**University Library Management System**

[INTRODUCTION 2](#_Toc132934850)

[PROJECT DESCRIPTION 3](#_Toc132934851)

[TOOLS AND TECHNOLOGY 4](#_Toc132934852)

[MODULES AND METHODS 5](#_Toc132934853)

[USER REQUIREMENTS 6](#_Toc132934854)

[DESIGN AND IMPLEMENTATION 7](#_Toc132934855)

[E-R Diagram 7](#_Toc132934856)

[FUNCTIONAL REQUIREMENTS 8](#_Toc132934857)

[NON-FUNCTIONAL REQUIREMENTS 9](#_Toc132934858)

[SCOPE 9](#_Toc132934859)

[CHALLENGES 9](#_Toc132934860)

[APPENDIX 10](#_Toc132934861)

[FRONT END SCREENSHOTS 10](#_Toc132934862)

[BACK END SCREENSHOTS 15](#_Toc132934863)

[REFERENCES 19](#_Toc132934864)

# INTRODUCTION

The purpose of the “University Library Management System” is to exclusively provide a system that is user friendly, reliable, to library users. In addition, this will provide the user with capabilities such as adding, editing, deleting books, etc. The system will have to modules: the User and Admin. The admin module will be able to manage the library users in terms of adding, editing, and deleting users. The users will also be able to view details of all entities, search for certain books according to certain criteria’s and also check out and return books. The systems data processing will be adequately fast.

# PROJECT DESCRIPTION

The university library management system is going to be implemented for the users who are the students and the admin.in detail, the main function of the system is to register and store users book records. User will be able to access their book records at the click of a button. They can update their records, checkout, return books, from the system. The user first has to login to enter the system or they have to sign up and create an account. The admin will be able to view all the details of the users and their records of books.

# TOOLS AND TECHNOLOGY

I have used the following:

* Visual Studio: This is the IDE I used for the project.
* Windows Forms: This is the UI framework I used for building and designing the system.
* C#: I used this programming language for the back-end part of the system and integrated it with visual studio IDE.
* SQL Server management studio: I went with this as my preferred database and this is where all the data will be stored.

# MODULES AND METHODS

**User:**

* Signup and Login on the library system
* View the dashboard and choose what to do
* Add, View book, return books, check out books, exit

**Admin:**

* Add and view user
* Update and delete specific user
* Issue books to user

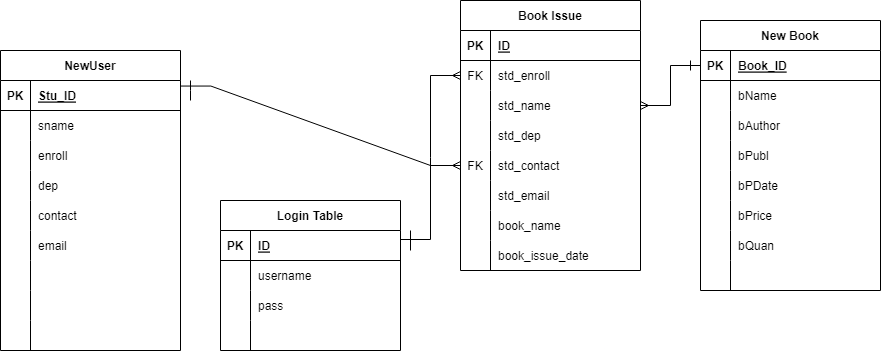
# USER REQUIREMENTS

|  |  |
| --- | --- |
| **User**  **Requirements:** |  |
| Login | All users shall be able to login to the system with a valid username and password. |
| Signup | Users must first sign up in order to be added to the system database. |
| View Dashboard | Users shall be able to view the dashboard once signed into the system. |
| Add book | Users are able to select and add books of their choice. |
| View book | Users will be able to view the books that they have added. |
| Return book | The users will be able to return the books once finished with them. |
| Checkout book | The user will also be able to check out the book that they have selected. |
| Exit | Lastly the user can exit the system after they are done. |

# DESIGN AND IMPLEMENTATION

For the system I mainly worked with windows forms and the .NET framework to create the layout design and for the backend part of the system I worked with C#, for the database side of stuff I worked with Microsoft SQL server management studio to implement it and for the database reference I designed an E-R diagram to show the layout of the database using draw.io software.

## E-R Diagram



# FUNCTIONAL REQUIREMENTS

* Users should be able to sign up into the system.
* Users should be able to login to the system with their username and password.
* Users should be able to view the dashboard.
* Users should be able to add books.
* Users should be able to view books.
* Users should be able to return books,
* Users should be able to check out books.
* Admin should be able to add a user.
* Admin should be able to view user records.

# NON-FUNCTIONAL REQUIREMENTS

* Understandability: The interface of the system should be user friendly and easy to understand.
* Accuracy: the system should be able to display all books that are open for selection.
* Security: The system should be able to secure the user private information from unauthorized accesses.
* Speed: The system should be fast enough so that a user should not have to wait for a page to load for more than 5 seconds.

# SCOPE

* Provides ease and availability of services to the users at all time.
* The users can access the system anywhere and anytime.
* The user can view, update and delete books.

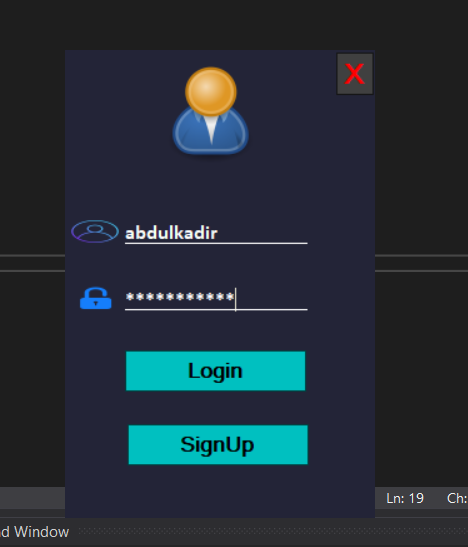
# CHALLENGES

* On working on this project, I have no experience with API and working with any projects that includes them, so on working on this project I had created a system but not an API system as I have to study on it and learn about it, so to this I am currently learning about API’s and trying to create projects including them.
* I had encountered a problem with the database as on the checkout and return pages were not responding to the databases and this problem caught up when I included “**ExecuteNonQuery”** it was giving me an error and it was not transferring the data to the database.

# APPENDIX

## FRONT END SCREENSHOTS

**Login Page:**

****

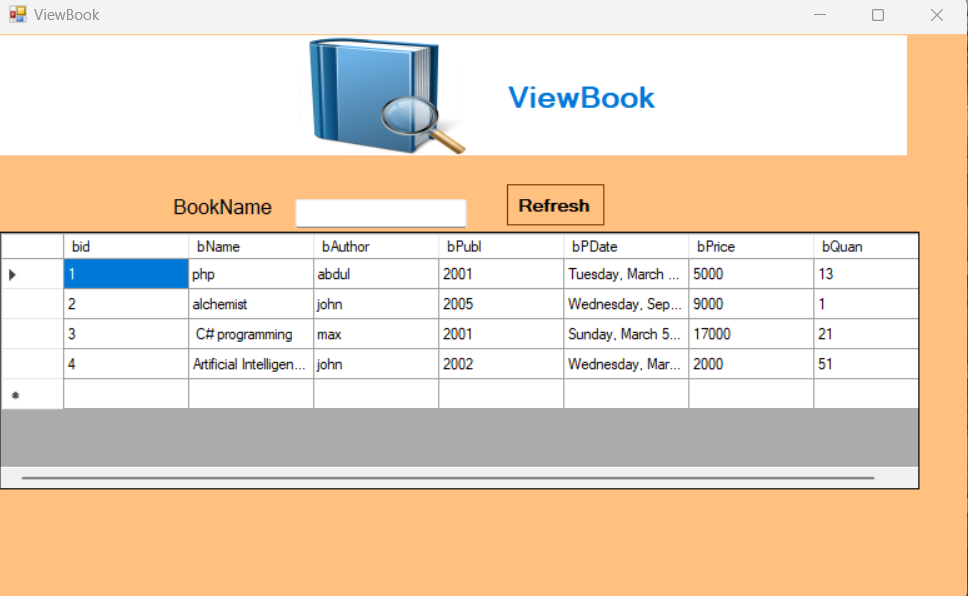
**Library Management System Dashboard:**



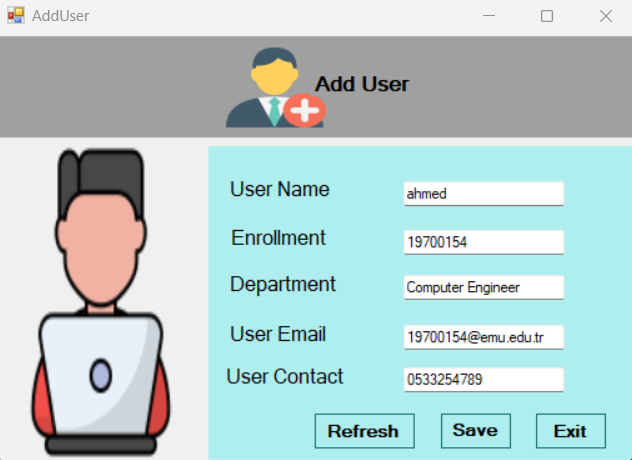
**Add Books:**



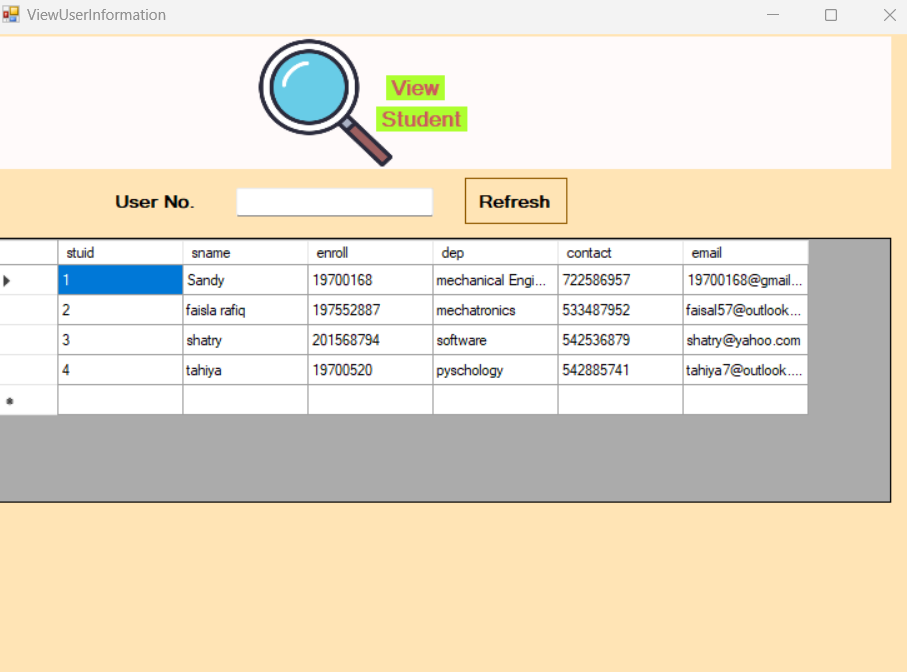
**View Book:**



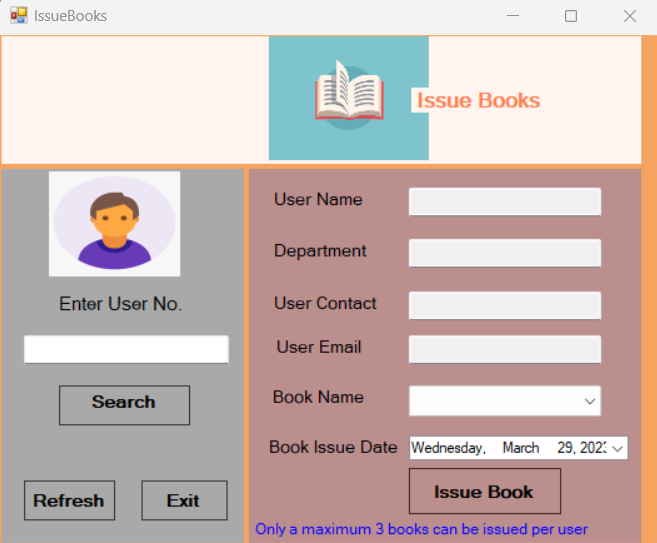
**Add User:**



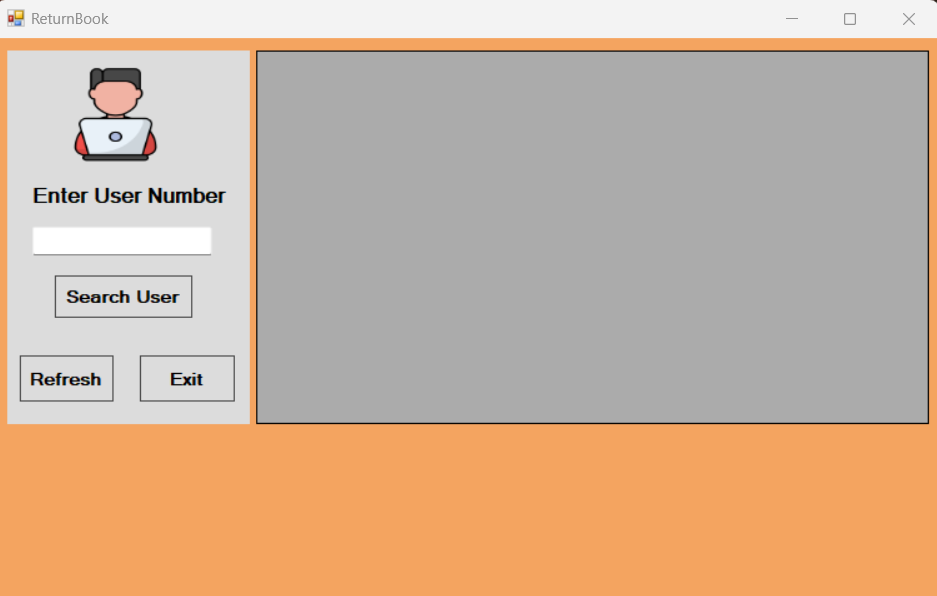
**View User:**



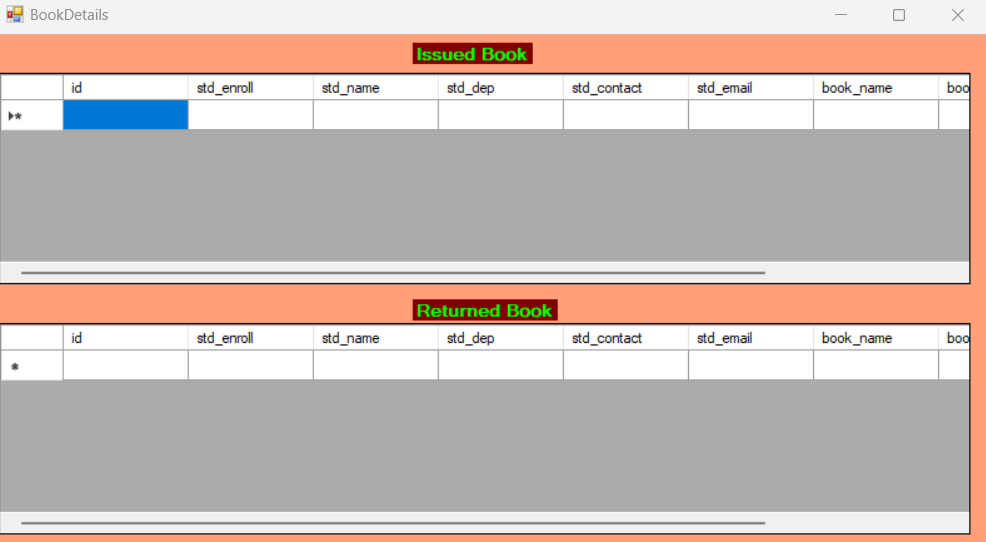
**Issue Books:**



**Return Book:**

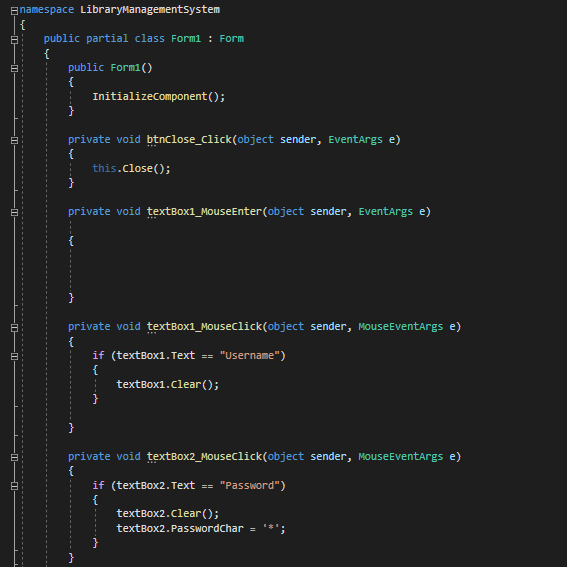


**Check Out Books:**

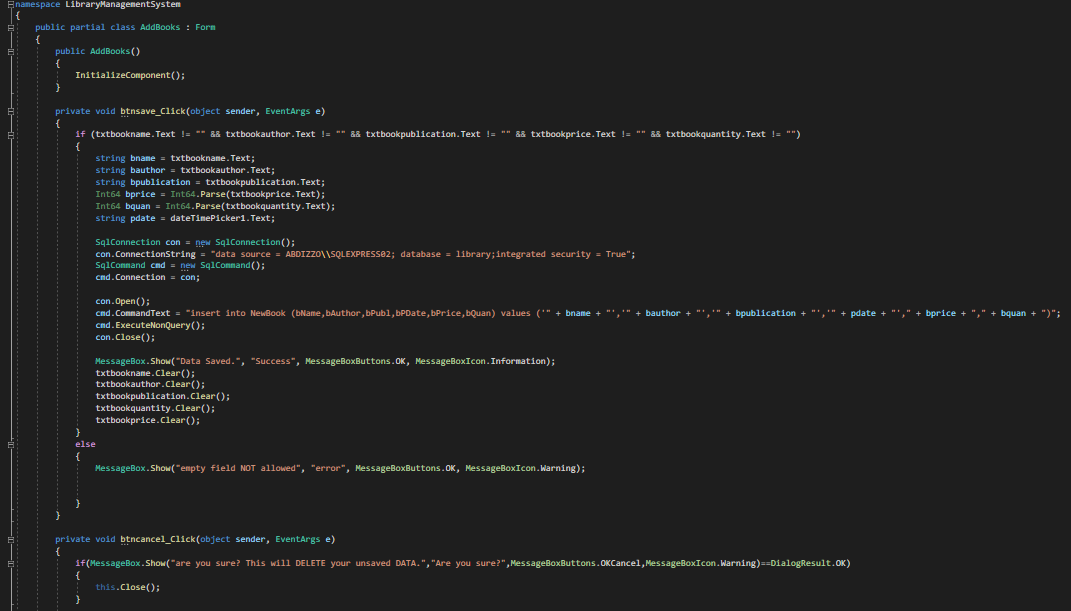


# BACK END SCREENSHOTS

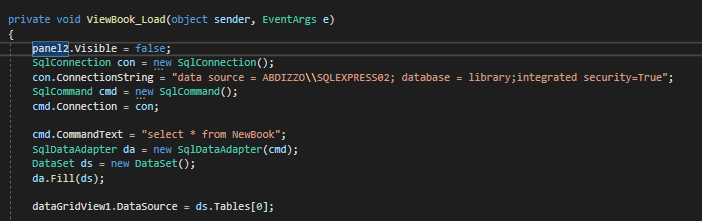
**Login Table:**



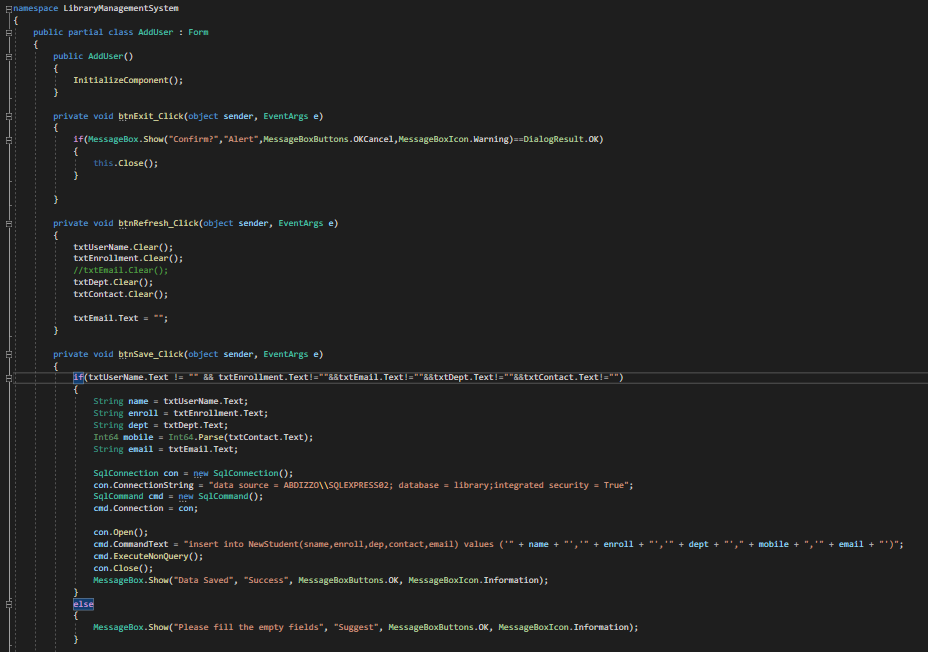
**Add Books:**



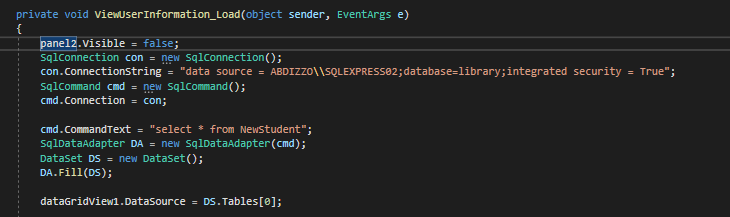
**View Book Code:**



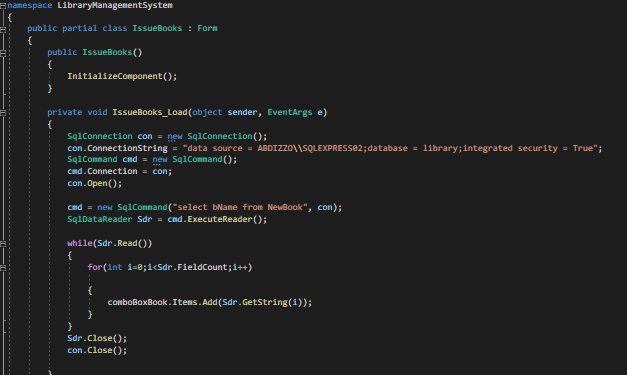
**Add User Code:**



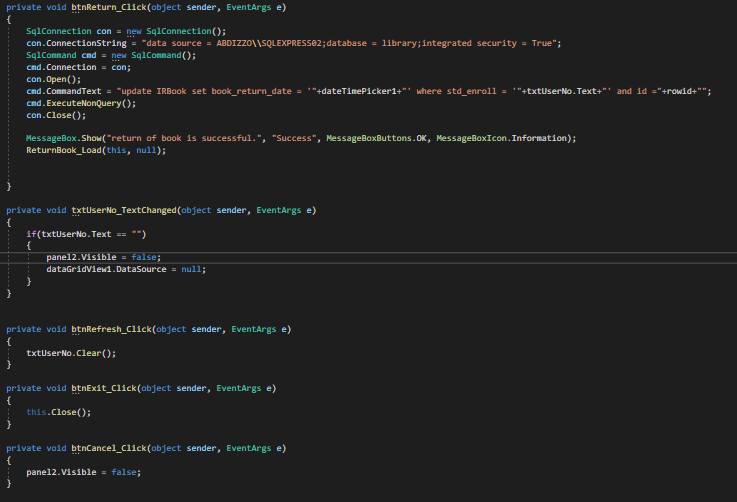
**View User Code:**



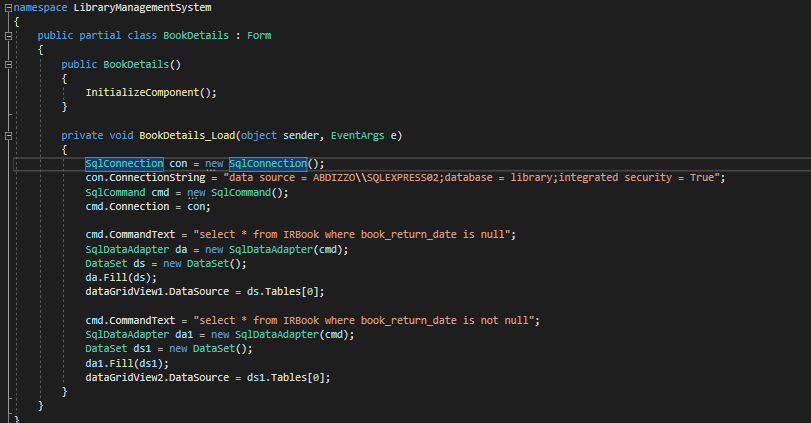
**Issue Books Code:**



**Return Books Code:**



**Check Out Books code:**



# REFERENCES

1. Notion – The all-in-one workspace for your notes, tasks, wikis, and databases. (2023). Notion. <https://sugared-collar-f8f.notion.site/API-Implementation-in-C-9cd5698c197541b081e59dfcdd0c3327>
2. Gaurav Chandak. (2021, May 31). Introduction to Clean Code and Software Design Principles. @Workat\_tech; https://workat.tech. <https://workat.tech/machine-coding/tutorial/introduction-clean-code-software-design-principles-nwu4qqc63e09>
3. BillWagner. (2023). C# docs - get started, tutorials, reference. Microsoft.com. <https://learn.microsoft.com/en-us/dotnet/csharp/>
4. markingmyname. (2023, March 31). SQL Server Management Studio (SSMS) - SQL Server Management Studio (SSMS). Microsoft.com. <https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16>
5. Days, Bt. (2020). Library Management System in Csharp (C#, Visual Studio, MsSQL Server) Complete Project(Step by Step) [YouTube Video]. In YouTube. <https://www.youtube.com/watch?v=YhAwNITpnno>