LIBRARY INVENTORY

SUBJECT: *PROGRAMMING FOR PROBLEM SOLVING USING PYTHON*

SUBMITTED TO: *DR. SAMEER FAROOQ*

COURSE CODE: *ETCCPP102*

MADE BY: *SONAM YADAV*

CLASS: *B.TECH CSE (AI ML)*

SECTION: *B*

ROLL NO: *2501730260*

INTRODUCTION

Libraries store a large number of books, and managing them manually often leads to confusion, mistakes, and loss of records. To solve this, digital library systems are used.
In this project, I have developed a **Library Inventory Management System** using Python. The main idea is to automate the basic operations like adding books, searching for books, issuing them, returning them, and maintaining their status.

The system uses **Object-Oriented Programming (OOP)** and **JSON file handling** for storing data permanently. Logging is also added to record each activity for safety. This project helped me understand how Python can be used to create real-world applications in a simple and efficient way.

OBJECTIVES

- ❖ To build a simple Python program that manages library books effectively.
- ❖ To store book details permanently using JSON.
- ❖ To implement functions for issuing and returning books safely.
- ❖ To allow users to easily search for books using title or ISBN
- ❖ To learn file handling, classes, objects, and logging in Python.

WORKING DESCRIPTION

The project **Library Inventory** is divided into 3 **Python** files.

1. **book.py**

   This file contains the **Book** class, which stores details of each books such as:
   - ✓ **Title**
   - ✓ **Author**
   - ✓ **ISBN**

- ✓ **Status** (available/ issued)

  It also includes methods for issuing, returning, and converting the book object into dictionary for saving in JSON.

2. **inventory.py**

   This file contains the **LibraryInventory** class, which manages a list of Book objects.
   Major functions include:
   - **add_book()** → Adds a new book while preventing duplicate ISBNs.
   - **issue_book()** → Issues a book if available.
   - **return_book()** → Marks a book as returned
   - **search_by_title()** → Searches based on title keywords
   - **search_by_isbn()** → Searches using ISBN
   - **save_catalog() / load_catalog()** → Saves and loads book data in books_catalog.json

This class acts as the “database” and main controller of the system.

3. **main.py**

   This is the user interface file.

   It shows a simple menu:

   1. Add Book
   2. Issue Book
   3. Return Book
   4. View All Books
   5. Search Book
   6. Exit

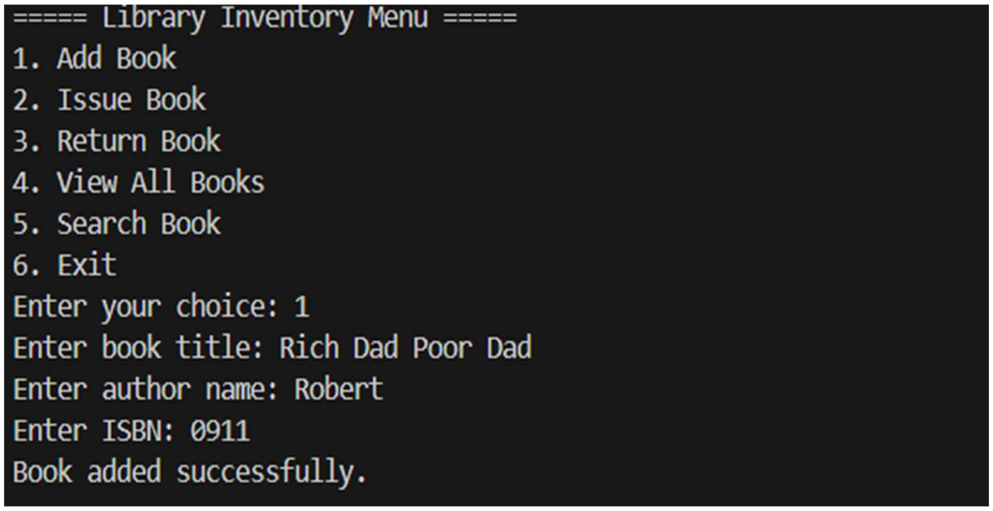
The user selects an option, enters the requested details, and the program performs the operations using the LibrayInventory class.

4. **JSON & Logging**

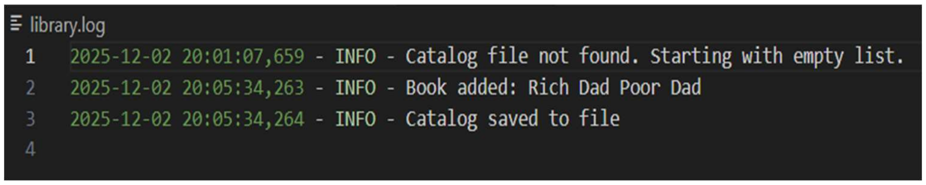
All books are saved in **books_catalog.json** so that data remains even after the program closes. Every important action (add, issue, return, error) is saved in **library.log** with timestamps.

OUTPUT SCREENSHOTS

1. **Adding a Book**

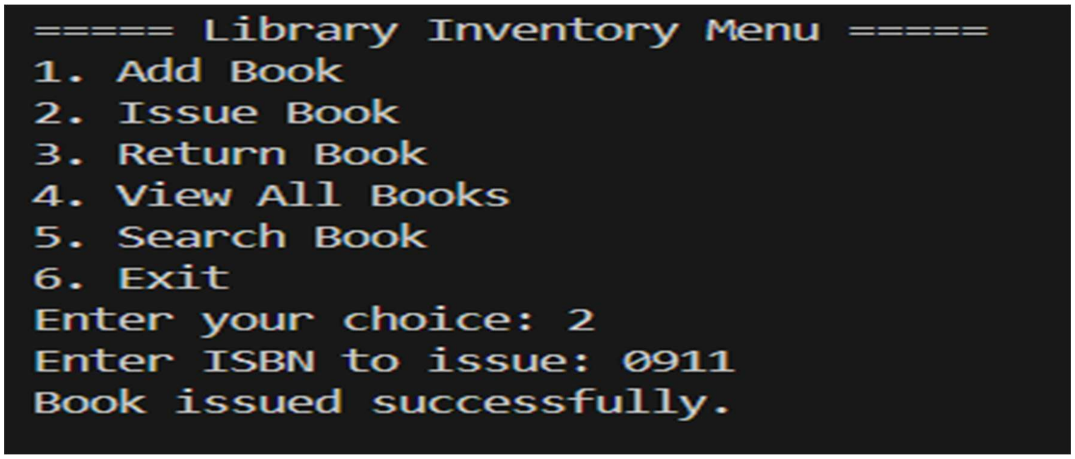


```
===== Library Inventory Menu =====
1. Add Book
2. Issue Book
3. Return Book
4. View All Books
5. Search Book
6. Exit
Enter your choice: 1
Enter book title: Rich Dad Poor Dad
Enter author name: Robert
Enter ISBN: 0911
Book added successfully.
```

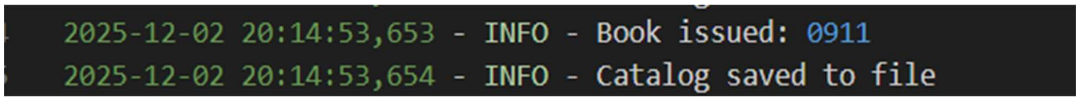


```
library.log
2025-12-02 20:01:07,659 - INFO - Catalog file not found. Starting with empty list.
2025-12-02 20:05:34,263 - INFO - Book added: Rich Dad Poor Dad
2025-12-02 20:05:34,264 - INFO - Catalog saved to file
```

2. **Issue Book**

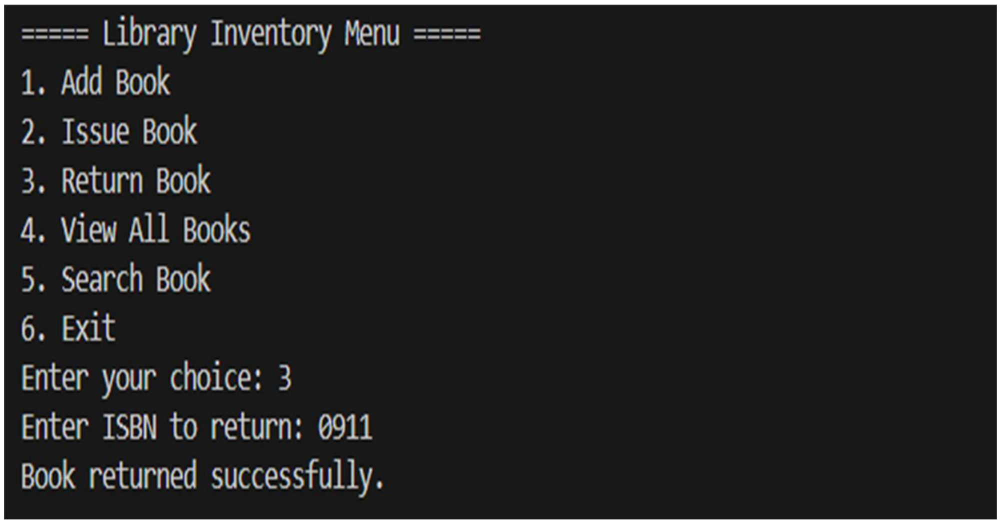


```
===== Library Inventory Menu =====
1. Add Book
2. Issue Book
3. Return Book
4. View All Books
5. Search Book
6. Exit
Enter your choice: 2
Enter ISBN to issue: 0911
Book issued successfully.
```

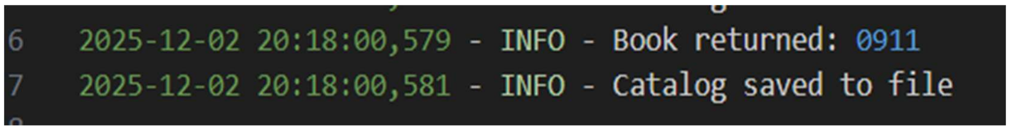


```
2025-12-02 20:14:53,653 - INFO - Book issued: 0911
2025-12-02 20:14:53,654 - INFO - Catalog saved to file
```

3. **Return Book**

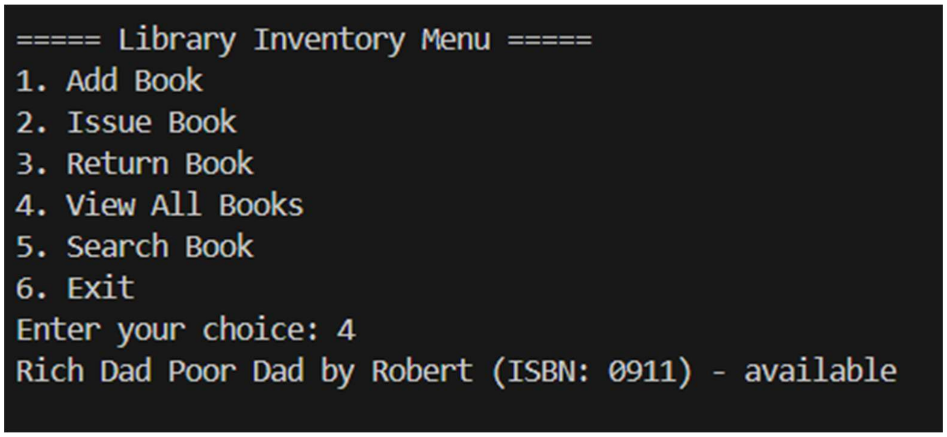


```
===== Library Inventory Menu =====
1. Add Book
2. Issue Book
3. Return Book
4. View All Books
5. Search Book
6. Exit
Enter your choice: 3
Enter ISBN to return: 0911
Book returned successfully.
```



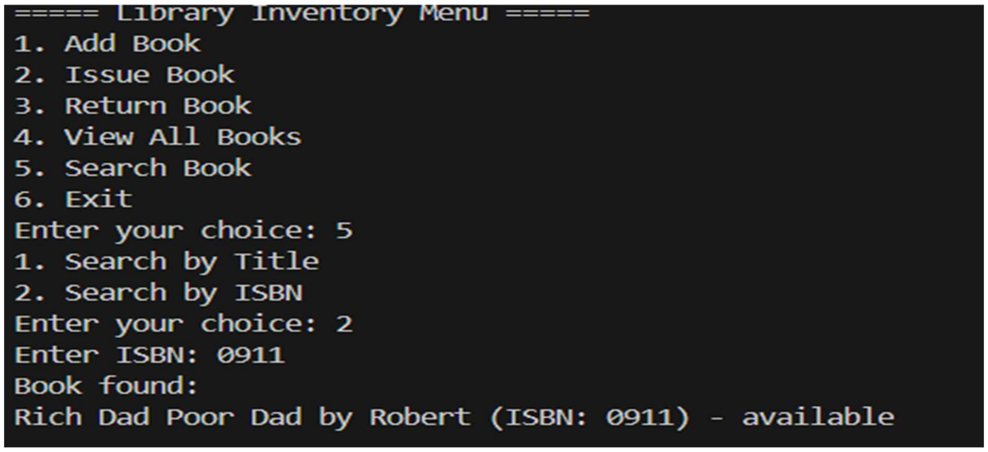
```
2025-12-02 20:18:00,579 - INFO - Book returned: 0911
2025-12-02 20:18:00,581 - INFO - Catalog saved to file
```

4. **View All Books**



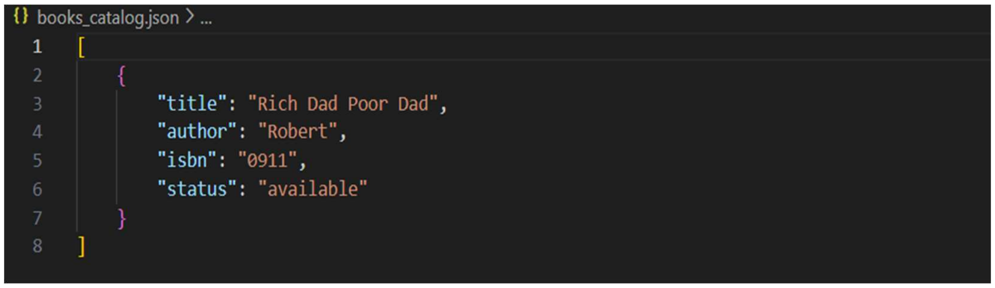
```
===== Library Inventory Menu =====
1. Add Book
2. Issue Book
3. Return Book
4. View All Books
5. Search Book
6. Exit
Enter your choice: 4
Rich Dad Poor Dad by Robert (ISBN: 0911) - available
```

5. Search Book



```
===== Library Inventory Menu =====
1. Add Book
2. Issue Book
3. Return Book
4. View All Books
5. Search Book
6. Exit
Enter your choice: 5
1. Search by Title
2. Search by ISBN
Enter your choice: 2
Enter ISBN: 0911
Book found:
Rich Dad Poor Dad by Robert (ISBN: 0911) - available
```

6. JSON FILE STORAGE



```json
[
    {
        "title": "Rich Dad Poor Dad",
        "author": "Robert",
        "isbn": "0911",
        "status": "available"
    }
]
```

REFLECTION

Working on this Python mini-project helped me understand how modules, packages, and file structures work together in real programs. While building the library manager, I initially faced issues with imports, especially when the script could not recognize modules from another directory. Fixing this taught me the importance of running Python from the correct project root and keeping a clean folder structure.

I also improved my understanding of classes, object-oriented design, and how individual components such as Book, Inventory, and CLI scripts interact with each other. Overall, this project strengthened my confidence in organizing Python code and solving small debugging problems that appear in real development workflows.

CONCLUSION

This project gave me hands-on experience in building a real-world Python application. I learned how to use classes and objects, how to store data using JSON, and how to maintain logs for system activities.

The Library Inventory Management System works smoothly for adding, issuing, returning, and searching books. It also ensures that data is saved safely and can be retrieved anytime.

Overall, this project helped me understand the importance of structured coding and data management, and it gave me confidence in applying Python to practical problems.

