# Beginner-Level Roadmap to Build a Simple Books Library Management System

## Project Goal

Create a basic **Books Library Management System** with these features:

1. Add books.
2. Remove books.
3. View all books.
4. Borrow books (assign a book to a person's name).
5. Show borrowed books (display borrowed books with names).
6. Return books (move the book back to the available collection).

## Beginner-Friendly Step-by-Step Learning Plan

### **1. Start with the Basics: Classes and Objects**

**What to Learn:**

* + A **class** is a blueprint for objects.
  + **Properties** are like variables inside a class to store information.

**Practical Task:**

* + Create a Book class with these properties:
    - Title (book's name)
    - Author (author's name)
    - IsBorrowed (true or false to check if it's borrowed)
    - BorrowedBy (name of the person who borrowed it)

### **2. Learn Lists to Store Multiple Books**

* **What to Learn:**
  + A **List** is used to store multiple items (in this case, books).
  + Learn how to add, remove, and loop through items in a list.
* **Practical Task:**
  + Create a list called availableBooks to hold all books in the library.
  + Create another list called borrowedBooks to store books that are currently borrowed.

### **3. Learn Methods to Perform Actions**

**What to Learn:**

* + A **method** is a reusable block of code that performs a specific task (e.g., add, remove, borrow a book).

**Practical Task:**

* + Write these methods in a Library class:
    1. AddBook() – Add a book to availableBooks.
    2. RemoveBook() – Remove a book from availableBooks.
    3. ViewAllBooks() – Loop through availableBooks and display the books.

### **4. Learn Console Input/Output for User Interaction**

**What to Learn:**

* + Use Console.ReadLine() to take input from the user.
  + Use Console.WriteLine() to display messages to the user.

**Practical Task:**

* + Allow the user to:
    - Enter the title and author when adding a book.
    - Enter the book's name when borrowing or returning it.

### **5. Learn Conditional Logic for Borrowing and Returning**

**What to Learn:**

* + Use if statements to check conditions.

**Practical Task:**

* + Borrow a book:
    - Check if the book is in availableBooks.
    - If yes, move it to borrowedBooks and assign the borrower's name.
    - If no, display a message saying the book is unavailable.
  + Return a book:
    - Check if the book is in borrowedBooks.
    - If yes, move it back to availableBooks.

### **6. Learn Loops to Display Books**

**What to Learn:**

* + Use foreach or for loops to go through all books in a list.

**Practical Task:**

* + Write a method to display all books in availableBooks and borrowedBooks.

## Plan for Building the System

### **Day 1-2: Classes and Objects**

* Create a Book class with properties: Title, Author, IsBorrowed, BorrowedBy.
* Create a Library class to manage books.

### **Day 3-4: Storing Books in Lists**

* Create lists for availableBooks and borrowedBooks.
* Write methods to add and remove books.

### **Day 5: Borrowing and Returning Books**

* Write methods for BorrowBook and ReturnBook.
* Use conditionals (if) to check if a book is available or already borrowed.

### **Day 6: Displaying Books**

* Write methods to show:
  + All available books.
  + All borrowed books with the names of the borrowers.

### **Day 7: User Interaction**

* Use Console.ReadLine() to take user input.
* Display menu options (add, remove, borrow, return, view books).

## Concepts Summary

1. **Classes and Objects:**
   * Represent books and the library as objects with properties and methods.
2. **Lists:**
   * Store and manage collections of books.
3. **Methods:**
   * Write reusable blocks of code for each feature (add, remove, borrow, return).
4. **Conditional Logic:**
   * Check conditions before performing actions (e.g., "Is the book available?").
5. **Loops:**
   * Use loops to display all books in the library.
6. **Console Input/Output:**
   * Take user input and display results.

## Simplified System Features (for Beginners)

1. **Add Book:**
   * Take title and author as input, then add the book to the availableBooks list.
2. **Remove Book:**
   * Take the book's name as input, find it in availableBooks, and remove it.
3. **Borrow Book:**
   * Take the book's name and borrower's name as input.
   * Check if the book exists in availableBooks.
   * Move it to borrowedBooks with the borrower's name.
4. **Return Book:**
   * Take the book's name as input.
   * Check if it exists in borrowedBooks.
   * Move it back to availableBooks.
5. **View All Books:**
   * Show all books in both availableBooks and borrowedBooks.

## Tools You’ll Use

* **C# Language Basics:**  
  Learn the syntax for classes, lists, methods, loops, and conditionals.
* **Visual Studio Code:**  
  Use VS Code to write, debug, and run your program.
* **Console Application:**  
  Create a simple console-based interface for the library system.

using System;

using System.Collections.Generic;

public class Books

{

public string Title { get; set; }

public string Author { get; set; }

public Books(string title, string author)

{

Title = title;

Author = author;

}

}

public class Customer

{

public string Name { get; set; }

public Customer(string name)

{

Name = name;

}

}

public class Library

{

private List<Books> books = new List<Books>(); // List to store all books

private List<Books> borrowedBooks = new List<Books>(); // List to store borrowed books

// Add a book to the library

public void AddBook()

{

Console.Write("Enter the title of the book: ");

string title = Console.ReadLine();

Console.Write("Enter the author of the book: ");

string author = Console.ReadLine();

Books newBook = new Books(title, author);

books.Add(newBook);

Console.WriteLine($"The book '{title}' by {author} has been added to the library.");

}

// Remove a book from the library

public void RemoveBook()

{

Console.Write("Enter the title of the book to remove: ");

string titleToRemove = Console.ReadLine();

bool bookFound = false;

// Search for the book by title using a simple loop

foreach (var book in books)

{

if (book.Title == titleToRemove)

{

books.Remove(book);

Console.WriteLine($"The book '{titleToRemove}' has been removed from the library.");

bookFound = true;

break; // Exit the loop once the book is found and removed

}

}

if (!bookFound)

{

Console.WriteLine("The book you entered does not exist in the library.");

}

}

// Borrow a book from the library

public void BorrowBook()

{

Console.Write("Enter your name: ");

string customerName = Console.ReadLine();

Console.Write("Enter the title of the book to borrow: ");

string bookTitle = Console.ReadLine();

bool bookFound = false;

// Find the book in the library

foreach (var book in books)

{

if (book.Title == bookTitle)

{

borrowedBooks.Add(book);

books.Remove(book);

Console.WriteLine($"{customerName} borrowed the book '{bookTitle}'.");

bookFound = true;

break;

}

}

if (!bookFound)

{

Console.WriteLine($"Sorry, the book '{bookTitle}' is not available.");

}

}

// Return a borrowed book to the library

public void ReturnBook()

{

Console.Write("Enter your name: ");

string customerName = Console.ReadLine();

Console.Write("Enter the title of the book to return: ");

string bookTitle = Console.ReadLine();

bool bookFound = false;

// Find the borrowed book to return

foreach (var book in borrowedBooks)

{

if (book.Title == bookTitle)

{

borrowedBooks.Remove(book);

books.Add(book);

Console.WriteLine($"{customerName} returned the book '{bookTitle}'.");

bookFound = true;

break;

}

}

if (!bookFound)

{

Console.WriteLine($"The book '{bookTitle}' is not borrowed.");

}

}

// Show all available books

public void ShowAvailableBooks()

{

if (books.Count == 0)

{

Console.WriteLine("No books available in the library.");

}

else

{

Console.WriteLine("\nAvailable Books in Library:");

foreach (var book in books)

{

Console.WriteLine($"- {book.Title} by {book.Author}");

}

}

}

// Show all borrowed books

public void ShowBorrowedBooks()

{

if (borrowedBooks.Count == 0)

{

Console.WriteLine("No books are currently borrowed.");

}

else

{

Console.WriteLine("\nBorrowed Books:");

foreach (var book in borrowedBooks)

{

Console.WriteLine($"- {book.Title} by {book.Author}");

}

}

}

}

public class Program

{

static void Main(string[] args)

{

Library library = new Library();

while (true)

{

Console.WriteLine("\nLibrary Menu:");

Console.WriteLine("1. Add Book");

Console.WriteLine("2. Remove Book");

Console.WriteLine("3. Borrow Book");

Console.WriteLine("4. Return Book");

Console.WriteLine("5. View Available Books");

Console.WriteLine("6. View Borrowed Books");

Console.WriteLine("7. Exit");

Console.Write("Choose an option: ");

string option = Console.ReadLine();

switch (option)

{

case "1":

library.AddBook();

break;

case "2":

library.RemoveBook();

break;

case "3":

library.BorrowBook();

break;

case "4":

library.ReturnBook();

break;

case "5":

library.ShowAvailableBooks();

break;

case "6":

library.ShowBorrowedBooks();

break;

case "7":

return; // Exit the program

default:

Console.WriteLine("Invalid choice. Try again.");

break;

}

}

}

}