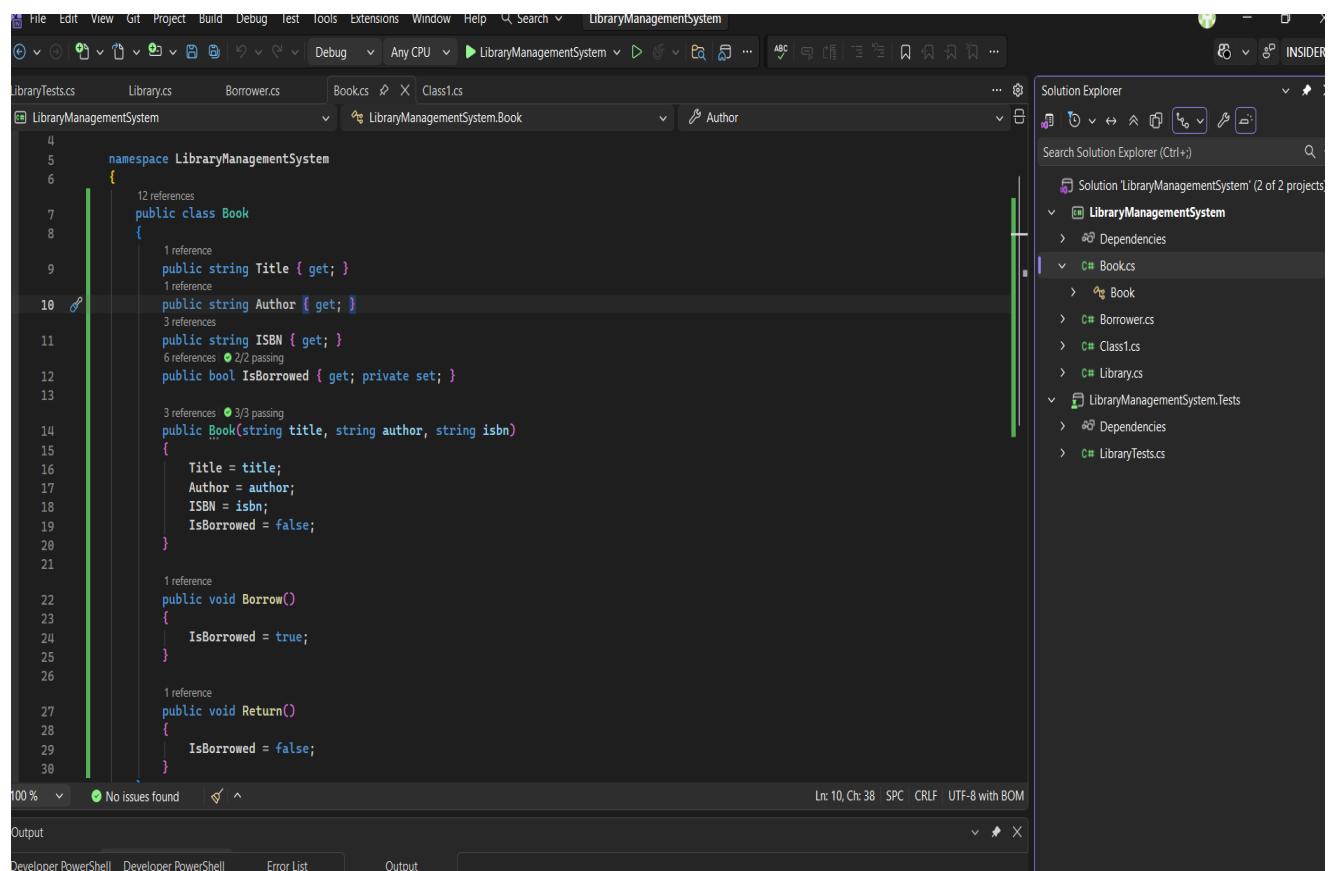


STEP 1: Create the Solution

1. Click Create a new project
2. Select Class Library (.NET)
3. Project Name: LibraryManagementSystem
4. This project will contain Book, Borrower, and Library classes

STEP 2: Create Required Classes



The screenshot shows the Microsoft Visual Studio interface. The code editor on the left displays the `Book.cs` file for the `LibraryManagementSystem` project. The file contains the following C# code:

```
4
5     namespace LibraryManagementSystem
6     {
7         public class Book
8         {
9             public string Title { get; }
10            public string Author { get; }
11            public string ISBN { get; }
12            public bool IsBorrowed { get; private set; }
13
14            public Book(string title, string author, string isbn)
15            {
16                Title = title;
17                Author = author;
18                ISBN = isbn;
19                IsBorrowed = false;
20            }
21
22            public void Borrow()
23            {
24                IsBorrowed = true;
25            }
26
27            public void Return()
28            {
29                IsBorrowed = false;
30            }
31        }
32    }
```

The Solution Explorer on the right shows the project structure:

- `LibraryManagementSystem`:
 - Dependencies
 - `Book.cs`
 - `Borrower.cs`
 - `Class1.cs`
 - `Library.cs`
- `LibraryManagementSystem.Tests`:
 - Dependencies
 - `LibraryTests.cs`

Book.cs Code :

```
using System;
using System.Collections.Generic;
using System.Text;
namespace LibraryManagementSystem
{
    public class Book
    {
        public string Title { get; }
        public string Author { get; }
        public string ISBN { get; }
        public bool IsBorrowed { get; private set; }

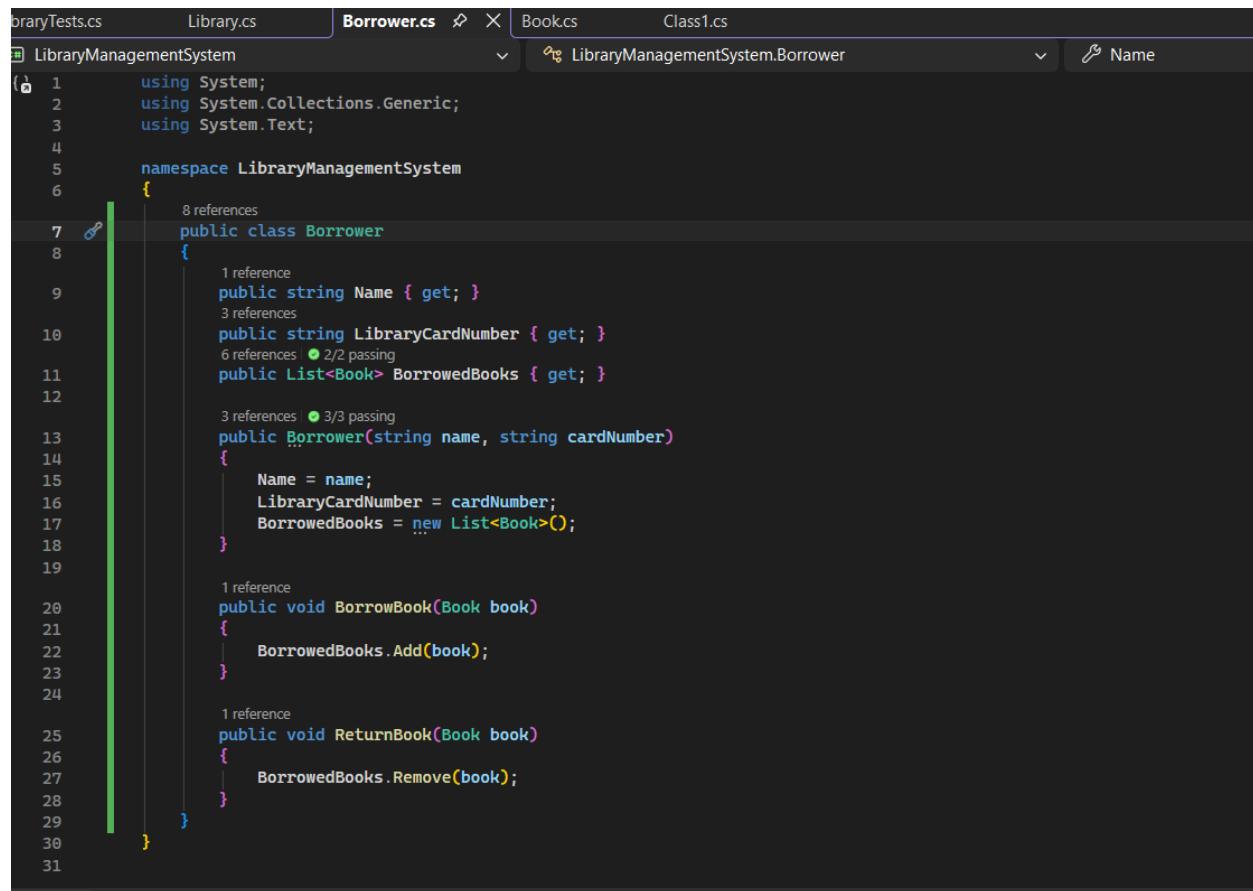
        public Book(string title, string author, string isbn)
        {
            Title = title;
            Author = author;
            ISBN = isbn;
            IsBorrowed = false;
        }

        public void Borrow()
        {
            IsBorrowed = true;
        }
    }
}
```

```
public void Return()
{
    IsBorrowed = false;
}

}
```

Borrower.cs



```
libraryTests.cs      Library.cs      Borrower.cs      Book.cs      Class1.cs
LibraryManagementSystem
Borrower.cs
namespace LibraryManagementSystem
{
    public class Borrower
    {
        public string Name { get; }
        public string LibraryCardNumber { get; }
        public List<Book> BorrowedBooks { get; }

        public Borrower(string name, string cardNumber)
        {
            Name = name;
            LibraryCardNumber = cardNumber;
            BorrowedBooks = new List<Book>();
        }

        public void BorrowBook(Book book)
        {
            BorrowedBooks.Add(book);
        }

        public void ReturnBook(Book book)
        {
            BorrowedBooks.Remove(book);
        }
    }
}
```

Borrower.cs Code:

```
using System;
using System.Collections.Generic;
using System.Text;
namespace LibraryManagementSystem
{
    public class Borrower
    {
        public string Name { get; }
        public string LibraryCardNumber { get; }
        public List<Book> BorrowedBooks { get; }

        public Borrower(string name, string cardNumber)
        {
            Name = name;
            LibraryCardNumber = cardNumber;
            BorrowedBooks = new List<Book>();
        }

        public void BorrowBook(Book book)
        {
            BorrowedBooks.Add(book);
        }

        public void ReturnBook(Book book)
        {
    }
```

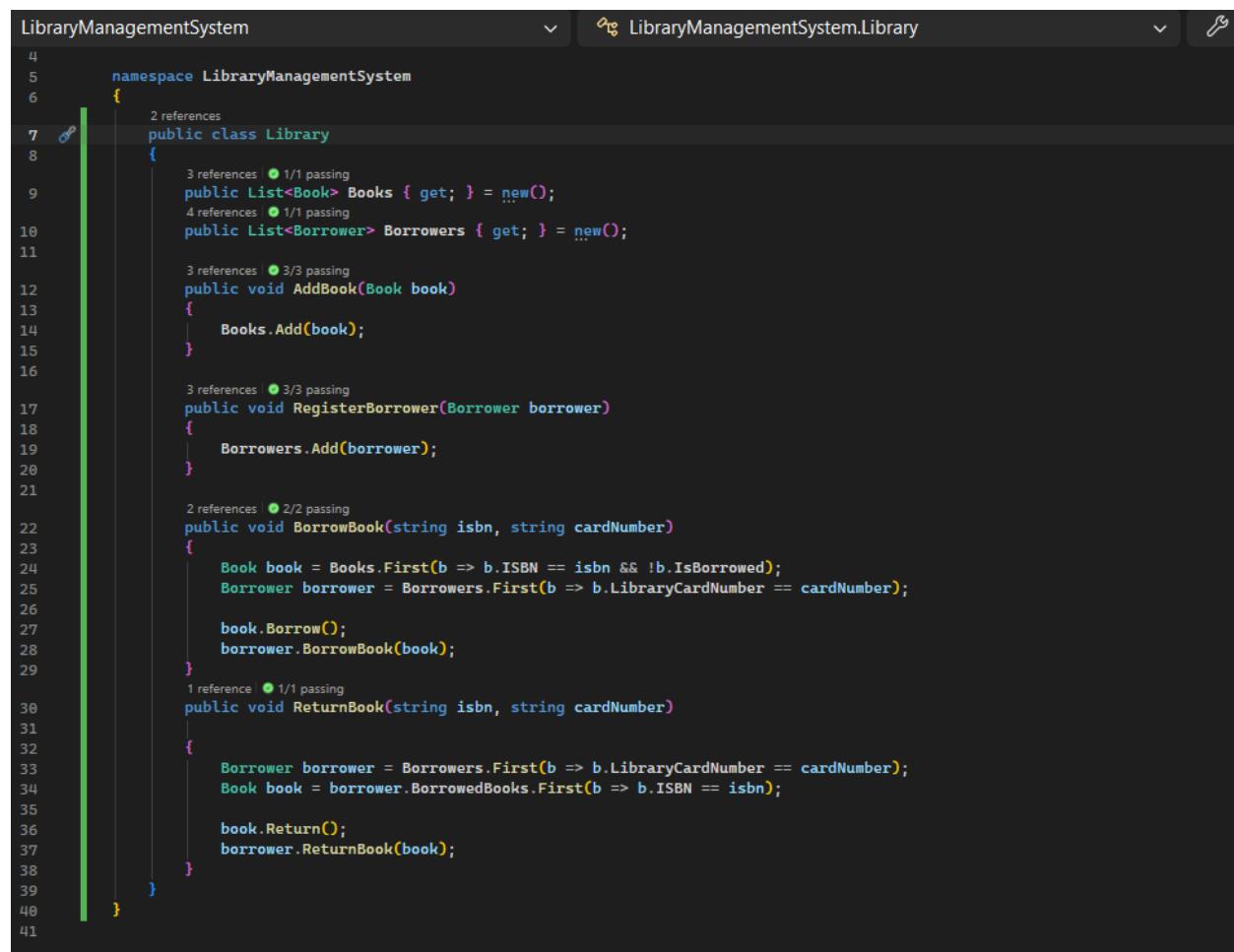
```
BorrowedBooks.Remove(book);

}

}

}
```

Library.cs



The screenshot shows a code editor window with the following details:

- Project:** LibraryManagementSystem
- Namespace:** LibraryManagementSystem
- File:** Library.cs
- Code Content:**

```
4
5     namespace LibraryManagementSystem
6 {
7         public class Library
8         {
9             public List<Book> Books { get; } = new();
10            public List<Borrower> Borrowers { get; } = new();
11
12            public void AddBook(Book book)
13            {
14                Books.Add(book);
15            }
16
17            public void RegisterBorrower(Borrower borrower)
18            {
19                Borrowers.Add(borrower);
20            }
21
22            public void BorrowBook(string isbn, string cardNumber)
23            {
24                Book book = Books.First(b => b.ISBN == isbn && !b.IsBorrowed);
25                Borrower borrower = Borrowers.First(b => b.LibraryCardNumber == cardNumber);
26
27                book.Borrow();
28                borrower.BorrowBook(book);
29            }
30
31            public void ReturnBook(string isbn, string cardNumber)
32            {
33                Borrower borrower = Borrowers.First(b => b.LibraryCardNumber == cardNumber);
34                Book book = borrower.BorrowedBooks.First(b => b.ISBN == isbn);
35
36                book.Return();
37                borrower.ReturnBook(book);
38            }
39        }
40    }
```
- Toolbars:** Standard Windows-style toolbar with icons for file operations.
- Status Bar:** Shows the current file path: LibraryManagementSystem.Library.

Library .cs Code

```
namespace LibraryManagementSystem

{
    public class Library
    {
        public List<Book> Books { get; } = new();
        public List<Borrower> Borrowers { get; } = new();

        public void AddBook(Book book)
        {
            Books.Add(book);
        }

        public void RegisterBorrower(Borrower borrower)
        {
            Borrowers.Add(borrower);
        }

        public void BorrowBook(string isbn, string cardNumber)
        {
            Book book = Books.First(b => b.ISBN == isbn && !b.IsBorrowed);
            Borrower borrower = Borrowers.First(b => b.LibraryCardNumber == cardNumber);

            book.Borrow();
            borrower.BorrowBook(book);
        }
    }
}
```

```
public void ReturnBook(string isbn, string cardNumber)

{

    Borrower borrower = Borrowers.First(b => b.LibraryCardNumber == cardNumber);

    Book book = borrower.BorrowedBooks.First(b => b.ISBN == isbn);

    book.Return();

    borrower.ReturnBook(book);

}

}

}
```

STEP 3: Create NUnit Test Project

1. Right-click Solution → Add → New Project
2. Select NUnit Test Project
3. Project Name: LibraryManagementSystem.Tests

STEP 4: Add Reference to Class Library

1. Right-click Test Project
2. Click Add Project Reference
3. Select LibraryManagementSystem

The screenshot shows the Visual Studio IDE interface with the following details:

- Menu Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Tools, Extensions, Window, Help.
- Toolbar:** Standard icons for file operations like Open, Save, Print, etc.
- Solution Explorer:** Shows the project structure with files: LibraryTests.cs, Library.cs, Borrower.cs, Book.cs, and Class1.cs.
- Toolbox:** Standard .NET development tools.
- Status Bar:** Shows "91 %", "No issues found", and a small icon.
- Code Editor:** Displays the content of LibraryTests.cs, which contains NUnit test code for a LibraryManagementSystem.

```
1  using NUnit.Framework;
2  using LibraryManagementSystem;
3
4  namespace LibraryManagementSystem.Tests
5  {
6      [TestFixture]
7      0 references
8      public class LibraryTests
9      {
10         private Library _library;
11
12         [SetUp]
13         0 references
14         public void Setup()
15         {
16             _library = new Library();
17
18             [Test]
19             0 references
20             public void AddBook_BookIsAddedSuccessfully()
21             {
22                 var book = new Book("C# Basics", "Microsoft", "ISBN001");
23                 _library.AddBook(book);
24
25                 Assert.That(_library.Books.Count, Is.EqualTo(1));
26             }
27
28             [Test]
29             0 references
30             public void RegisterBorrower_BorrowerIsRegistered()
31             {
32                 var borrower = new Borrower("John", "CARD001");
33                 _library.RegisterBorrower(borrower);
34
35                 Assert.That(_library.Borrowers.Count, Is.EqualTo(1));
36             }
37         }
38     }
39 }
```

```
 34
 35     [Test]
 36     public void BorrowBook_BookIsMarkedAsBorrowed()
 37     {
 38         var book = new Book("C#", "Author", "ISBN002");
 39         var borrower = new Borrower("Alice", "CARD002");
 40
 41         _library.AddBook(book);
 42         _library.RegisterBorrower(borrower);
 43
 44         _library.BorrowBook("ISBN002", "CARD002");
 45
 46         Assert.That(book.IsBorrowed, Is.True);
 47         Assert.That(borrower.BorrowedBooks.Count, Is.EqualTo(1));
 48     }
 49
 50
 51     [Test]
 52     public void ReturnBook_BookIsAvailableAgain()
 53     {
 54         var book = new Book("Java", "Author", "ISBN003");
 55         var borrower = new Borrower("Bob", "CARD003");
 56
 57         _library.AddBook(book);
 58         _library.RegisterBorrower(borrower);
 59         _library.BorrowBook("ISBN003", "CARD003");
 60
 61         _library.ReturnBook("ISBN003", "CARD003");
 62
 63         Assert.That(book.IsBorrowed, Is.False);
 64         Assert.That(borrower.BorrowedBooks.Count, Is.EqualTo(0));
 65     }
 66 }
 67 }
```

LibraryTests.cs Code

```
using NUnit.Framework;
using LibraryManagementSystem;
namespace LibraryManagementSystem.Tests
{
    [TestFixture]
    public class LibraryTests
    {
        private Library _library;
```

```
[SetUp]
public void Setup()
{
    _library = new Library();
}

[Test]
public void AddBook_BookIsAddedSuccessfully()
{
    var book = new Book("C# Basics", "Microsoft", "ISBN001");
    _library.AddBook(book);

    Assert.That(_library.Books.Count, Is.EqualTo(1));
}

[Test]
public void RegisterBorrower_BorrowerIsRegistered()
{
    var borrower = new Borrower("John", "CARD001");
    _library.RegisterBorrower(borrower);

    Assert.That(_library.Borrowers.Count, Is.EqualTo(1));
}

[Test]
```

```
public void BorrowBook_BookIsMarkedAsBorrowed()
{
    var book = new Book("C#", "Author", "ISBN002");
    var borrower = new Borrower("Alice", "CARD002");

    _library.AddBook(book);
    _library.RegisterBorrower(borrower);

    _library.BorrowBook("ISBN002", "CARD002");

    Assert.That(book.IsBorrowed, Is.True);
    Assert.That(borrower.BorrowedBooks.Count, Is.EqualTo(1));
}
```

[Test]

```
public void ReturnBook_BookIsAvailableAgain()
{
    var book = new Book("Java", "Author", "ISBN003");
    var borrower = new Borrower("Bob", "CARD003");

    _library.AddBook(book);
    _library.RegisterBorrower(borrower);
    _library.BorrowBook("ISBN003", "CARD003");

    _library.ReturnBook("ISBN003", "CARD003");
```

```

        Assert.That(book.IsBorrowed, Is.False);

        Assert.That(borrower.BorrowedBooks.Count, Is.EqualTo(0));

    }

}

}

```

STEP 5: Run Unit Tests

1. Open Test Explorer
2. Click Run All

