TEST REPORT

Name: Hatim Mullajiwala Date: 26/12/2024

Library Management System

Test Objective: To validate the functionality of the Library Management System, ensuring that users can add, borrow, return, and view available books seamlessly while maintaining data integrity and meeting the specified requirements.

Scope: The test cases will cover the core functionalities of the Library Management System, including:

1. Adding books with valid details and verifying their presence in the database.
2. Borrowing books and ensuring availability status updates correctly.
3. Handling errors for borrowing unavailable or non-existent books.
4. Returning books and validating the update of availability status.
5. Viewing the list of all available books and ensuring it reflects the correct data.

Test type: Unit Testing

Test Environment:

Operating System: Windows 10

Development Tools: Visual Studio Code (VS Code)

Frameworks and Libraries:

- Streamlit: For the application interface.
- SQLAlchemy: For database interaction.
- unittest: For writing and executing test cases.

Database: SQL (SQLite used via SQLAlchemy).

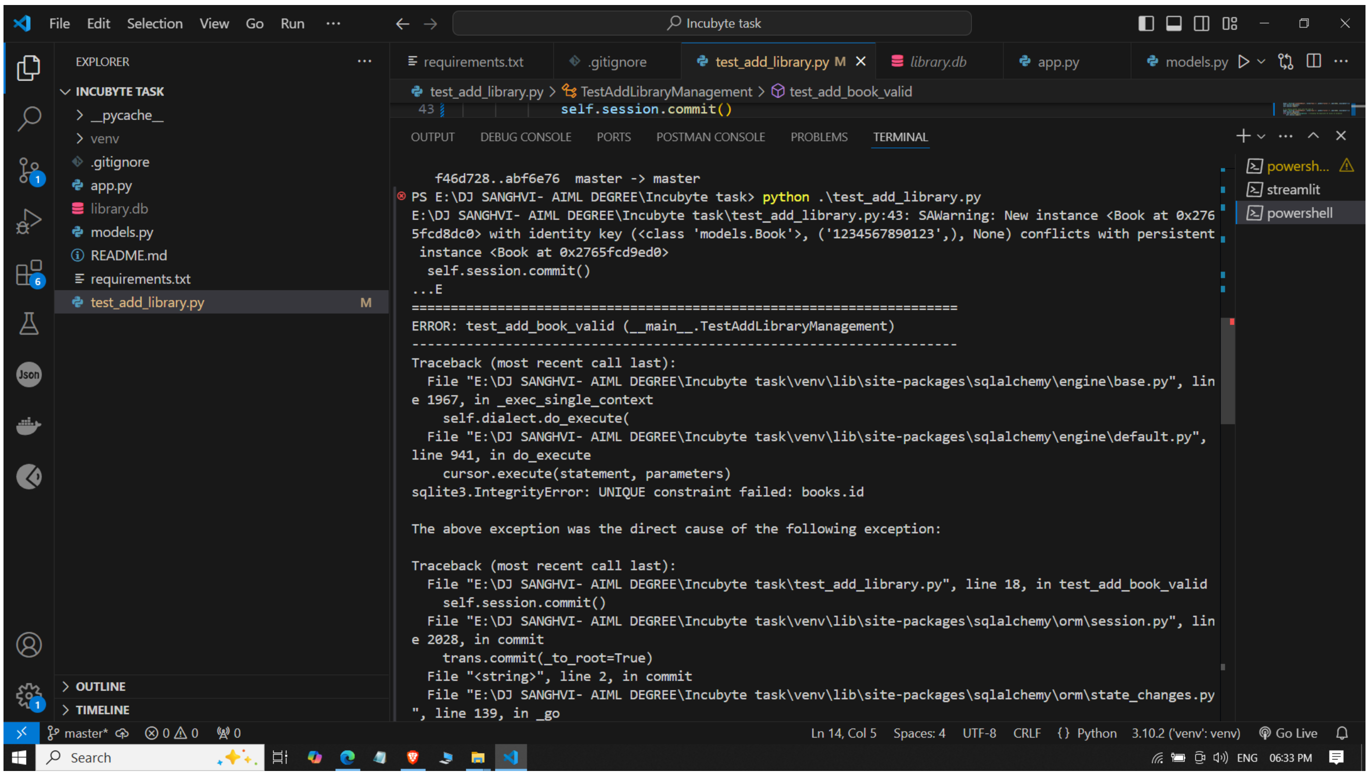
Python Version: Ensure Python 3.x is installed and configured**.**

Test Cases:

1) Adding Books in the library System:

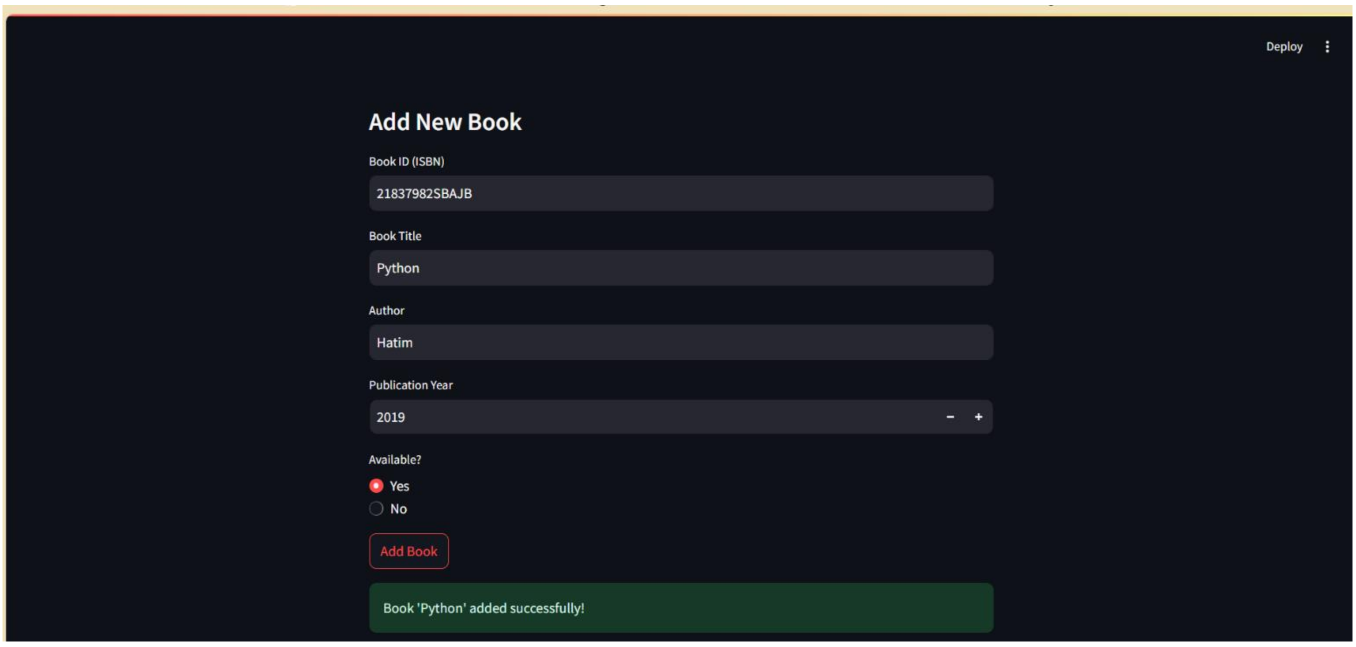
Test Case Id:	Description	Precondition	Steps	Expected Result	Status
1	Adding a Valid Book	All field should be filled	Select "Add Book" and fill in details (ISBN, Title, Author, Year, Available) then click on Add Book.	A new book can only be added if all fields are filled; otherwise, raise an error for	PASS

				empty fields.	
2	Adding a Book with same ISBN number	A Book should be there in the library.	Select "Add Book" and fill in details (ISBN, Title, Author, Year, Available) then click on Add Book. Add a new book with an existing ISBN in the library.	The system should raise an error for duplicate entries.	PASS
3	Add a book with invalid Year of Publication	Add a new book with unique ISBN number.	Select "Add Book" and fill in details (ISBN, Title, Author, Year, Available) then click on Add Book. Add a new book with an invalid year like 1400 or 2030.	The system should raise an error if the publication year is not between 1800 and 2024.	PASS



```
self.session.commit()
```

```
   f46d728..abf6e76  master -> master
PS E:\DJ SANGHVI- AIML DEGREE\Incubyte task> python .\test_add_library.py
E:\DJ SANGHVI- AIML DEGREE\Incubyte task\test_add_library.py:43: SAWarning: New instance <Book at 0x276
5fcd8dc0> with identity key (<class 'models.Book'>, ('1234567890123',), None) conflicts with persistent
 instance <Book at 0x2765fcd9ed0>
  self.session.commit()
...E
======================================================================
ERROR: test_add_book_valid (__main__.TestAddLibraryManagement)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "E:\DJ SANGHVI- AIML DEGREE\Incubyte task\venv\lib\site-packages\sqlalchemy\engine\base.py", lin
e 1967, in _exec_single_context
    self.dialect.do_execute(
  File "E:\DJ SANGHVI- AIML DEGREE\Incubyte task\venv\lib\site-packages\sqlalchemy\engine\default.py",
line 941, in do_execute
    cursor.execute(statement, parameters)
sqlite3.IntegrityError: UNIQUE constraint failed: books.id

The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "E:\DJ SANGHVI- AIML DEGREE\Incubyte task\test_add_library.py", line 18, in test_add_book_valid
    self.session.commit()
  File "E:\DJ SANGHVI- AIML DEGREE\Incubyte task\venv\lib\site-packages\sqlalchemy\orm\session.py", lin
e 2028, in commit
    trans.commit(_to_root=True)
  File "<string>", line 2, in commit
  File "E:\DJ SANGHVI- AIML DEGREE\Incubyte task\venv\lib\site-packages\sqlalchemy\orm\state_changes.py
", line 139, in _go
```



Add New Book

Book ID (ISBN): 21837982SBAJB

Book Title: Python

Author: Hatim

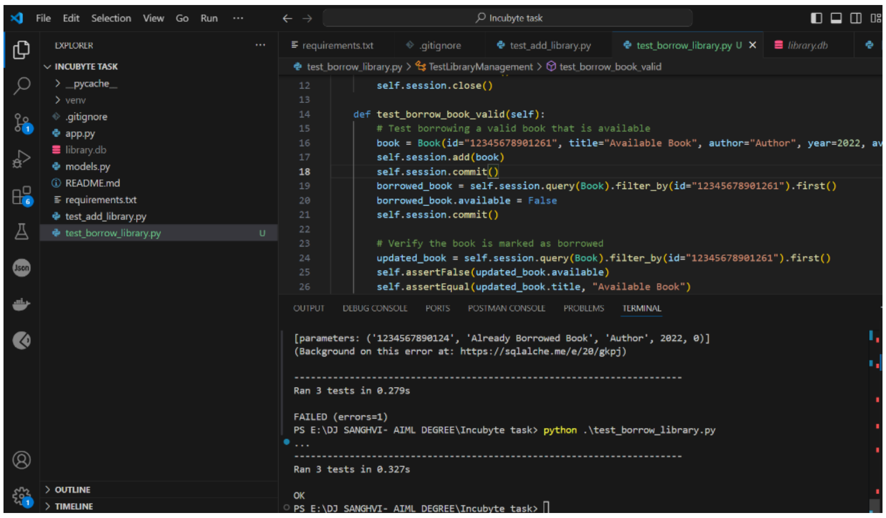
Publication Year: 2019

Available?: [✓ Yes] No

Book 'Python' added successfully!

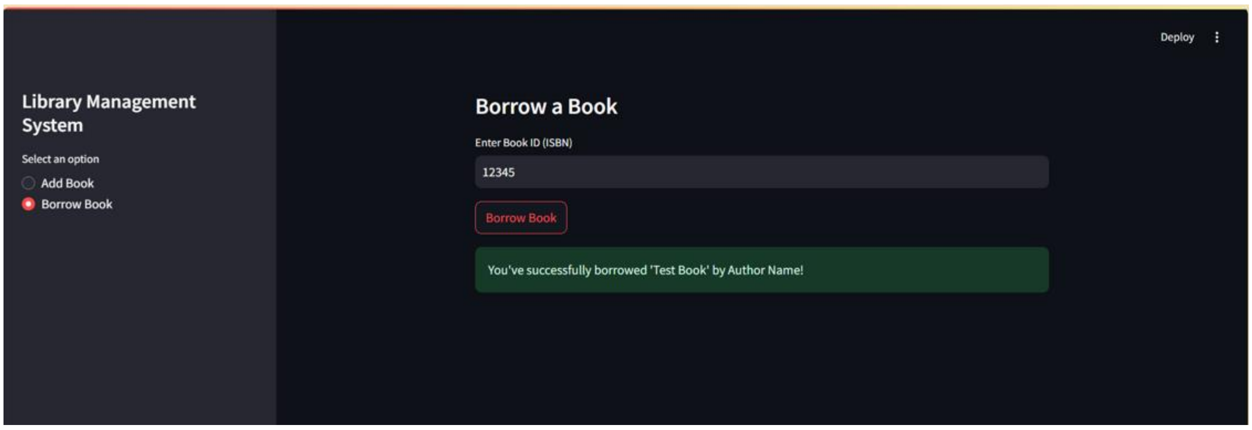
2) Borrowing Books in the Library System:

Test Case Id:	Description	Precondition	Steps	Expected Result	Status
1	Borrowing a Book from the library	A book should be there in the library	Select "Borrow Book," enter the ISBN number in the field, and click on borrow.	The book should be borrowed, and its availability should be updated to false.	PASS
2	Borrow a book which is not there in library	There should be no books in the library	Select "Borrow Book," enter the ISBN number, and attempt to borrow a book with an ISBN number that does not exist.	The system should display an error indicating that no book was found for the provided ISBN number.	PASS
3	Borrow a book which is already borrowed	The books in the library should be borrowed	Select "Borrow Book" and enter the ISBN number of a book that has already been borrowed.	The system should display an error stating that the book is not available.	PASS



```python
        self.session.close()

    def test_borrow_book_valid(self):
        # Test borrowing a valid book that is available
        book = Book(id="12345678901261", title="Available Book", author="Author", year=2022, av
        self.session.add(book)
        self.session.commit()
        borrowed_book = self.session.query(Book).filter_by(id="12345678901261").first()
        borrowed_book.available = False
        self.session.commit()

        # Verify the book is marked as borrowed
        updated_book = self.session.query(Book).filter_by(id="12345678901261").first()
        self.assertFalse(updated_book.available)
        self.assertEqual(updated_book.title, "Available Book")
```

```
[parameters: ('1234567890124', 'Already Borrowed Book', 'Author', 2022, 0)]
(Background on this error at: https://sqlalche.me/e/20/gkpj)

----------------------------------------------------------------------
Ran 3 tests in 0.279s

FAILED (errors=1)
PS E:\DJ SANGHVI- AIML DEGREE\Incubyte task> python .\test_borrow_library.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.327s

OK
PS E:\DJ SANGHVI- AIML DEGREE\Incubyte task> 
```



Borrow a Book

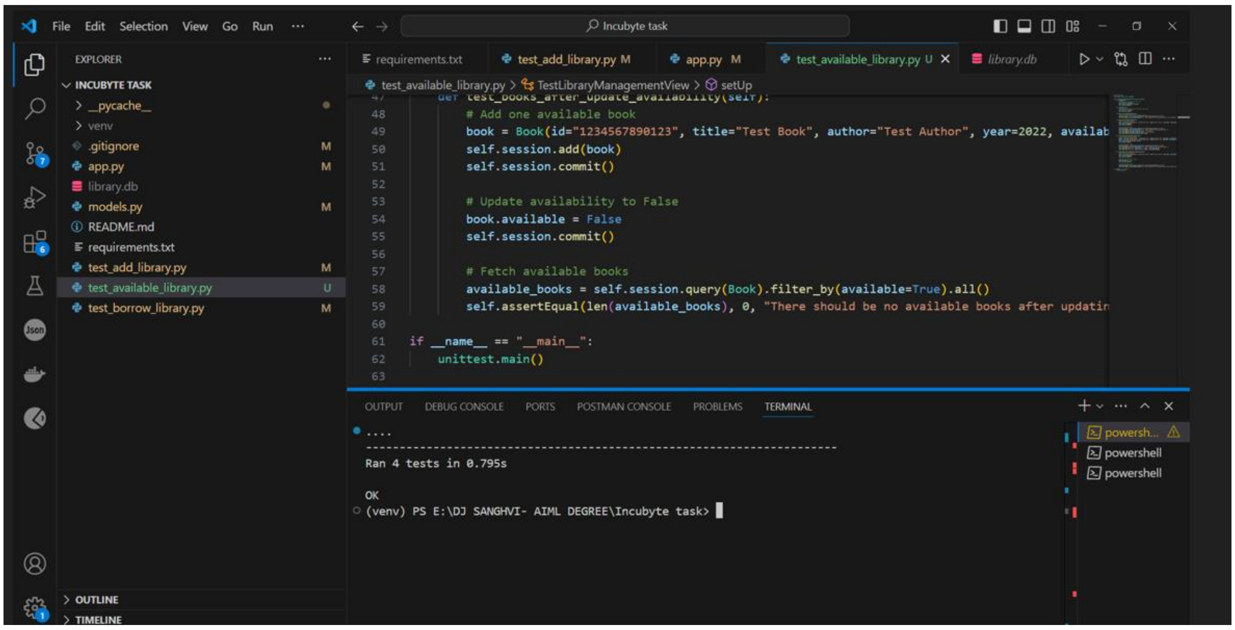
Enter Book ID (ISBN): 12345

Select an option: Add Book [✓ Borrow Book]

You've successfully borrowed 'Test Book' by Author Name!

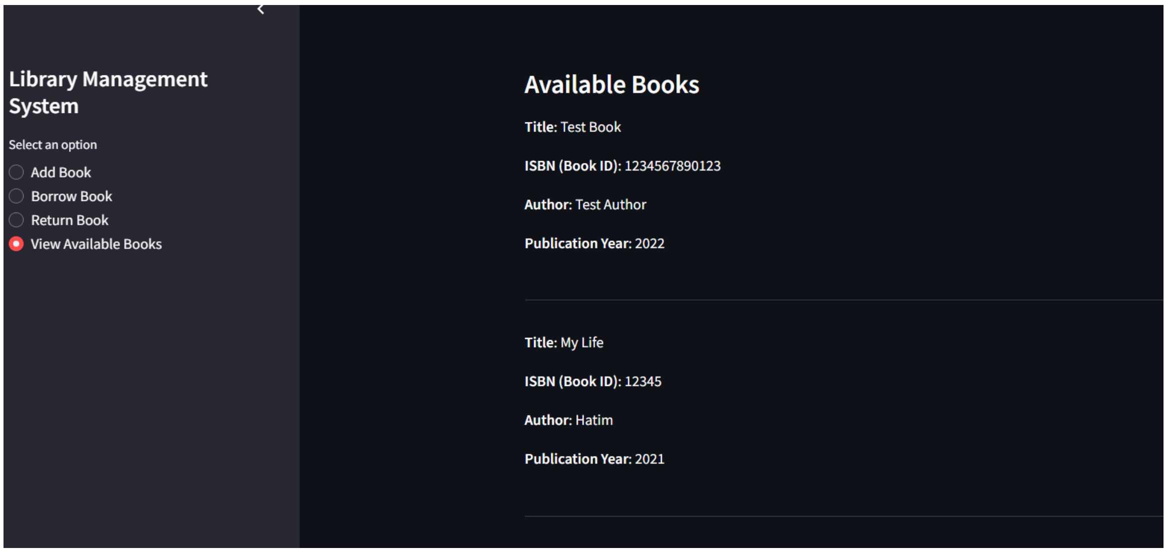
3) Displaying Available Books in the Library System:

Test Case Id:	Description	Precondition	Steps	Expected Result	Status
1	Fetching all the Available Books	There should be books in the library that are available for borrowing.	Select "Available books" from the sidebar.	All the books in the library that are not borrowed should be displayed.	PASS
2	Updating the book's availability and fetching the list of books.	There should be books in the library that are available for borrowing.	Select "Borrow Book," enter the ISBN number in the field, and click on borrow. Then, click on "Available Books."	The system should remove the borrowed book from the list of available books.	PASS



```python
    def test_books_after_update_availability(self):
        # Add one available book
        book = Book(id="1234567890123", title="Test Book", author="Test Author", year=2022, availab
        self.session.add(book)
        self.session.commit()

        # Update availability to False
        book.available = False
        self.session.commit()

        # Fetch available books
        available_books = self.session.query(Book).filter_by(available=True).all()
        self.assertEqual(len(available_books), 0, "There should be no available books after updatin

if __name__ == "__main__":
    unittest.main()
```

```
....
----------------------------------------------------------------------
Ran 4 tests in 0.795s

OK
(venv) PS E:\DJ SANGHVI- AIML DEGREE\Incubyte task>
```



Available Books

Title: Test Book

ISBN (Book ID): 1234567890123

Author: Test Author

Publication Year: 2022

Title: My Life

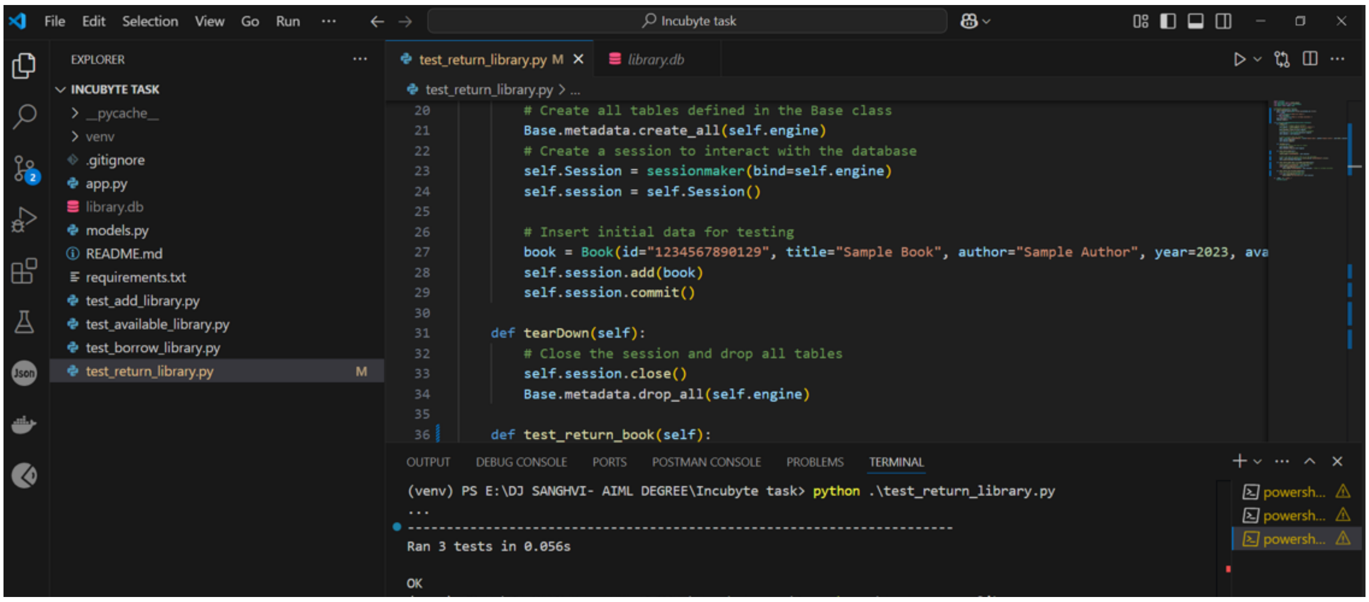
ISBN (Book ID): 12345

Author: Hatim

Publication Year: 2021

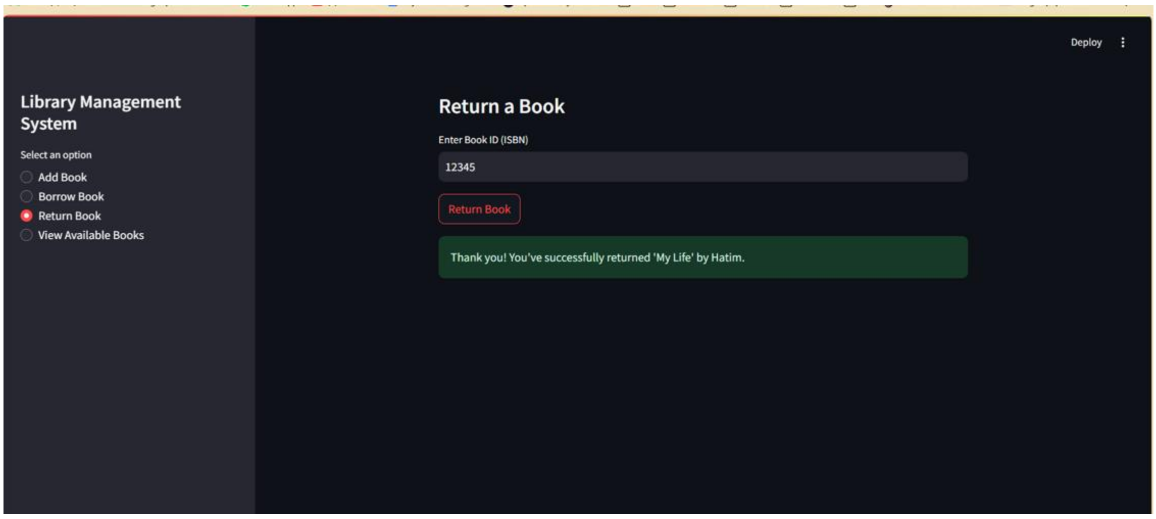
4) Returning Books in the Library System:

Test Case Id:	Description	Precondition	Steps	Expected Result	Status
1	Returning a book	A Book should be there in the library.	Select "Borrow Book," enter the ISBN number in the field, and click on borrow. Then, click on "Return," enter the ISBN number, and click the return button.	The returned book should now be marked as available.	PASS
2	Returning a book which is already available	A Book should be there in the library.	Select "Borrow Book," enter the ISBN number in the field, and click on the return button. The ISBN entered should belong to a book that is already available.	The system should display an error indicating that the book has already been returned.	PASS
3	Add a book with invalid Year	Add a new book with unique ISBN number.	Add a new book with an invalid year like 1400 or 2030.	The system should raise an error if the publication year is not between 1800 and 2024.	PASS



```python
        # Create all tables defined in the Base class
        Base.metadata.create_all(self.engine)
        # Create a session to interact with the database
        self.Session = sessionmaker(bind=self.engine)
        self.session = self.Session()

        # Insert initial data for testing
        book = Book(id="1234567890129", title="Sample Book", author="Sample Author", year=2023, ava
        self.session.add(book)
        self.session.commit()

    def tearDown(self):
        # Close the session and drop all tables
        self.session.close()
        Base.metadata.drop_all(self.engine)

    def test_return_book(self):
```

```
(venv) PS E:\DJ SANGHVI- AIML DEGREE\Incubyte task> python .\test_return_library.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.056s

OK
```



Select an option: Add Book [✓ Return Book] Borrow Book View Available Books

Enter Book ID (ISBN): 12345

Thank you! You've successfully returned 'My Life' by Hatim.