Library Management System

This project is a Library Management System developed in Java. It allows users to manage books, including adding new books, borrowing and returning books, and viewing available books.

The project utilizes object-oriented programming principles to handle various functionalities and follows the Test-Driven Development (TDD) approach to ensure code quality and reliability.

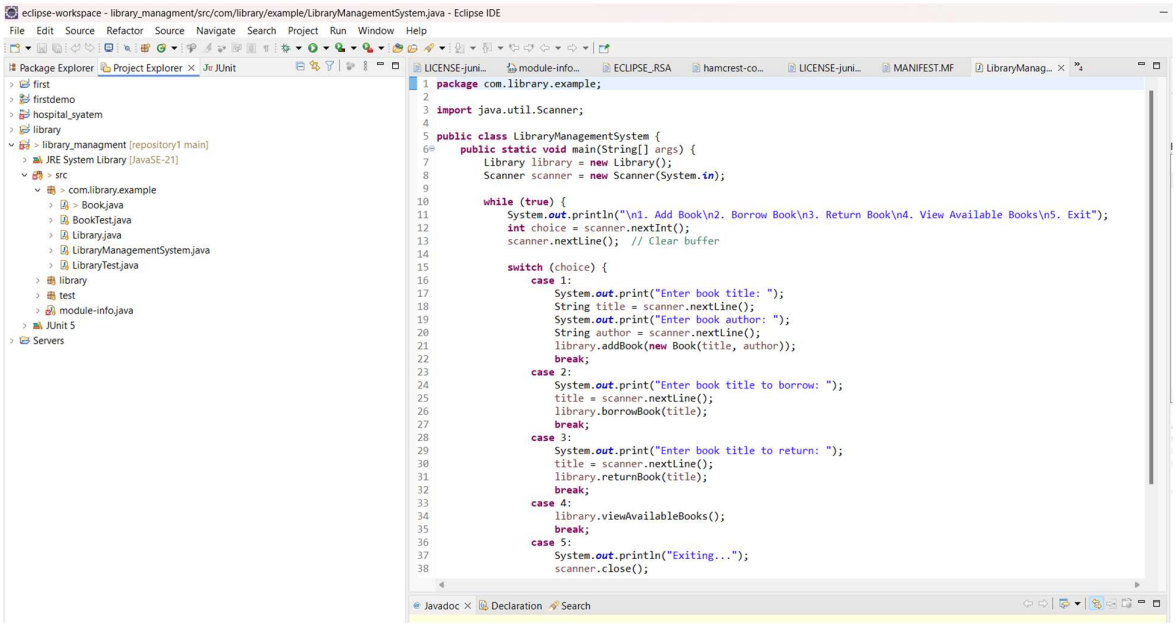
Features - Add Books: Add new books to the library.

Borrow Books: Borrow available books.

Return Books: Return borrowed books.

View Books: View a list of available books.

Project structure in IDE:



Book Class:

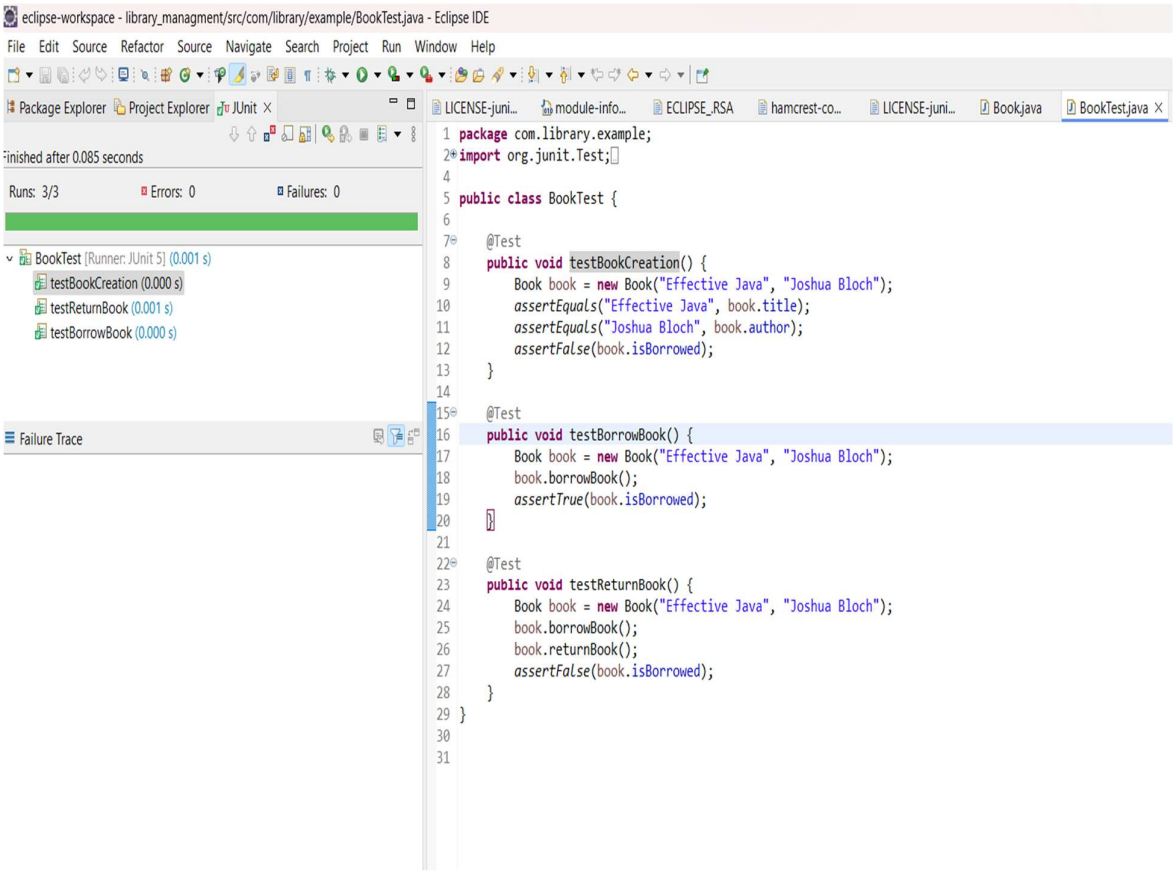
```
package com.library.example;

public class Book {
   String title;
   String author;
   boolean isBorrowed;

   public Book(String title, String author) {
      this.title = title;
      this.author = author;
      this.isBorrowed = false;
   }
```

```
    public void borrowBook() {
        isBorrowed = true;
    }

    public void returnBook() {
        isBorrowed = false;
    }

    public boolean isAvailable() {
        return !isBorrowed;
    }
}
```

BookTest Class for TDD:

```
package com.library.example;
import org.junit.Test;
import static org.junit.Assert.*;

public class BookTest {

    @Test
    public void testBookCreation() {
        Book book = new Book("Effective Java", "Joshua Bloch");
        assertEquals("Effective Java", book.title);
        assertEquals("Joshua Bloch", book.author);
        assertFalse(book.isBorrowed);
    }

    @Test
    public void testBorrowBook() {
        Book book = new Book("Effective Java", "Joshua Bloch");
        book.borrowBook();
        assertTrue(book.isBorrowed);
    }

    @Test
    public void testReturnBook() {
        Book book = new Book("Effective Java", "Joshua Bloch");
        book.borrowBook();
        book.returnBook();
        assertFalse(book.isBorrowed);
    }
}
```

BookTest class junit test result:



Library Class:

```
package com.library.example;

import java.util.ArrayList;

public class Library {
   ArrayList<Book> books = new ArrayList<>();

   public void addBook(Book book) {
      books.add(book);
   }

   public void borrowBook(String title) {
      for (Book book : books) {
         if (book.title.equalsIgnoreCase(title) && book.isAvailable()) {
            book.borrowBook();
            System.out.println("You borrowed: " + title);
            return;
         }
      }
      System.out.println("Book not available.");
   }
```

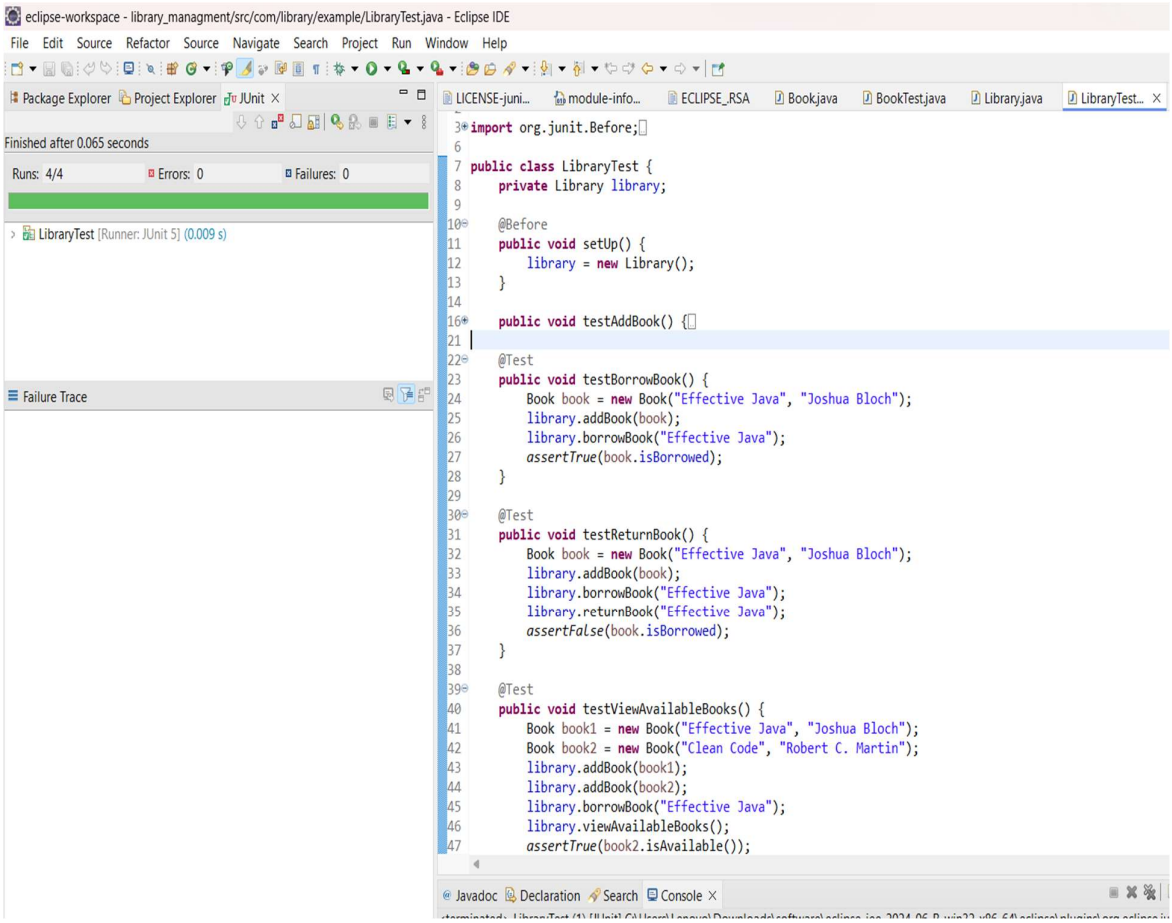
```
    public void returnBook(String title) {
        for (Book book : books) {
            if (book.title.equalsIgnoreCase(title) && !book.isAvailable()) {
                book.returnBook();
                System.out.println("You returned: " + title);
                return;
            }
        }
        System.out.println("Book was not borrowed.");
    }

    public void viewAvailableBooks() {
        System.out.println("Available books:");
        for (Book book : books) {
            if (book.isAvailable()) {
                System.out.println(book.title + " by " + book.author);
            }
        }
    }
}
```

LibraryTest Class for TDD:

```
package com.library.example;

import org.junit.Before;
import org.junit.Test;
import static org.junit.Assert.*;

public class LibraryTest {
    private Library library;

    @Before
    public void setUp() {
        library = new Library();
    }

    @Test
    public void testAddBook() {
        Book book = new Book("Effective Java", "Joshua Bloch");
        library.addBook(book);
        assertTrue(library.books.contains(book));
    }

    @Test
    public void testBorrowBook() {
        Book book = new Book("Effective Java", "Joshua Bloch");
        library.addBook(book);
        library.borrowBook("Effective Java");
        assertTrue(book.isBorrowed);
    }
```

```
@Test
  public void testReturnBook() {
     Book book = new Book("Effective Java", "Joshua Bloch");
     library.addBook(book);
     library.borrowBook("Effective Java");
     library.returnBook("Effective Java");
     assertFalse(book.isBorrowed);
  }

  @Test
  public void testViewAvailableBooks() {
     Book book1 = new Book("Effective Java", "Joshua Bloch");
     Book book2 = new Book("Clean Code", "Robert C. Martin");
     library.addBook(book1);
     library.addBook(book2);
     library.borrowBook("Effective Java");
     library.viewAvailableBooks();
     assertTrue(book2.isAvailable());
     assertFalse(book1.isAvailable());
  }
}
```

LibraryTest class junit test result:



LibraryManagementSystem Class:

```
package com.library.example;

import java.util.Scanner;

public class LibraryManagementSystem {
   public static void main(String[] args) {
      Library library = new Library();
      Scanner scanner = new Scanner(System.in);

      while (true) {
         System.out.println("\n1. Add Book\n2. Borrow Book\n3. Return Book\n4. View Available
Books\n5. Exit");
         int choice = scanner.nextInt();
         scanner.nextLine();  // Clear buffer

         switch (choice) {
            case 1:
               System.out.print("Enter book title: ");
               String title = scanner.nextLine();
               System.out.print("Enter book author: ");
               String author = scanner.nextLine();
               library.addBook(new Book(title, author));
               break;
            case 2:
               System.out.print("Enter book title to borrow: ");
               title = scanner.nextLine();
               library.borrowBook(title);
               break;
            case 3:
               System.out.print("Enter book title to return: ");
               title = scanner.nextLine();
               library.returnBook(title);
               break;
            case 4:
               library.viewAvailableBooks();
               break;
            case 5:
               System.out.println("Exiting...");
               scanner.close();
               return;
            default:
               System.out.println("Invalid choice. Try again.");
         }
      }
   }
}
```

Main class result:

```
1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
1
Enter book title: Glimpses of World History
Enter book author: Rakesh vora

1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
1
Enter book title: The Golden Gate
Enter book author: vikram sheth

1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
1
Enter book title: Great Expectations
Enter book author: williams
```

```
1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
2
Enter book title to borrow: The Golden Gate
You borrowed: The Golden Gate

1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
4
Available books:
Glimpses of World History by Rakesh vora
Great Expectations by williams

1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
3
Enter book title to return: The Golden Gate
You returned: The Golden Gate
```

```
1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
4
Available books:
Glimpses of World History by Rakesh vora
The Golden Gate by vikram sheth
Great Expectations by williams

1. Add Book
2. Borrow Book
3. Return Book
4. View Available Books
5. Exit
5
Exiting...
```

Output Summary

So, I was working on my Library Management System project and here's what I did:

Add Book

First, I added a few books to the library. I added "Glimpses of World History" by Rakesh Vora, "The Golden Gate" by Vikram Seth, and "Great Expectations" by Williams .

Borrow Boo

Then, I decided to test the borrow function. I borrowed "The Golden Gate." The system correctly showed that I borrowed the book.

View Available Books (After Borrowing)

After borrowing, I checked the available books. The system listed "Glimpses of World History" and "Great Expectations," which is right because I had borrowed "The Golden Gate."

Return Book

Next, I tested the return function by returning "The Golden Gate." The system confirmed the return.

View Available Books (After Returning)

Finally, I checked the available books again, and all three books were listed correctly: "Glimpses of World History," "The Golden Gate," and "Great Expectations."