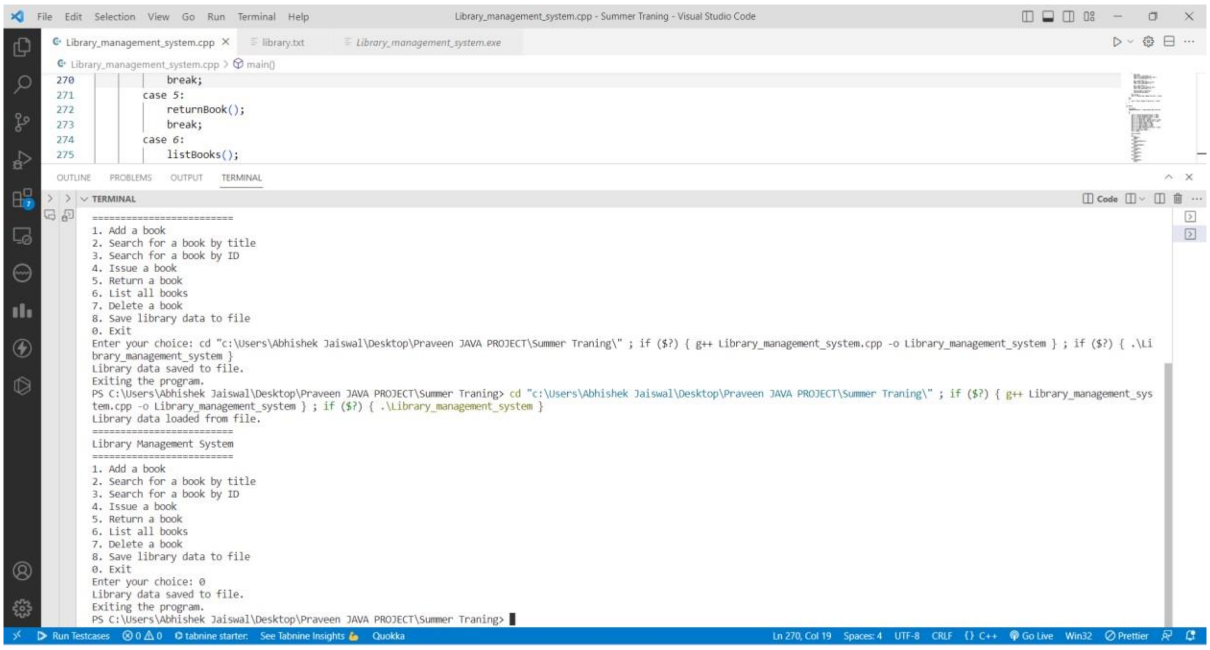
```
PS C:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning> cd "c:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning\" ; if ($?) { g++ Library_management_sys
tem.cpp -o Library_management_system } ; if ($?) { .\Library_management_system }
Library data loaded from file.
==========================
Library Management System
==========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 8
Library data saved to file.
==========================
Library Management System
==========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 
```

Txt File as a database



```
23,Physics,HC verma ,1
23,Chemistry,HC Verma,1
25,Maths,RD Sharma,1
```

Exiting the Main Menu



```
        break;
    case 5:
        returnBook();
        break;
    case 6:
        listBooks();
```

```
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: cd "c:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning\" ; if ($?) { g++ Library_management_system.cpp -o Library_management_system } ; if ($?) { .\Li
brary_management_system }
Library data saved to file.
Exiting the program.
PS C:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning> cd "c:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning\" ; if ($?) { g++ Library_management_sys
tem.cpp -o Library_management_system } ; if ($?) { .\Library_management_system }
Library data loaded from file.
=========================
Library Management System
=========================
1. Add a book
2. Search for a book by title
3. Search for a book by ID
4. Issue a book
5. Return a book
6. List all books
7. Delete a book
8. Save library data to file
0. Exit
Enter your choice: 0
Library data saved to file.
Exiting the program.
PS C:\Users\Abhishek Jaiswal\Desktop\Praveen JAVA PROJECT\Summer Traning>
```

Future Work

While the Simple Library Management System project provides a solid foundation for managing books in a library, there are several areas where further enhancements and additions can be made. Hereare some potential future work ideas to consider:

1. User Authentication and Roles:

Implement a user authentication system to secure the library management system. Introduce different user roles such as librarians, administrators, and students, each with specific privileges and access levels.

2. Book Reservation System:

Add a feature that allows students to reserve books in advance. Implement a reservation queue or priority queue to manage book reservations and ensure fairness in the allocation of books.

3. Fine Calculation:

Enhance the system by including a fine calculation mechanism for late book returns. Implement logic to calculate fines based on predefined rules, such as a daily or weekly rate.

4. Book Recommendations:

Integrate a book recommendation system to suggest relevant books to users based on their interests, previous borrowing history, or popular book trends.

5. Notifications and Reminders:

Develop a notification system that sends reminders to students for book due dates, overdue books, and reservation status updates. This can be implemented through email notifications or SMS alerts.

6. Analytics and Reporting:

Incorporate analytical capabilities to generate reports on book usage, popular book genres, frequent borrowers, and other relevant metrics. These insights can aid librarians in making data-driven decisions.

7. Integration with Online Catalogs:

Integrate the library management system with online book catalogs or ISBN databases to automatically fetch book details and reduce manual data entry. This can improve efficiency and accuracy in adding new books to the system.

8. Mobile Application:

Create a mobile application version of the library management system to provide convenient access to students and librarians. This would enable them to search for books, issue or return books, and receive notifications on their mobile devices.

9. Data Backup and Recovery:

Implement a robust data backup mechanism to ensure the system's data is protected and can be restored in case of any unforeseen issues or system failures.

10. Multilingual Support:

Extend the system's capabilities by adding multilingual support to cater to diverse user populations. This would involve translating the user interface and book details into different languages.

These future work ideas can further enhance the functionality, usability, and efficiency of the library management system. Prioritize the features based on user requirements and the resources available for development and implementation.

CONCLUSION

In conclusion, the Simple Library Management System project provides a practical and valuable learning experience for students. By implementing various data structures such as arrays, linked lists, stacks, and queues, students gain hands-on experience in utilizing these structures to manage book dataefficiently. The project also involves implementing essential algorithms like searching and sorting to enhance the system's functionality.

Throughout the project, students have the opportunity to apply their knowledge of programming concepts, data structures, and algorithms in a real-world scenario. They gain a deeper understanding of how these concepts are employed in practical applications and develop problem-solving skills by tackling challenges encountered during the project.

The project's features, including adding new books, searching for books, issuing and returning books, listing all books, and deleting books, provide a comprehensive set of functionalities that meet the needs of librarians in managing a library effectively. Students also learn about the importance of user interface design and user experience considerations when building software applications.

Moreover, the project's expected learning outcomes encompass a broader perspective of software development skills. Students learn to analyze requirements, design and implement efficient algorithms, select appropriate data structures, and consider factors such as scalability, security, and usability. They also gain insights into the importance of documentation and report writing, as demonstrated by the preparation of a project report.

Looking towards the future, the project opens up possibilities for further enhancements and expansions. Additional features like user authentication and roles, book reservation systems, fine calculations, and integration with external systems can be considered for future iterations of the librarymanagement system. These improvements would contribute to a more comprehensive and sophisticated solution that meets evolving user needs.

In summary, the Simple Library Management System project serves as a valuable learning opportunity for students to apply their programming skills, deepen their understanding of data structures and algorithms, and develop a practical software application. It equips them with essential knowledge and experiences that are transferable to other software development projects in the future.

Moreover, the project's expected learning outcomes encompass a broader perspective of software development skills. Students learn to analyze requirements, design and implement efficient algorithms, select appropriate data structures, and consider factors such as scalability, security, and usability. They also gain insights into the importance of documentation and report writing, as demonstrated by the preparation of a project report.

After we have completed the project we are sure the problems in the existing system would overcome. The Library Management System process made computerized to reduce human errors and to increase the efficiency. The main focus of this project is to lessen human efforts. The maintenance of the records is made efficient, as all the records are stored in the ACCESS database, through which data can be retrieved easily. The navigation control is provided in all the forms to navigate through the large amount of records. If the numbers of records are very large then user has to just type in the search string and user gets the results immediately. The editing is also made simpler. The user has to just type in the required field and press the update button to update the desired field.

The Books and Students are given a particular unique id no. So that they can be accessed correctly and without errors. Our main aim of the project is to get the correct information about a particular student and books available in the library.

The problems, which existed in the earlier system, have been removed to a large extent. And it is expected that this project will go a long way in satisfying users requirements. The computerization of the Library Management will not only improves the efficiency but will also reduce human stress thereby indirectly improving human recourses.

When the programmer needs a specific user interface feature such as a button, he selects the appropriate ready to use component provided by the Visual programming environment. These components can be moved, resized and renamed as required. So a Visual programming environment automates the process of creating the user interface by designing Visual interface using the ready to use components. In addition, it also provides the means of associating the user written logically defined code with the components used in a project.

This project gives us the complete information about the library. We can enter the record of new books and retrieve the details of books available in the library. We can issue the books to the students and maintain their records and can also check how many books are issued and stock available in the library. In this project we can maintain the late fine of students who returns the issued books after the due date .

Throughout the project the focus has been on presenting information and comments in an easy and intelligible manner. The project is very useful for those who want to know about Library Management System.

REFERENCES

[1]. HonghaiKan , Zhimin Yang, Yue Wang, Nana Qi, “Research on Library Management System for CDs Attached to Books Based on Cloud Computing”, in Proceedings of the 14th International Conference on Computer Supported Cooperative Work in Design 2010.

[2]. Bao Sun, JiangweiFeng and Ling Liu, “A Study on How to Construct the Prediction Model of Library Lending of University Library”, International Conference on Information Science and Technology March 26-28, 2011 Nanjing, Jiangsu, China.

[3]. Erxiang Chen,Minghui Liu,“Research and Design on Library Management System Based on Struts and Hibernate Framework”, WASE International Conference on Information Engineering2009.

[4]. JianhuZheng, YunqingFeng, Yun Zhao, “A Unified Modeling Language-Based cybernetics and information technologies.

[5]. Michael Hitchens, Andrew Firmage,“The Design of a Flexible Class Library

[6]. WeihongYang,“Design and Implementation of Library Management System”, International Conference on Management Science and Innovative Education (MSIE 2015).

[7]. Bretthauer, D. “Open source software in libraries. Library Hi Tech News, 18 (5), 8-9(2001).

[8]. Barve, S., &Dahibhate, N. B.,“Open source software for library services”, DESIDOC Journal of Library & Information Technology, 32(5)(2012).

[9]. Albee, B. & Chen, Hsin-liang, “Public library staff’s perceived value and satisfaction of an opensource library system”. Electronic Library, 32(3), 39.-402(2014).

[10]. Singh, V., “Expectation versus experience: librarians using open source integrated library systems”, The Electronic Library, 32 (5), 688-709(2014).

[11]. Ching-yu Huang and Patricia A, “Morreale A Web-based, Self-Controlled Mechanism to Support Students Learning SQL” IEEE Integrated STEM Education Conference (ISEC)2016.

SOURCE CODE:- https://github.com/Praveenkumartiwari321/Library-Management-System.git