Project Report Submission

Introduction

Course – VITYARTHI Python Course

Programming Language – Python

Topic – Library Management System

Name – Aryaman Saxena

Registration Number – 25BCE10482

Faculty Member – Monika Vyas

Problem Statement:-

In small or old libraries, it is not easy to manage or track the entire book inventory. As, most of the times it is done manually it Is not easy to maintain a complete accurate record and it is very common to do small mistakes. Because of human error, wrong quantities can be entered in the data or the process of transaction, checking due dates might not be that good at all. Also, it is difficult to maintain accountability or even report an error.

In order to provide a solution to this problem, I have created a Library Management System where all of this can be done automatically. This increases accuracy and efficiency and also saves our time. With the help of it, we can easily check the list of the books, which book was borrowed when was it borrowed and all other things quite easily.

Objectives: -

The main objectives of the project are: -

- Implement CRUD Operations: The project should be able to create, record, update and delete records for both books and user.
- Data Persistence: A JSON file should be created where the data will be saved preventing data loss.
- Automatic Processes: Users should be able to borrow, return and get other information automatically without any need of a person.
- Tracking Deadlines: It should be able to track the deadline.

- The project should also be able to list the available books with the required details. Also, brief overview of other crucial and extra things should be shown.

Functional Requirements: -

The required functional requirements have been met: -

- Three major modules: - I have used os module along with json module and datetime module as well
- Clear Input / Output Structures: -
  In my programs very easily a user can enter the input and will also get the desired output.
  For Example – You have several options, a total of 10 to 11 from where you can select and random one and get the desired output.
- Other things such as Book Management where one can view all the books or check their availability or even trap top 5 books as well. For user, he can add another user, see the registered users list already, borrow books as well as return a book and also check the view borrowed list.
- Along with that an Overdue Report which generates a report of the list of all transactions is available. The data is saved properly showing data persistence and along with that reminders are also displayed where the system reminds the user as to return the book.
- Logical Workflow Overview: -
  In my project, once the commands are run the Library object loads the already present data via library_data.json file. Based on that, the user is presented a menu with 10 options giving many features. User can select any one of them like list all

books, etc, if any input is required then it will be asked and then the later methods will be applied and based on the logic the output will be displayed. After this, as everything is in loop again the options will be presented until option 10 is selected which is “Save and Exit Operation”.

Non Functional Requirements: -

- Reliability or Data Persistence: - In my project, the Library class in library.py uses the saved() method to write the entire application state to the library_data.json file. The load_data() method ensures that the application state is successfully restored when the program starts.
- Usability: - The system is easy to use. As I have already explained the workflow of my program, the program gives 10 options and user can do anything they want. Basic input validation and error handling has been taken care of. In cases of errors, proper messages such as “Book Not available” can be seen. The output from reports also provides clear and immediate response.
- Maintainability: - The project is made using OOPS which keeps the code quite maintained and segregated. Also, different files are made and kept to keep things separate and clean.
- Portability: - This project can run on several operating systems like (Windows, macOS, Linux) without significant change in the code as it has been built in Python 3 and normal or simple built in libraries have been used which are datetime, json and os.

System Architecture: -

1.) Frond End: -

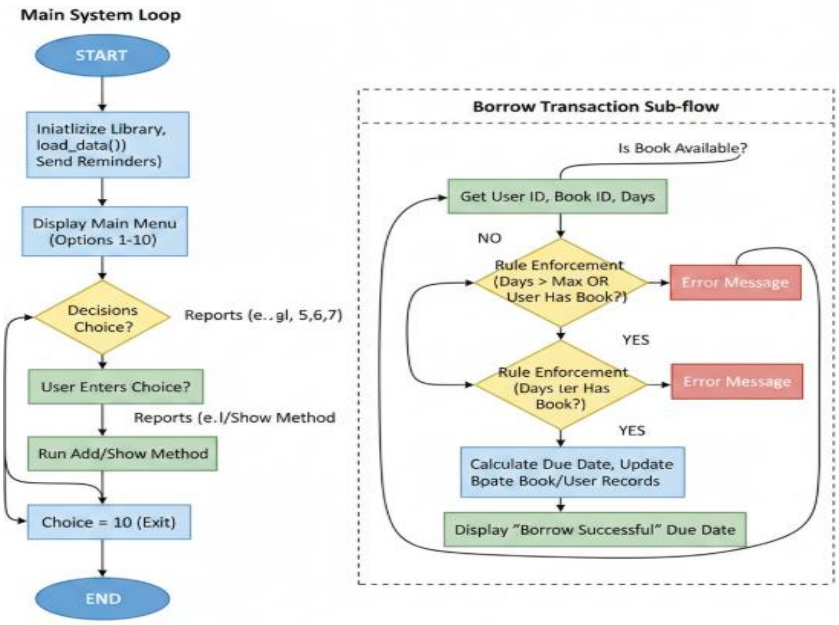
The main role Is to check all inputs and outputs.It will present the main menu, and will take user input, verify that and display the message based on the logic written earlier in the code.

2.)Business Logic/Application Layer: -

This is the most important and crucial part of the code. All the classes such as Library, User, Book, Reports, Notifications are included. The main class which is Library is main and also acts as Controller and manages the book and user object. These two classes have the main data and logic. Based on which, reports and notifications classes do extra work.

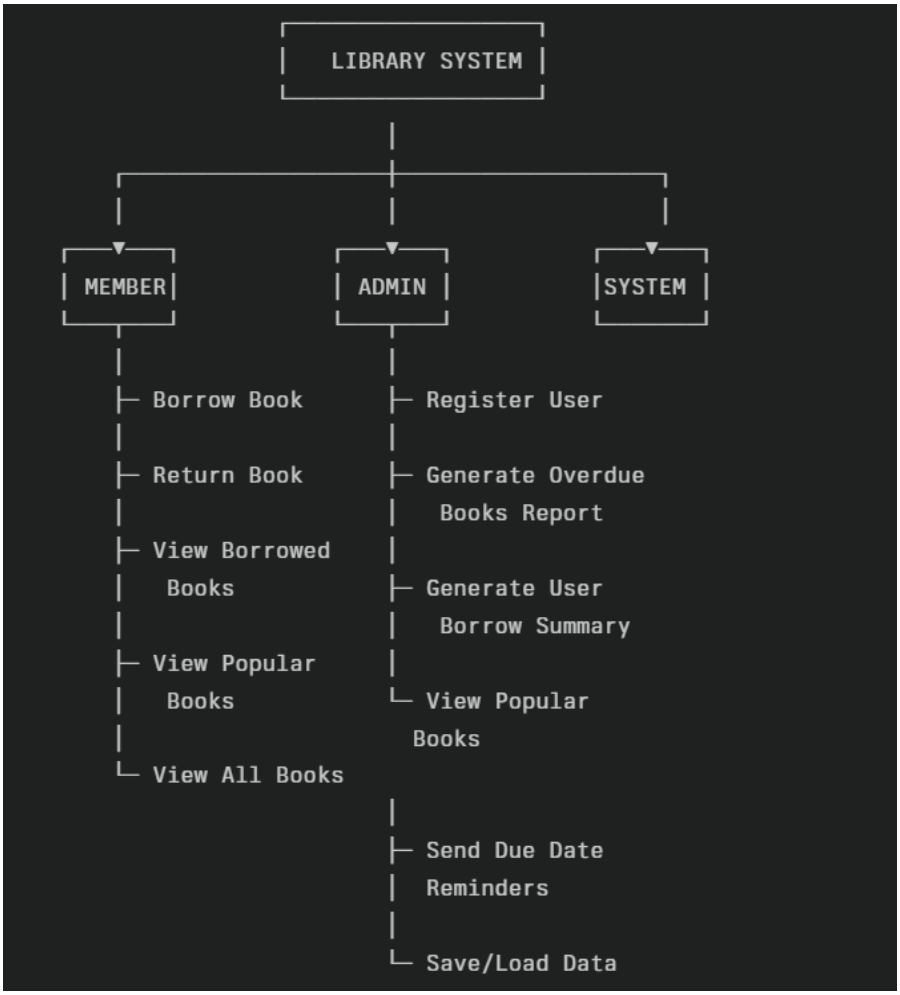
3.)Data Persistence Layer (Back – End): -

The library_data.json file is used here. Ensures reliability by converting Python objects into JSON style and also saves it.

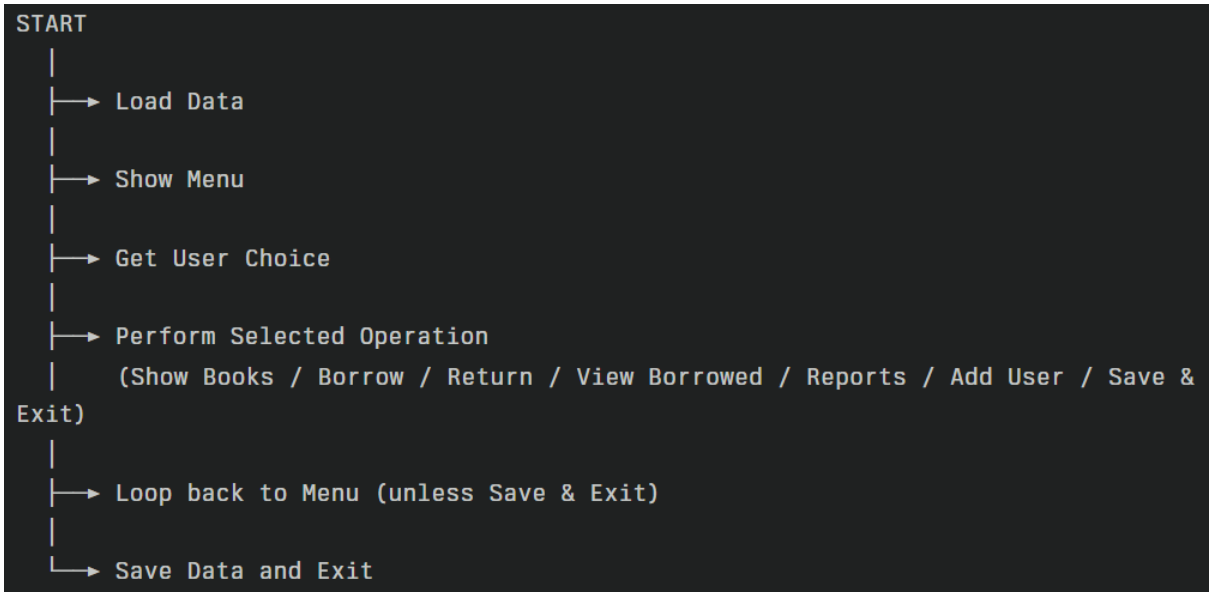


Design Diagrams: -

1.) Use Case Diagram: -

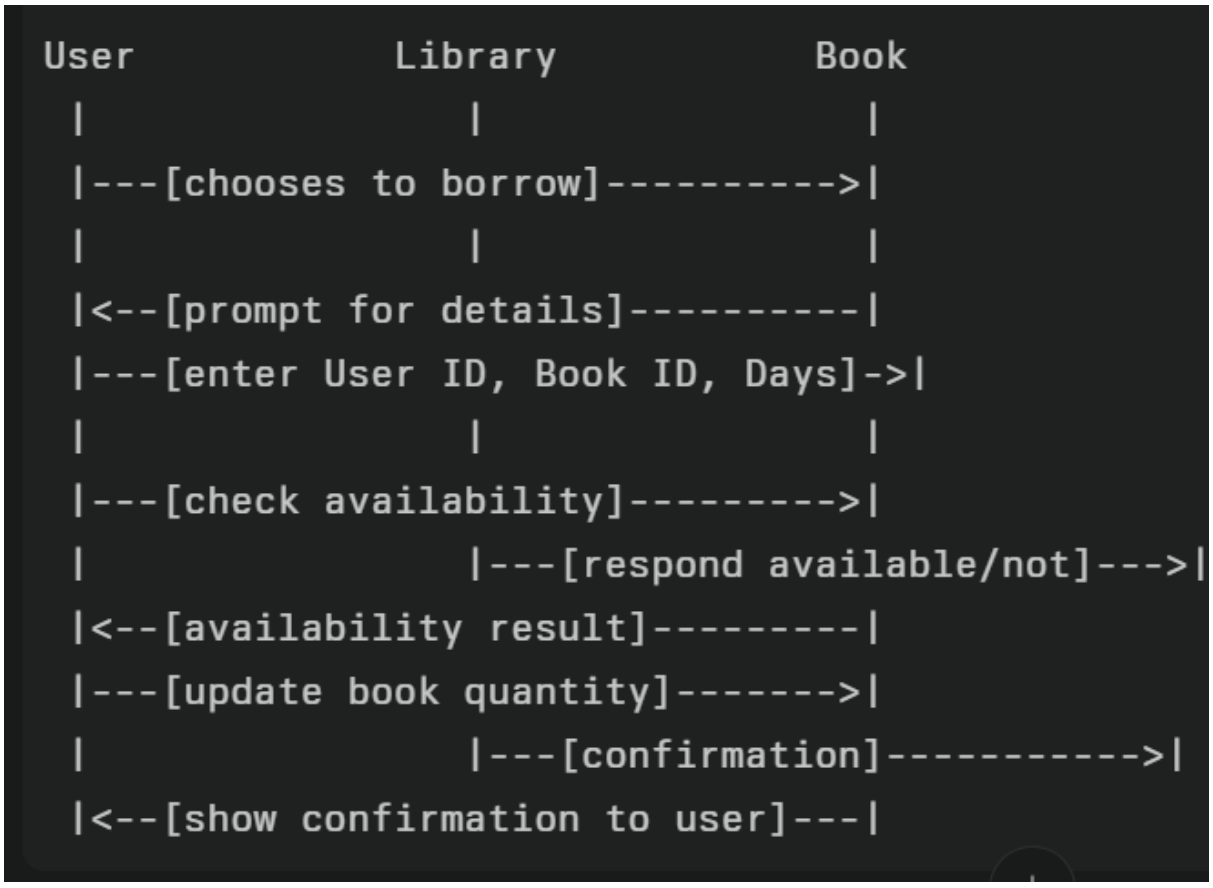


2.) Workflow Diagram: -

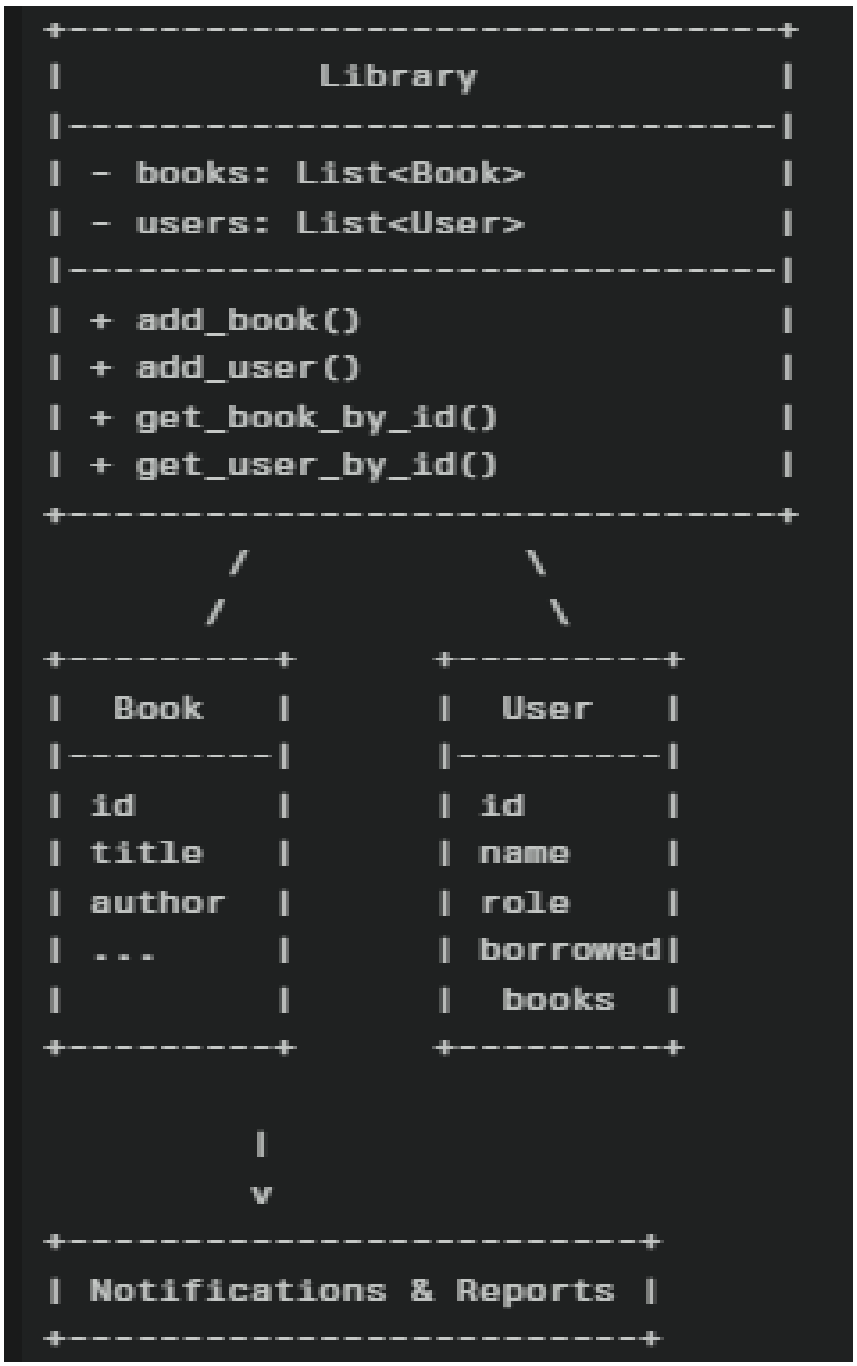


3.)Sequence Diagram: -

This is a flowchart for sequence for borrowing a book.



4.) Class/Component Diagram: -



Design Decisions and Rationale: -

- Decision – 1: Object – Oriented Architecture

  Using OOPS is better to separate our code properly. Each block or code manages its own data. We can reuse instead of copying the code, thereby making the code less lengthy and simple. Also, it is easy to add new features hence making our program very scalable.

- Decision – 2: JSON for Data Persistence
  Our program stores data in JSON file instead of a relational database. We do not need any complex database. It also allows manual verification and you can make changes very easily. Also, it can be shared and is quite portable.

- Decision – 3: Giving Options to the Users Like a Menu
  In our program, Interface is not GUI instead line wise. However, all the necessary things are performed by our program. It displays all the concepts properly and can work on any system.

- Decision – 4: Static Methods for Services:
  Main advantages are that we do not need instance data and it can be called without creation of object. Each call is independent and also similar to common python patters.

- Decision – 5: Simple Error Handling:
  Prevents crashes from invalid input. Also, helps in making sure that appropriate responses and feedback are given to the user and the system does not crash if something different is entered.

Implementation Details

1.)Core Classes and Responsibilities: -

Book Class – It is used to manage all details about a single book which includes its title, name and other things. Methods also update the book's status and print the information

User Class – Manages the data of the users and other things like id and their role.

Library Class - Orchestrates the entire operation, maintain collection of books and users. Also, provides several functions such as add, search, display and save or even load the data.

Notifications – Checks the borrowed books and print reminder to return the books.

Reports – Used to generate overdue books reports and summarize other counts of borrowing of the books.

2.) Data Handling: -

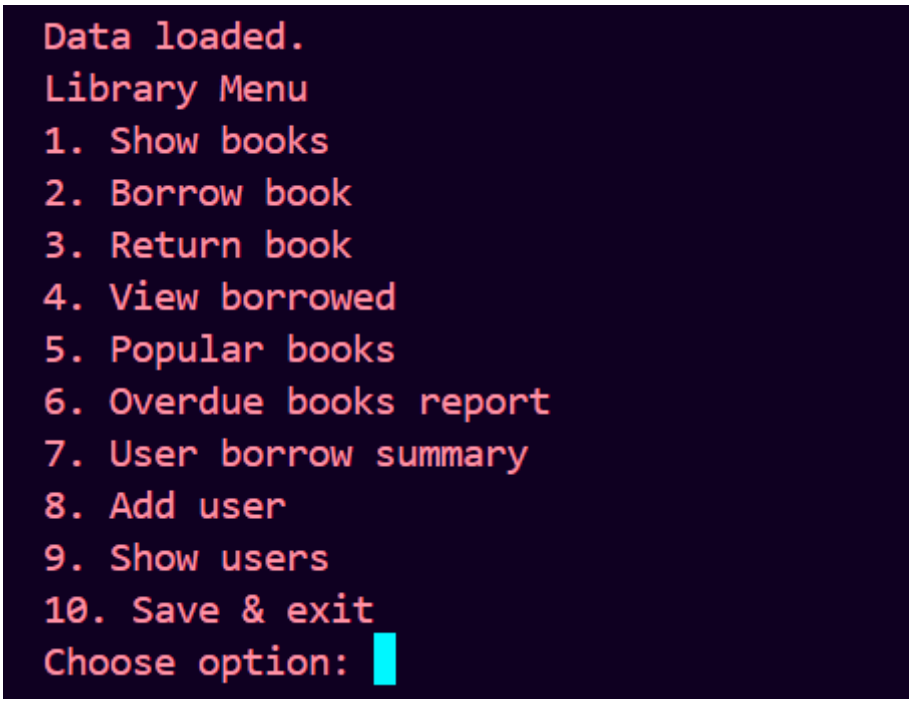
All permanent records are stored in a JSON File. Borrowed book and their data is also stored so that when the next reload happens that data is saved and can be used for other options. JSON file makes sure that this is possible.

3.)Workflow: -

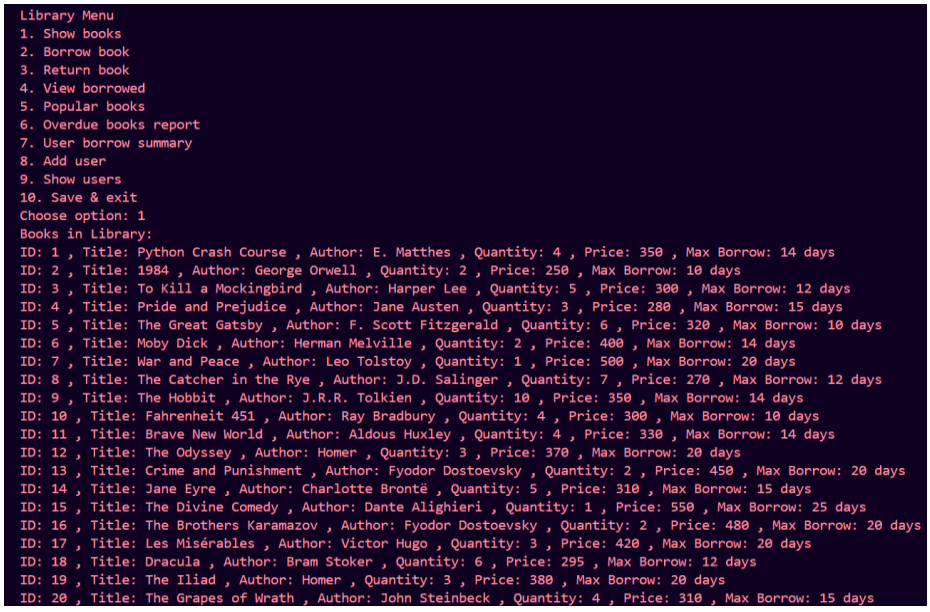
The user interacts with the system with the help of a menu system comprising of a total of 10 options. Admins can also add books while members can borrow or even return them.
The system also check invalid inputs and also gives proper responses

Screenshots and Results: -

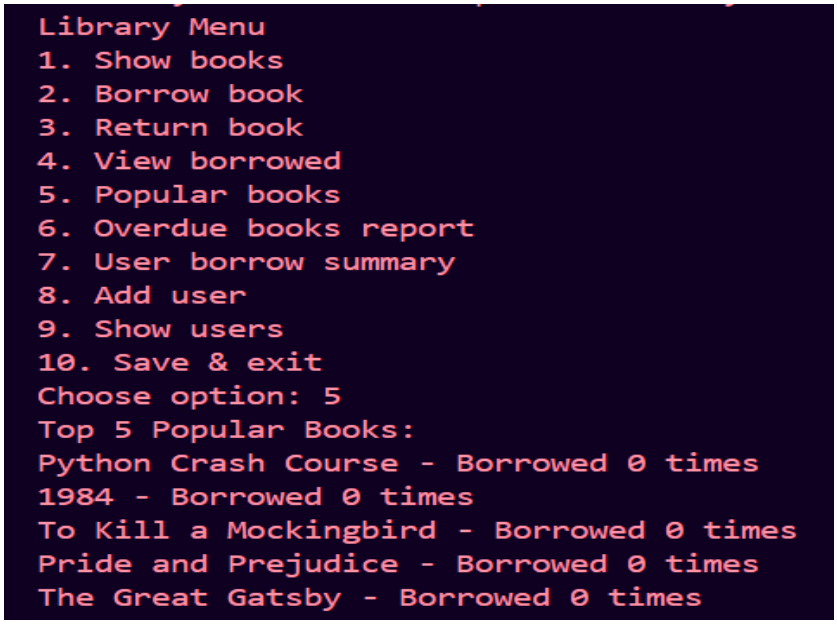
1.)Output where entire menu is available: -



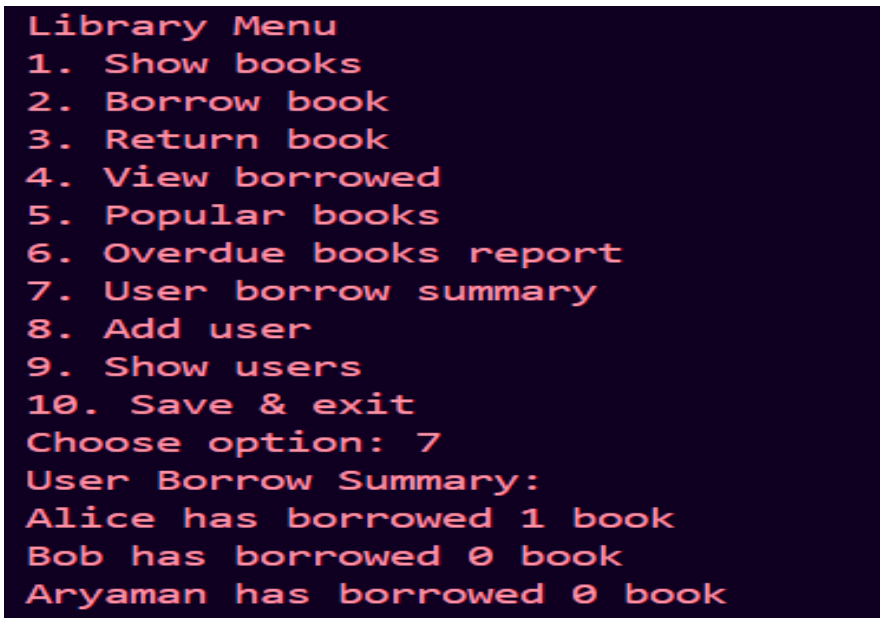
2.)Output being displayed of the list of books after the user selects option 2: -



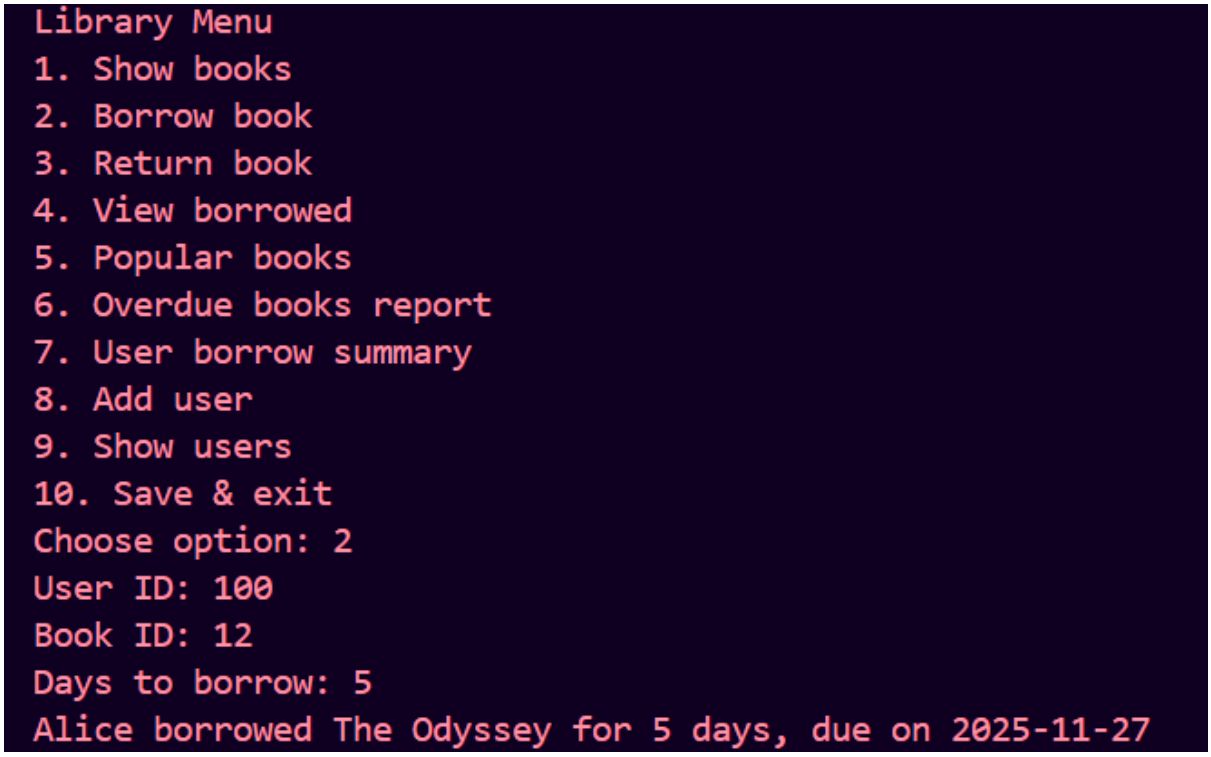
3.)Output being displayed when a user selects option 5



4.)Output being displayed after user decides to check the past record



5.) Output being displayed after user selects to borrow a book with the required inputs



Testing Approach: -

1.) Unit Testing: -

Each class and individual method are tested alone in this. Main focus is on the logic present within these methods and not considering other parts. Such as,

a. Verifying book borrowing, returning updates or even status.
b. Checking role of the user and what permissions the get
c. Ensuring that calculation of dates, or how much time is left are accurate.

d. Also, testing static functions to get correct output for setting sample data. These help in catching the error early

2.) Integration Testing: -

It checks how different modules affect each other: -

a. Testing book borrowing process and also considering the user's borrowed data.
b. Verifying reminders and notifications only come after appropriate time.
c. To check menu driven logic, such as adding a book, or registering a user and other methods.
d. Ensuring the correct reading and writing of JSON data is happening.

3.) System Testing: -

It focuses on validating the complete system: -

a. To simulate real world operations like adding books or users and returning or letting them borrow the books.
b. Testing extreme cases such as, to borrow a book which is not available or has been borrowed till its limit.
c. To ensure reports handle the latest data properly as well
d. To check the menu logic for errors and invalid input given by user.

4.) User Acceptance Test: -

It verifies that the system solves genuine needs of the user

a. Involves user to run the different scenarios.
b. Collect responses about different things like error messages, accuracy from the user.

c. Making adjustments based on their feedback and upgrading our project

The UAT test is very important as it ensures that our program is practical and useful to the users.

Challenges Faced: -

While making this project several challenges or tasks were considered by me, some of the important ones are: -

1.) Data Persistence: -

Storing and Retrieving data is a very difficult task considering I was using JSON file and I had to be very careful because:

a. I had to prevent data loss or corruption of the data.
b. Ensure that all the changes done by the user are included in the next reload.
c. Had to handle synchronization between in memory objects and file storage especially when multiple users or threads are involved.

2.) User input and Error Handling: -

The system is based on a menu like program and is completely dependent on user inputs. So a lot of things or logic was considered such as:

a. Managing invalid entries.
b. Providing proper response or feedback so that the user itself understands where he made the mistake.
c. Prevent the system from crashing by using try and except functions.

3.) Role Based Access Control: -

In the program, a strict difference between the users has been made:

a. This is done so that unauthorized actions can be prevented and permission logic is included.
b. As, it is also seen in real life that extra perks are given based on your membership status so I decided to include it in my project.

4.) Accurate Overdue and Notification Logic: -

Date-based notifications depended on accurate calculations which was not easy at all, as different time zones and clock irregularities were also considered.

5.) Scalability and Performance: -

Although the current project is small scale but can be scaled quite easily. For this:

a. Large scale data storing bases will be needed and JSON is not sufficient for that.
b. Had to perform tasks more quickly as the datasets increase.
c. Also, most likely we would have to switch from JSON to a proper database related solution.

6.) Testing and Debugging: -

It is the most important things to verify all the operations used,

a. Testing should cover all options and cases.
b. Debugging the errors should be in such a way that other methods are not affected.
c. This can require much significant effort.

Learning and key take ways:

I have a learnt a lot from this project and here are the few take aways I want to note down: -

1.) I have gained a lot of practical experience in writing this Python code for a real world problem which made several of my topics strong such as oops and using different files, something which I have done for the first time.

2.) It has improved my problem – solving skills by applying things which I learnt in the class such as user management, handling error and oops.

3.) Learned the importance of a clean program design in menu style, using JSON file to save the data and also make sure that errors are handled properly.

4.)With the help of this project, I developed a deeper understanding of data persistence and got an idea as to how to structure a program properly.

Future Enhancements: -

Lots of upgrades can happen and some crucial one of them are:

1.) The system can be upgraded and a graphical user interface(GUI) can be added.

2.)Cloud Based storage can also be added as in the future the data will increase a lot.

3.) Advance search options can also be added in the program and several algorithms can be used to suggest new books to the user based on their preferences.

4.) Features such as feedback collection, different languages support, more strict access controls and adding more intense features.

5.) Also, those features can be added where support for mobile applications are also enabled.

These additions will of course make my system better and make it better performing.

References: -

In this project, I have taken important references and help from official Python documentation and code articles to get idea of syntax and application of several new codes and built – in functions. Also, I checked geeksforgeeks website to get more code related idea. Also, I I watched videos on YouTube to get the idea. Also, I took ideas from watching VITYARTHI course videos.