```
#include <iostream>
#include <string>
#include <vector>

using namespace std;

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

// Function to add a new book to the library
void addBook(vector<Book> &library)
{
    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!\n";
}

// Function to search for a book by title or ID
void searchBook(const vector<Book> &library)
{
    string keyword;
    cout << "Enter Title or ID of the Book to Search: ";
    cin.ignore();
    getline(cin, keyword);

    bool found = false;
    for (const Book &book : library)
    {
        if (book.title == keyword || to_string(book.id) == keyword)
        {
            cout << "Book 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.\n";
}

// Function to issue a book
```

```cpp
void issueBook(vector<Book> &library)
{
    int bookId;
    cout << "Enter Book ID to Issue: ";
    cin >> bookId;

    bool found = false;
    for (Book &book : library)
    {
        if (book.id == bookId && book.available)
        {
            book.available = false;
            cout << "Book issued successfully!\n";
            found = true;
            break;
        }
    }

    if (!found)
        cout << "Book not available or does not exist.\n";
}

// Function to return a book
void returnBook(vector<Book> &library)
{
    int bookId;
    cout << "Enter Book ID to Return: ";
    cin >> bookId;

    bool found = false;
    for (Book &book : library)
    {
        if (book.id == bookId && !book.available)
        {
            book.available = true;
            cout << "Book returned successfully!\n";
            found = true;
            break;
        }
    }

    if (!found)
        cout << "Invalid book ID or book already available.\n";
}

// Function to list all books in the library
void listBooks(const vector<Book> &library)
{
    cout << "----- List of Books -----\n";
    for (const Book &book : library)
    {
        cout << "Book ID: " << book.id << endl;
        cout << "Title: " << book.title << endl;
        cout << "Author: " << book.author << endl;
        cout << "Status: " << (book.available ? "Available" : "Issued") << endl;
        cout << "------------------------\n";
    }
}
```

```
// Function to delete a book from the library
void deleteBook(vector<Book> &library)
{
    int bookId;
    cout << "Enter Book ID to Delete: ";
    cin >> bookId;

    for (auto it = library.begin(); it != library.end(); ++it)
    {
        if (it->id == bookId)
        {
            library.erase(it);
            cout << "Book deleted successfully!\n";
            return;
        }
    }

    cout << "Book not found.\n";
}

int main()
{
    vector<Book> library;

    while (true)
    {
        cout << "\n----- Library Management System -----\n";
        cout << "1. Add New Book\n";
        cout << "2. Search for a Book\n";
        cout << "3. Issue a Book\n";
        cout << "4. Return a Book\n";
        cout << "5. List All Books\n";
        cout << "6. Delete a Book\n";
        cout << "7. Exit\n";
        cout << "-------------------------------------\n";
        cout << "Enter your choice: ";

        int choice;
        cin >> choice;

        switch (choice)
        {
        case 1:
            addBook(library);
            break;
        case 2:
            searchBook(library);
            break;
        case 3:
            issueBook(library);
            break;
        case 4:
            returnBook(library);
            break;
        case 5:
            listBooks(library);
            break;
        case 6:
            deleteBook(library);
```

```
                break;
            case 7:
                cout << "Thank you for using the Library Management System.\n";
                return 0;
            default:
                cout << "Invalid choice. Please try again.\n";
            }
        }

        return 0;
    }
```