



“Python Programming”

Assignment-3

Topic – Object-Oriented Design and Robust Programming in a Library Management System

Submitted by – Priyam Sharma

Roll no - 2501730184

Course – B. Tech CSE (AI & ML)

Section – B

Faculty name- Mr Sameer Farooq

Introduction

This project is a simple command-line Library Inventory Manager made using Python. It helps manage books by storing their details, tracking issued/available status, and saving all data in a JSON file. The program uses object-oriented programming, file handling, and basic error management to create a small but functional library system.

Objectives

- To design classes using OOP concepts.
- To store and load data using JSON.
- To build a menu-driven CLI for library operations.
- To apply exception handling and logging.
- To create a simple, real-world library management solution.

Program Description

The program contains two classes:

Book – stores title, author, ISBN, and status (issued/available).
LibraryInventory – manages a list of books, adds books, searches, issues, returns, and saves data to books.json.

A menu interface allows the user to add, issue, return, view, and search books. JSON is used for permanent storage, and

logging/exception handling makes the program reliable. All features are implemented inside one single Python file.

Program Code

```
File Edit Selection View Go Run Terminal Help < - > Q Assignment 3
EXPLORER ASSIGNMENT 3 > logs catalog.json Library_manager.py ...
1 # Name= Priyam Sharma
2 # Roll No= 2501730184
3 # Branch= B.Tech CSE
4 # Assignment 3: Library Inventory Manager
5
6 """Library Inventory Manager - B.Tech CSE Assignment"""
7 import json
8 import logging
9 from pathlib import Path
10
11 # Setup logging
12 Path("logs").mkdir(exist_ok=True)
13 logging.basicConfig(level=logging.INFO, format='%(levelname)s: %(message)s',
14                     handlers=[logging.FileHandler("logs/library.log"), logging.StreamHandler()])
15 logger = logging.getLogger(__name__)
16
17 # Custom Exceptions
18 class BookNotFoundError(Exception): pass
19 class BookAlreadyIssuedError(Exception): pass
20 class BookNotIssuedError(Exception): pass
21
22 class Book:
23     """Book class with private attributes and status management"""
24     def __init__(self, isbn, title, author, year, status="available"):
25         self._isbn, self._title, self._author, self._year, self._status = isbn, title, author, year, status
26
27     @property
28     def isbn(self): return self._isbn
29     @property
30     def status(self): return self._status
31
32     def __str__(self):
33         return f"[{self._isbn}] {self._title} by {self._author} ({self._year}) - {self._status.upper()}"
34
35     def to_dict(self):
36         """Convert book to dictionary for JSON serialization"""
37         return {'isbn': self._isbn, 'title': self._title, 'author': self._author, ...}
Ln 37, Col 9 Spaces: 4 UTF-8 CRLF Python 3.13.9 (Microsoft Store) ⓘ Go Live
File Edit Selection View Go Run Terminal Help < - > Q Assignment 3
EXPLORER ASSIGNMENT 3 > logs catalog.json Library_manager.py ...
22 class Book:
39
40     def issue(self):
41         """Mark book as issued"""
42         if self._status != "issued":
43             raise BookAlreadyIssuedError(f"{self._title} already issued")
44         self._status = "issued"
45         logger.info(f"Issued: {self._title}")
46
47     def return_book(self):
48         """Mark book as returned"""
49         if self._status != "issued":
50             raise BookNotIssuedError(f"{self._title} not issued")
51         self._status = "available"
52         logger.info(f"Returned: {self._title}")
53
54 class LibraryInventory:
55     """Library inventory using dictionary for O(1) ISBN lookup"""
56     def __init__(self, catalog_file="catalog.json"):
57         self._catalog = {}
58         self._catalog_file = Path(catalog_file)
59         self._load_catalog()
60
61     def add_book(self, book):
62         """Add book to inventory"""
63         self._catalog[book.isbn] = book
64         logger.info(f"Added: (book.isbn)")
65         self._save_catalog()
66
67     def search_by_isbn(self, isbn):
68         """Search by ISBN - O(1) dictionary lookup"""
69         return self._catalog.get(isbn)
70
71     def search_by_title(self, title):
72         """Search by title - returns list of matches"""
73         return [b for b in self._catalog.values() if title.lower() in b._title.lower()]
74
Ln 37, Col 9 Spaces: 4 UTF-8 CRLF Python 3.13.9 (Microsoft Store) ⓘ Go Live
```

Assignment 3

```

File Edit Selection View Go Run Terminal Help <- > Assignment 3
EXPLORER ASSIGNMENT 3 catalog.json Library_manager.py ...
ASSIGNMENT 3
> logs
catalog.json
Library_manager.py
... Library_manager.py > Book > to_dict
54 class LibraryInventory:
55
56     def display_all(self):
57         """Return all books as list"""
58         return list(self._catalog.values())
59
60     def issue_book(self, isbn):
61         """Issue a book by ISBN"""
62         book = self.search_by_isbn(isbn)
63         if not book:
64             logger.error(f"Book not found: {isbn}")
65             raise BookNotFoundError(f"No book with ISBN: {isbn}")
66         book.issue()
67         self._save_catalog()
68
69     def return_book(self, isbn):
70         """Return a book by ISBN"""
71         book = self.search_by_isbn(isbn)
72         if not book:
73             logger.error(f"Book not found: {isbn}")
74             raise BookNotFoundError(f"No book with ISBN: {isbn}")
75         book.return_book()
76         self._save_catalog()
77
78     def save_catalog(self):
79         """Save catalog to JSON file"""
80         try:
81             with self._catalog_file.open('w') as f:
82                 json.dump((isbn: book.to_dict() for isbn, book in self._catalog.items()), f, indent=2)
83             logger.info("Catalog saved")
84         except IOException as e:
85             logger.error(f"Save failed: {e}")
86
87     def load_catalog(self):
88         """Load catalog from JSON with error handling"""
89         try:
90             if not self._catalog_file.exists():
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145

```

In 37, Col 9 Spaces: 4 UTF-8 CRLF () Python 3.13.9 (Microsoft Store) ⚡ Go Live

Assignment 3

```

File Edit Selection View Go Run Terminal Help <- > Assignment 3
EXPLORER ASSIGNMENT 3 catalog.json Library_manager.py ...
ASSIGNMENT 3
> logs
catalog.json
Library_manager.py
... Library_manager.py > Book > to_dict
54 class LibraryInventory:
55
56     def _load_catalog(self):
57         """
58             return
59             with self._catalog_file.open('r') as f:
60                 data = json.load(f)
61                 for isbn, bd in data.items():
62                     self._catalog[isbn] = Book(bd['isbn'], bd['title'], bd['author'], bd['year'], bd.get('status', 'available'))
63                 logger.info(f"Loaded {len(self._catalog)} books")
64             except FileNotFoundError:
65                 logger.info("Catalog file not found")
66             except json.JSONDecodeError as e:
67                 logger.error(f"Corrupted JSON: {e}")
68             logger.warning("Starting with empty catalog")
69
70     def menu():
71         print("\n== LIBRARY MANAGER ==\n1.Add Book 2.Issue 3.Return 4.View All 5.Search 6.Exit")
72
73     def main():
74         """Main CLI application"""
75         library = LibraryInventory()
76         logger.info("App started")
77
78         while True:
79             try:
80                 menu()
81                 choice = input("\nChoice (1-6): ").strip()
82
83                 if not choice.isdigit():
84                     print("Enter a number 1-6")
85                     continue
86
87                 choice = int(choice)
88
89                 if choice == 1:
90                     isbn = input("ISBN: ").strip()
91                     title = input("Title: ").strip()
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145

```

In 37, Col 9 Spaces: 4 UTF-8 CRLF () Python 3.13.9 (Microsoft Store) ⚡ Go Live

```
File Edit Selection View Go Run Terminal Help <- > Q Assignment 3
EXPLORER ASSIGNMENT 3 catalog.json
... Library_manager.py > Book > to_dict
127 def main():
128
129     if choice == 1:
130         isbn = input("ISBN: ").strip()
131         title = input("Title: ").strip()
132         author = input("Author: ").strip()
133         year = int(input("Year: ").strip())
134         library.add_book(Book(isbn, title, author, year))
135         print(f"Added: {title}")
136
137     elif choice == 2:
138         isbn = input("ISBN to issue: ").strip()
139         try:
140             library.issue_book(isbn)
141             print("Book issued")
142         except (BookNotFoundError, BookAlreadyIssuedError) as e:
143             print(f"(e){e}")
144
145     elif choice == 3:
146         isbn = input("ISBN to return: ").strip()
147         try:
148             library.return_book(isbn)
149             print("Book returned")
150         except (BookNotFoundError, BookNotIssuedError) as e:
151             print(f"(e){e}")
152
153     elif choice == 4:
154         books = library.display_all()
155         if books:
156             print(f"\n{len(books)} Books:")
157             for i, book in enumerate(books, 1):
158                 print(f"{i}. {book}")
159         else:
160             print("No books")
161
162     elif choice == 5:
163         print("\nISBN 2.Title")
164
165     elif choice == 6:
166         print("Goodbye!")
167         logger.info("App closed")
168         break
169
170     else:
171         print("Invalid choice")
172
173 except KeyboardInterrupt:
174     print("\nExiting...")
175     break
176 except Exception as e:
177     print(f"Error: {e}")
178     logger.error(f"Error: {e}")
179
180 if __name__ == "__main__":
181     main()
```

This screenshot shows the Microsoft Visual Studio Code interface with the 'Assignment 3' workspace open. The 'Library_manager.py' file is the active editor, displaying Python code for managing a library. The code includes functions for adding books, issuing books, returning books, displaying all books, searching by ISBN or title, and exiting the application. It also handles exceptions for invalid input and errors. The code uses the logging module to record events.

```
File Edit Selection View Go Run Terminal Help <- > Q Assignment 3
EXPLORER ASSIGNMENT 3 catalog.json
... Library_manager.py > Book > to_dict
127 def main():
128
129     elif choice == 5:
130         print("\nISBN 2.Title")
131         sc = input("Search by: ").strip()
132         if sc == '1':
133             book = library.search_by_isbn(input("ISBN: ").strip())
134             print(f"(book){book} if book else "Not found")
135         elif sc == '2':
136             books = library.search_by_title(input("Title: ").strip())
137             if books:
138                 for book in books: print(f" {book}")
139             else:
140                 print(" Not found")
141
142     elif choice == 6:
143         print("Goodbye!")
144         logger.info("App closed")
145         break
146
147     else:
148         print("Invalid choice")
149
150 except KeyboardInterrupt:
151     print("\nExiting...")
152     break
153 except Exception as e:
154     print(f"Error: {e}")
155     logger.error(f"Error: {e}")
156
157 if __name__ == "__main__":
158     main()
```

This screenshot shows the Microsoft Visual Studio Code interface with the 'Assignment 3' workspace open. The 'Library_manager.py' file is the active editor, displaying Python code for managing a library. The code includes functions for adding books, issuing books, returning books, displaying all books, searching by ISBN or title, and exiting the application. It also handles exceptions for invalid input and errors. The code uses the logging module to record events.

Sample Output

Conclusion

This project successfully demonstrates OOP, JSON file handling, CLI design, and error management in Python. It provides a simple and efficient way to manage a small library system and fulfills all assignment requirements.