Class 12th

Informatics Practices

Project Work

On

“LIBRARY MANAGEMENT SYSTEM USING PYTHON”

Name Of Student: Ronak Kumar Bothra

Name Of School: Modi Public School

Subject: Informatics Practices (CODE 065)

Session: 2024-25

Submitted To: Mr. Vijay Nagar

Date Of Submission: 16th December 2024

INDEX

S.No	Title	Page
1.	CERTIFICATE	1
2.	ACKNOWLEDGEMENT	2
3.	INTRODUCTION	3
4.	APPROACH	4
5.	FEATURES OF THE SYSTEM	5-14
6.	HOW TO USE THE PROGRAM	15
7.	CONCLUSION	16
8.	BIBLIOGRAPHY	17

CERTIFICATE

This is to certify that **Ronak Kumar Bothra** has successfully completed the Informatics Practices project for Class 12 under the CBSE curriculum. The project titled **Library Management System Using Python** was undertaken at **Modi Public School** during the academic year **2024-25**.

This project involved the application of various concepts covered in the Informatics Practices curriculum, including data handling using Pandas Library and programming in Python. The student demonstrated outstanding skills and creativity in executing the project, showcasing their ability to apply theoretical knowledge in practical scenarios.

We commend the student for their dedication and hard work throughout this project, which reflects their commitment to excellence in the field of Informatics.

Vijay Nagar Sir   Examiner

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to all those who supported me throughout the journey of my Informatics Practices project. This endeavor would not have been possible without the encouragement, guidance, and contributions of several individuals.

First and foremost, I would like to extend my sincere thanks to my teacher, Vijay Nagar Sir, whose unwavering support and insightful feedback were instrumental in shaping my project.

Also, I would like to thank my family, whose constant encouragement and understanding provided me with the motivation to pursue my academic goals. Their belief in my abilities and willingness to help me create a conducive study environment were invaluable.

Lastly, I would like to acknowledge the resources and materials provided by Modi Public School, which facilitated my research and project development.

To all who contributed to my project, thank you for your support. Your encouragement has been a beacon of motivation and has inspired me to strive for excellence.

INTRODUCTION

A Library Management System (LMS) is a type of software that assists in automating the organization of resources such as books, library members, and even transactions such as issuing and returning members. Libraries, in the past, have relied upon manual record keeping, which came with its own challenges such as being tedious as well as the chances of human error. Creating such a system will greatly assist the library by improving the overall accuracy and efficiency of operations.

This project is aimed at developing a Library Management System using Python programming language with its various libraries like Pandas for accomplishing many subtasks related to the management of a library. The emphasis is to develop a system that will help library staff in books and members and transactions maintenance with minimum input and maximum effectiveness as well as accuracy.

The system allows users (library staff or administrators) to perform a variety of tasks, including

- Managing Books and Members records.
- Handling Book Issue and Return Transactions.

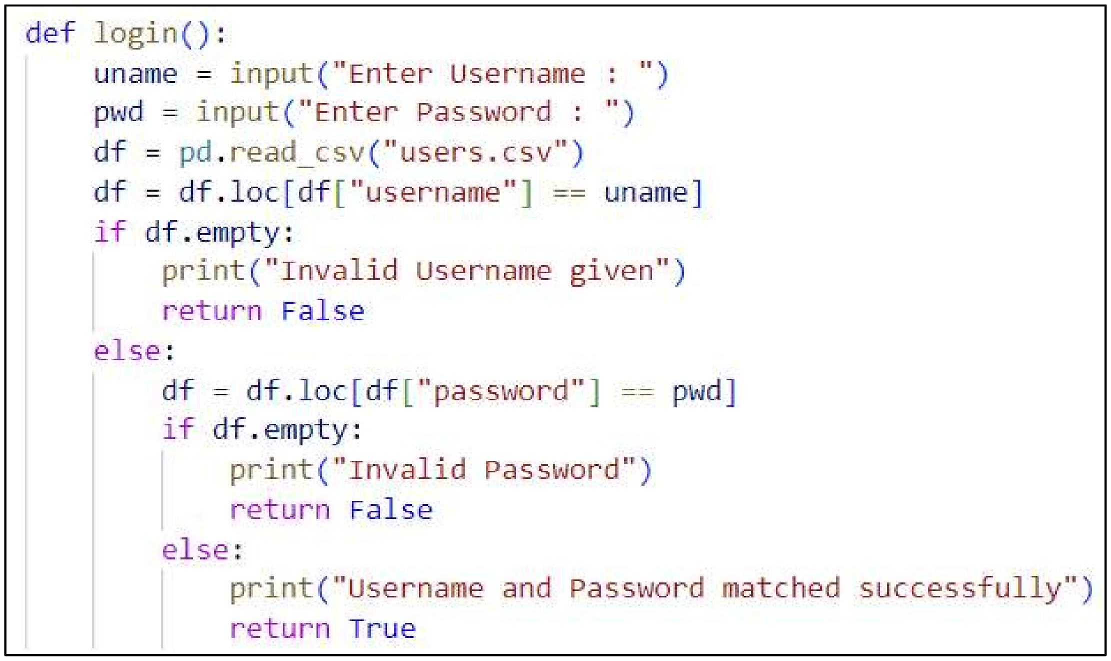
APPROACH

- **Planning**: The planning process began with analyzing the core tasks that needed to be automated in the Library Management System. This included identifying key operations such as adding and removing books from the library's collection, managing library members, and handling the borrowing and return of books. By breaking down these essential tasks, the project could be organized into manageable components, ensuring that each function was efficiently addressed and that the system would streamline the overall library operations.
- **Design**: The system was divided into modules:
  - **Book Management**: Includes adding new books, searching, deleting, and displaying all books.
  - **Member Management**: Includes adding members, searching, deleting, and displaying members.
  - **Issue and Return Management**: Handles issuing and returning books to/from members.
- **Implementation**: Python was used as the primary programming language, with CSV files as the storage medium for books, members, and issued books. Pandas is used for handling these CSV files, making data manipulation more efficient

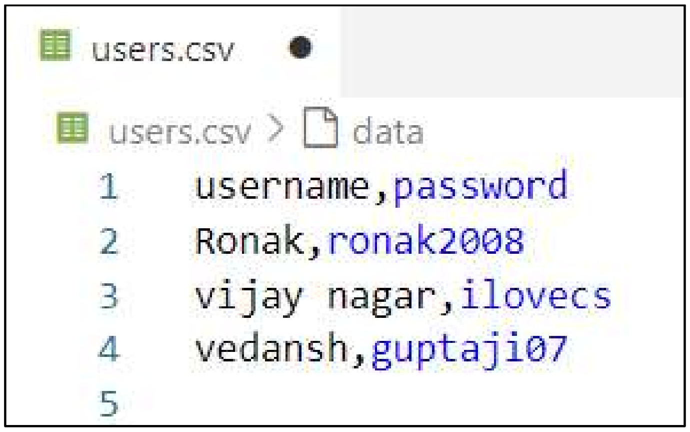
FEATURES

1. Admin Authentication:

A login system is implemented where users must enter a valid username and password to access the system. This is done by using a function called login() and storing the user login detail on a file named users.csv.



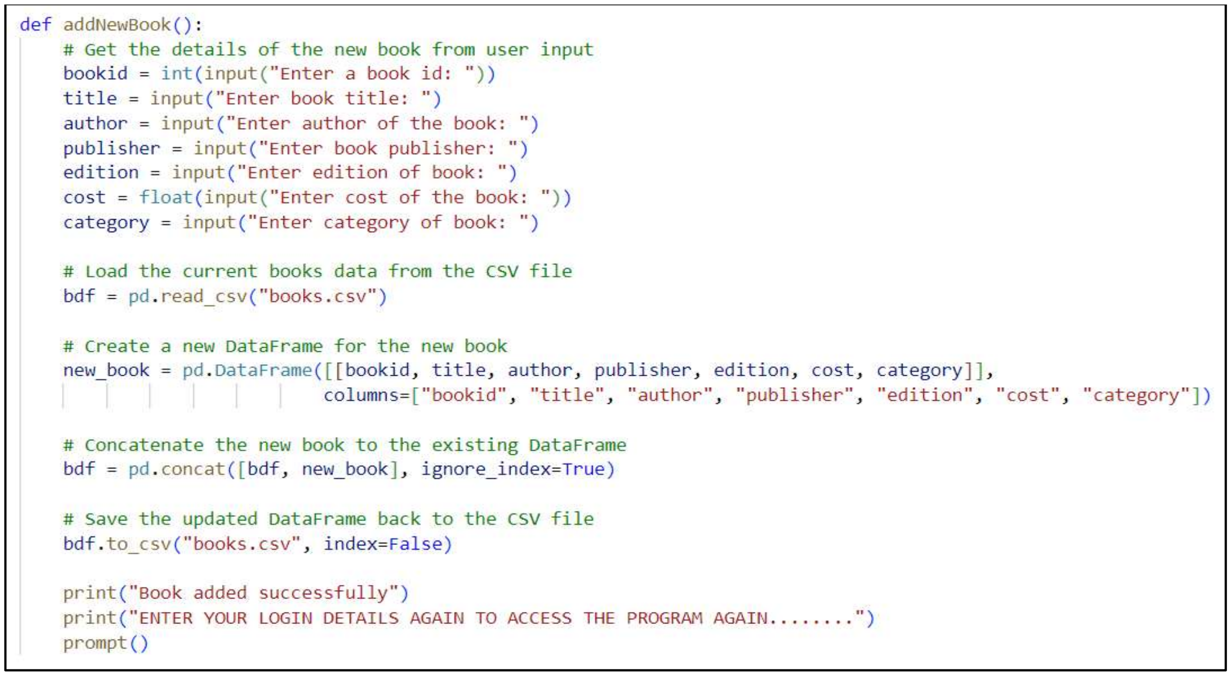
```python
def login():
    uname = input("Enter Username : ")
    pwd = input("Enter Password : ")
    df = pd.read_csv("users.csv")
    df = df.loc[df["username"] == uname]
    if df.empty:
        print("Invalid Username given")
        return False
    else:
        df = df.loc[df["password"] == pwd]
        if df.empty:
            print("Invalid Password")
            return False
        else:
            print("Username and Password matched successfully")
            return True
```



```
username,password
Ronak,ronak2008
vijay nagar,ilovecs
vedansh,guptaji07
```

2. Adding a New Book:

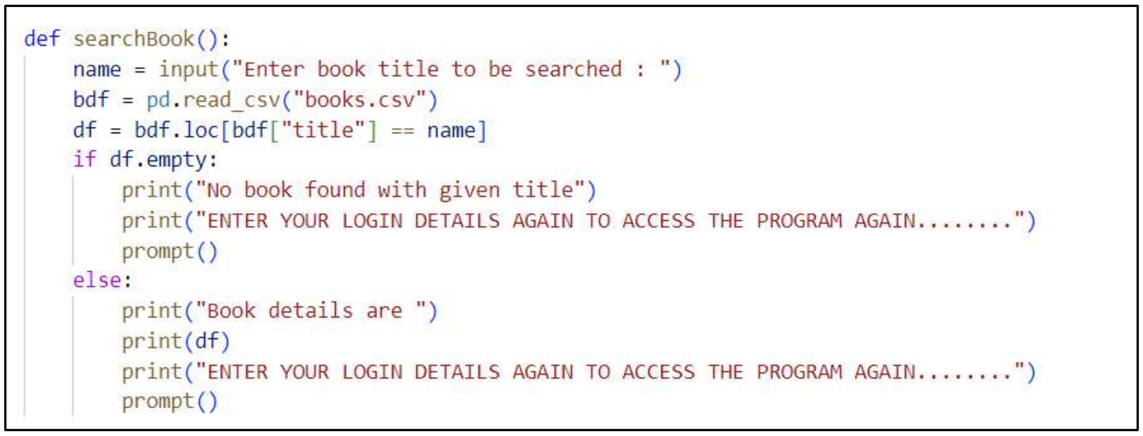
Users can add new books with attributes like book ID, title, author, publisher, edition, cost, and category. This is implemented using a function named addNewBook().



```python
def addNewBook():
    # Get the details of the new book from user input
    bookid = int(input("Enter a book id: "))
    title = input("Enter book title: ")
    author = input("Enter author of the book: ")
    publisher = input("Enter book publisher: ")
    edition = input("Enter edition of book: ")
    cost = float(input("Enter cost of the book: "))
    category = input("Enter category of book: ")

    # Load the current books data from the CSV file
    bdf = pd.read_csv("books.csv")

    # Create a new DataFrame for the new book
    new_book = pd.DataFrame([[bookid, title, author, publisher, edition, cost, category]],
                            columns=["bookid", "title", "author", "publisher", "edition", "cost", "category"])

    # Concatenate the new book to the existing DataFrame
    bdf = pd.concat([bdf, new_book], ignore_index=True)

    # Save the updated DataFrame back to the CSV file
    bdf.to_csv("books.csv", index=False)

    print("Book added successfully")
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
    prompt()
```

3. Searching for a Book:

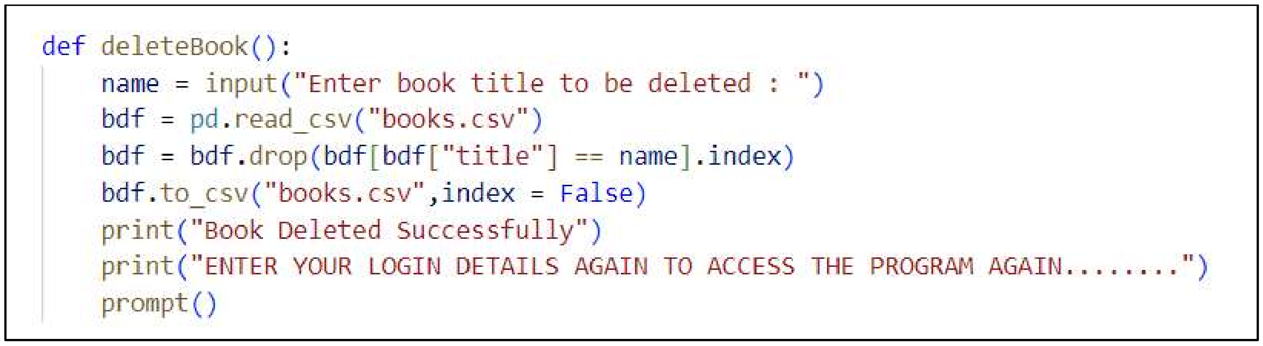
Allows searching for a book by its title. This is implemented by using a function called searchBook().



```python
def searchBook():
    name = input("Enter book title to be searched : ")
    bdf = pd.read_csv("books.csv")
    df = bdf.loc[bdf["title"] == name]
    if df.empty:
        print("No book found with given title")
        print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
        prompt()
    else:
        print("Book details are ")
        print(df)
        print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
        prompt()
```

4. Delete a Book:

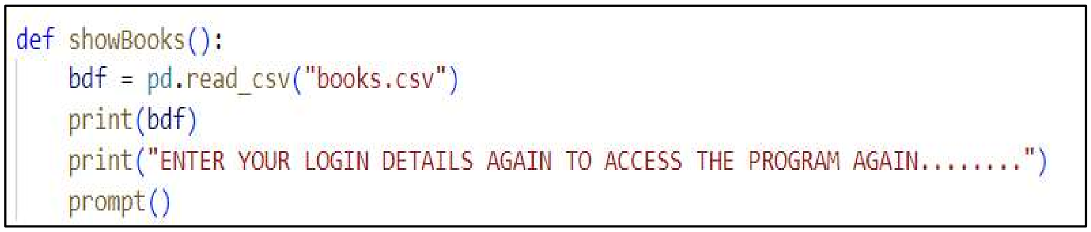
Enables deletion of a book by title. This is implemented by using a function called deleteBook().



```
def deleteBook():
    name = input("Enter book title to be deleted : ")
    bdf = pd.read_csv("books.csv")
    bdf = bdf.drop(bdf[bdf["title"] == name].index)
    bdf.to_csv("books.csv",index = False)
    print("Book Deleted Successfully")
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
    prompt()
```

5. Show all Books:

Displays a list of all available books in the library. This is implemented by using a function called showBooks()

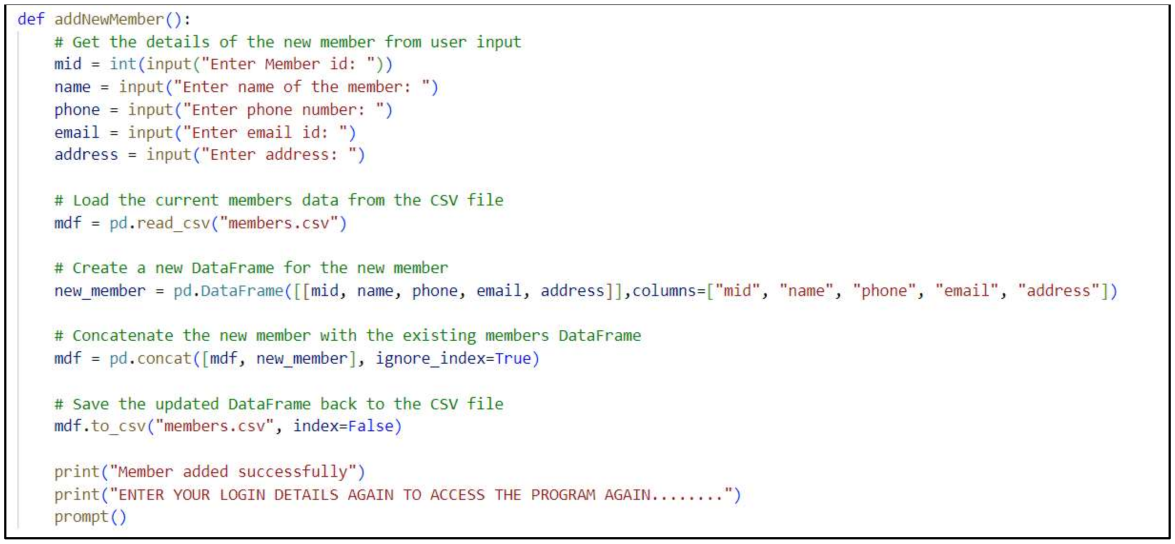


```
def showBooks():
    bdf = pd.read_csv("books.csv")
    print(bdf)
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN.......")
    prompt()
```

____

6. Adding a Member:

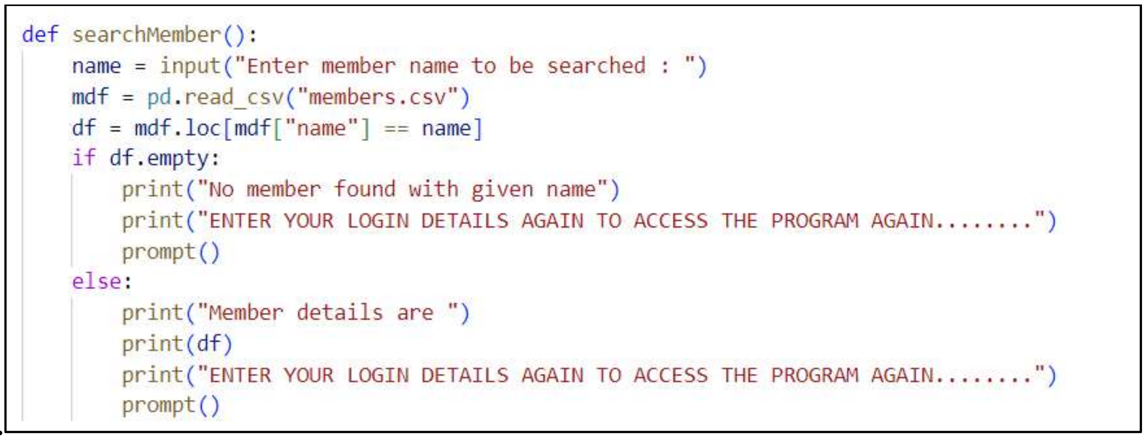
Adds a new library member with details such as name, phone, email, and address. This is implemented by addNewMember() function.



```python
def addNewMember():
    # Get the details of the new member from user input
    mid = int(input("Enter Member id: "))
    name = input("Enter name of the member: ")
    phone = input("Enter phone number: ")
    email = input("Enter email id: ")
    address = input("Enter address: ")

    # Load the current members data from the CSV file
    mdf = pd.read_csv("members.csv")

    # Create a new DataFrame for the new member
    new_member = pd.DataFrame([[mid, name, phone, email, address]],columns=["mid", "name", "phone", "email", "address"])

    # Concatenate the new member with the existing members DataFrame
    mdf = pd.concat([mdf, new_member], ignore_index=True)

    # Save the updated DataFrame back to the CSV file
    mdf.to_csv("members.csv", index=False)

    print("Member added successfully")
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
    prompt()
```

7. Searching for a Member:

Allows searching for a member by name. This was implemented by the means of a function named searchMember().

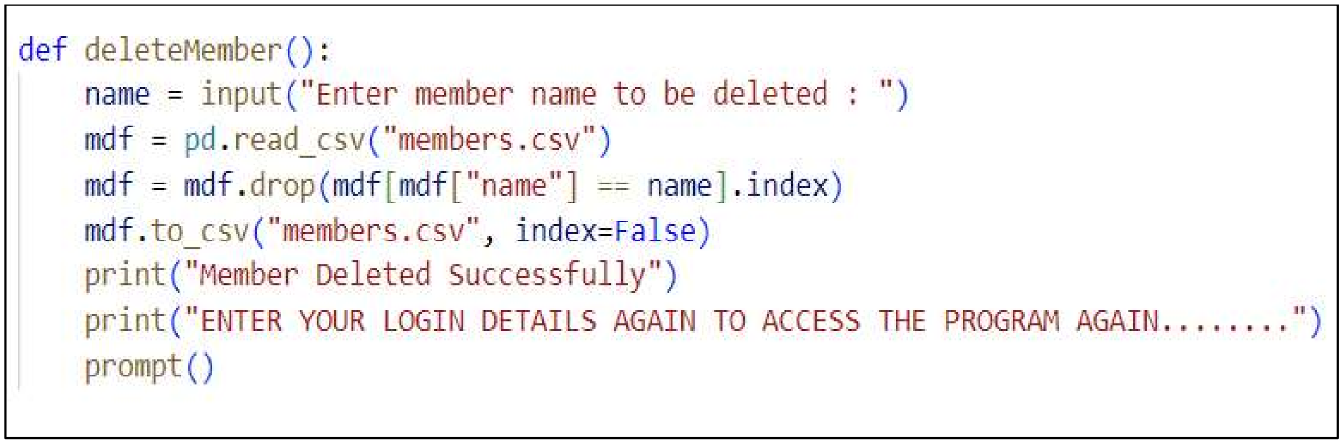


```python
def searchMember():
    name = input("Enter member name to be searched : ")
    mdf = pd.read_csv("members.csv")
    df = mdf.loc[mdf["name"] == name]
    if df.empty:
        print("No member found with given name")
        print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
        prompt()
    else:
        print("Member details are ")
        print(df)
        print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
        prompt()
```

.

8. Delete a Member:

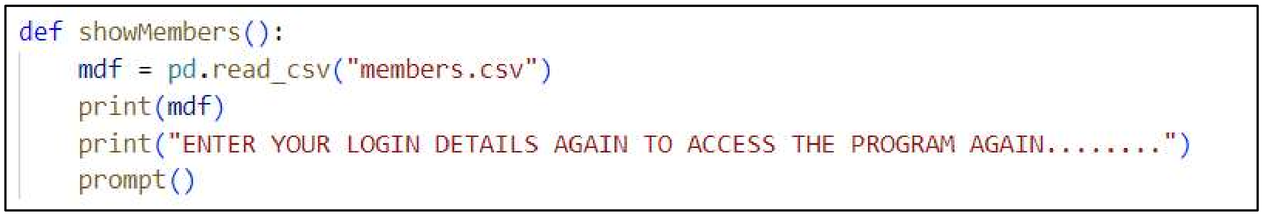
Enables deletion of a member by name. This is implemented by using a function called deleteMember().



```
def deleteMember():
    name = input("Enter member name to be deleted : ")
    mdf = pd.read_csv("members.csv")
    mdf = mdf.drop(mdf[mdf["name"] == name].index)
    mdf.to_csv("members.csv", index=False)
    print("Member Deleted Successfully")
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
    prompt()
```

9. Show all Members:

Displays a list of all members in the library. This is implemented by using a function called showMembers()

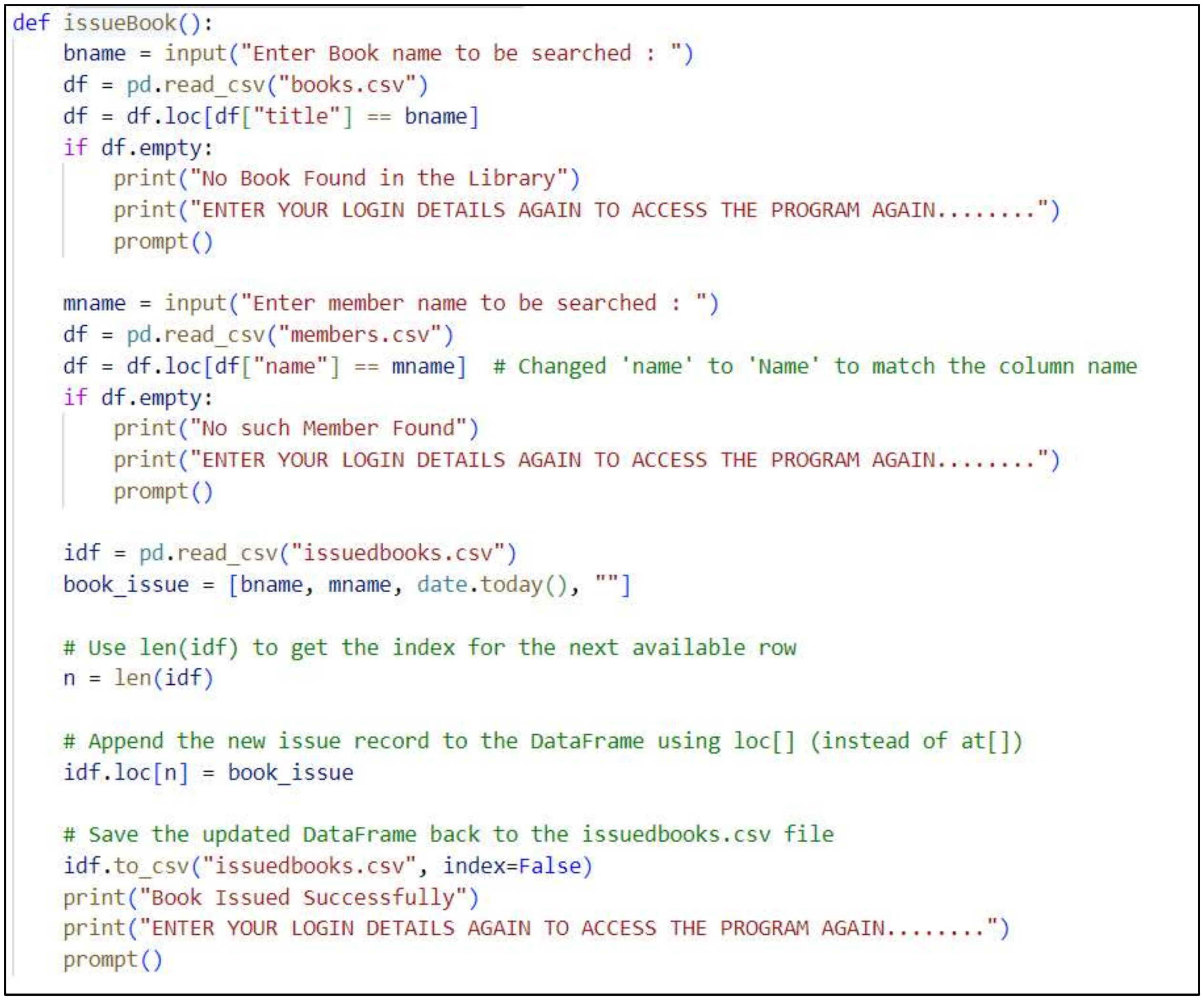


```
def showMembers():
    mdf = pd.read_csv("members.csv")
    print(mdf)
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
    prompt()
```

____

10. Issuing a Book:

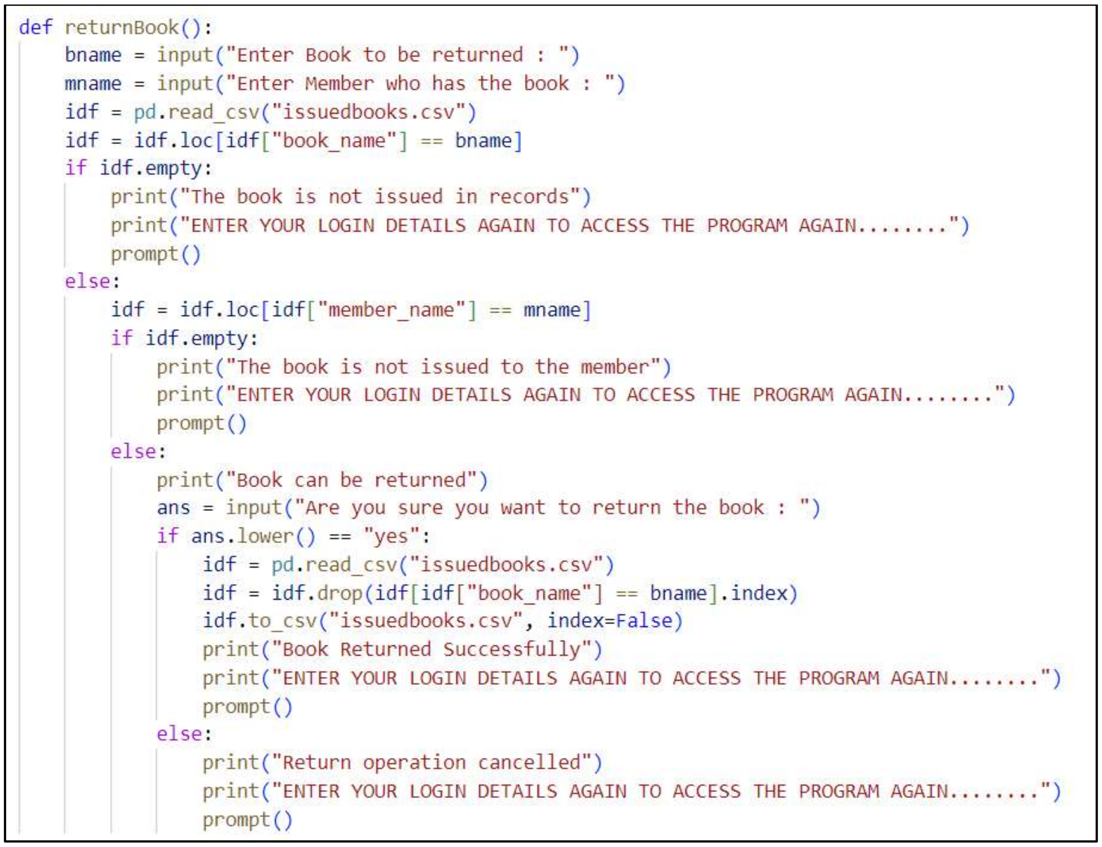
This is the feature of the LMS that deals with the transactions related stuff. It is down using the functions issueBook() along with the file issuedBooks,csv for store the data of the transactions.



```python
def issueBook():
    bname = input("Enter Book name to be searched : ")
    df = pd.read_csv("books.csv")
    df = df.loc[df["title"] == bname]
    if df.empty:
        print("No Book Found in the Library")
        print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
        prompt()

    mname = input("Enter member name to be searched : ")
    df = pd.read_csv("members.csv")
    df = df.loc[df["name"] == mname]  # Changed 'name' to 'Name' to match the column name
    if df.empty:
        print("No such Member Found")
        print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
        prompt()

    idf = pd.read_csv("issuedbooks.csv")
    book_issue = [bname, mname, date.today(), ""]

    # Use len(idf) to get the index for the next available row
    n = len(idf)

    # Append the new issue record to the DataFrame using loc[] (instead of at[])
    idf.loc[n] = book_issue

    # Save the updated DataFrame back to the issuedbooks.csv file
    idf.to_csv("issuedbooks.csv", index=False)
    print("Book Issued Successfully")
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
    prompt()
```

11. Returning a Book:

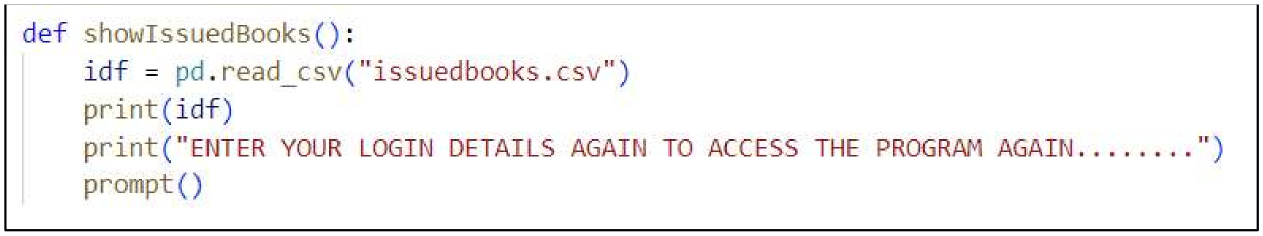
This is the feature of the LMS that also deals with the transactions related stuff. It is down using the functions returnBook().



```
def returnBook():
    bname = input("Enter Book to be returned : ")
    mname = input("Enter Member who has the book : ")
    idf = pd.read_csv("issuedbooks.csv")
    idf = idf.loc[idf["book_name"] == bname]
    if idf.empty:
        print("The book is not issued in records")
        print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
        prompt()
    else:
        idf = idf.loc[idf["member_name"] == mname]
        if idf.empty:
            print("The book is not issued to the member")
            print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
            prompt()
        else:
            print("Book can be returned")
            ans = input("Are you sure you want to return the book : ")
            if ans.lower() == "yes":
                idf = pd.read_csv("issuedbooks.csv")
                idf = idf.drop(idf[idf["book_name"] == bname].index)
                idf.to_csv("issuedbooks.csv", index=False)
                print("Book Returned Successfully")
                print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
                prompt()
            else:
                print("Return operation cancelled")
                print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
                prompt()
```

12. Showing all Issued Books:

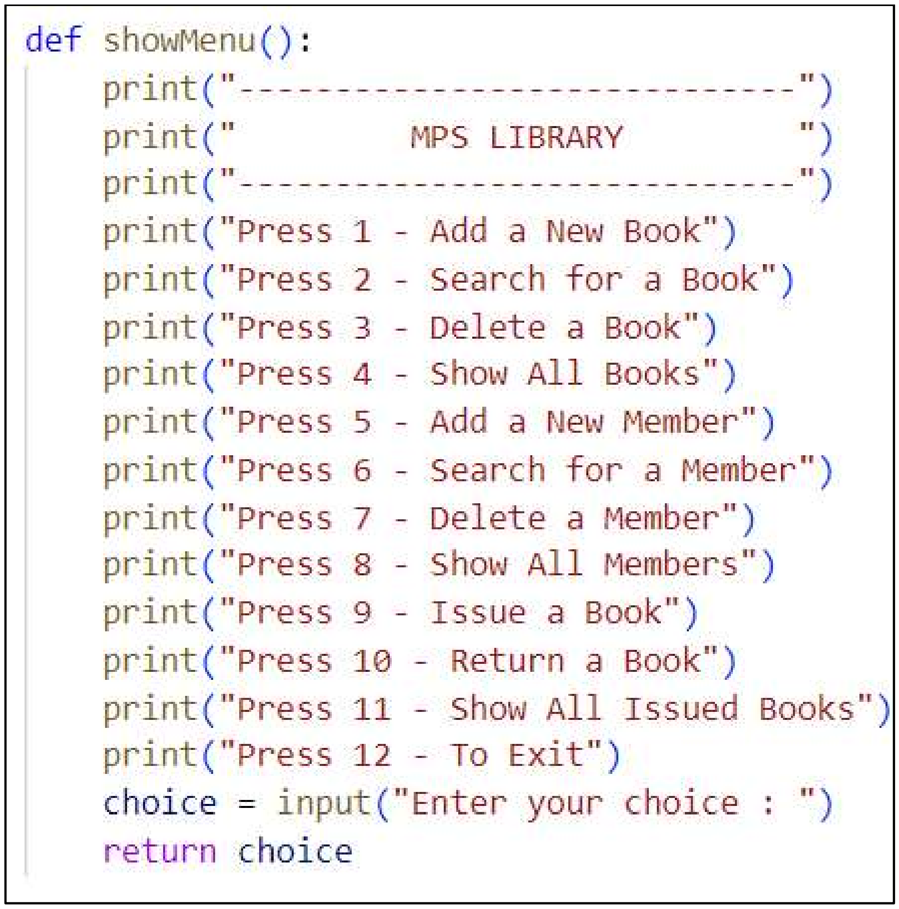
The showIssuedBooks() function does this while also auto entering the date of issue using the datetime library.



```
def showIssuedBooks():
    idf = pd.read_csv("issuedbooks.csv")
    print(idf)
    print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN........")
    prompt()
```

13. Main Menu:

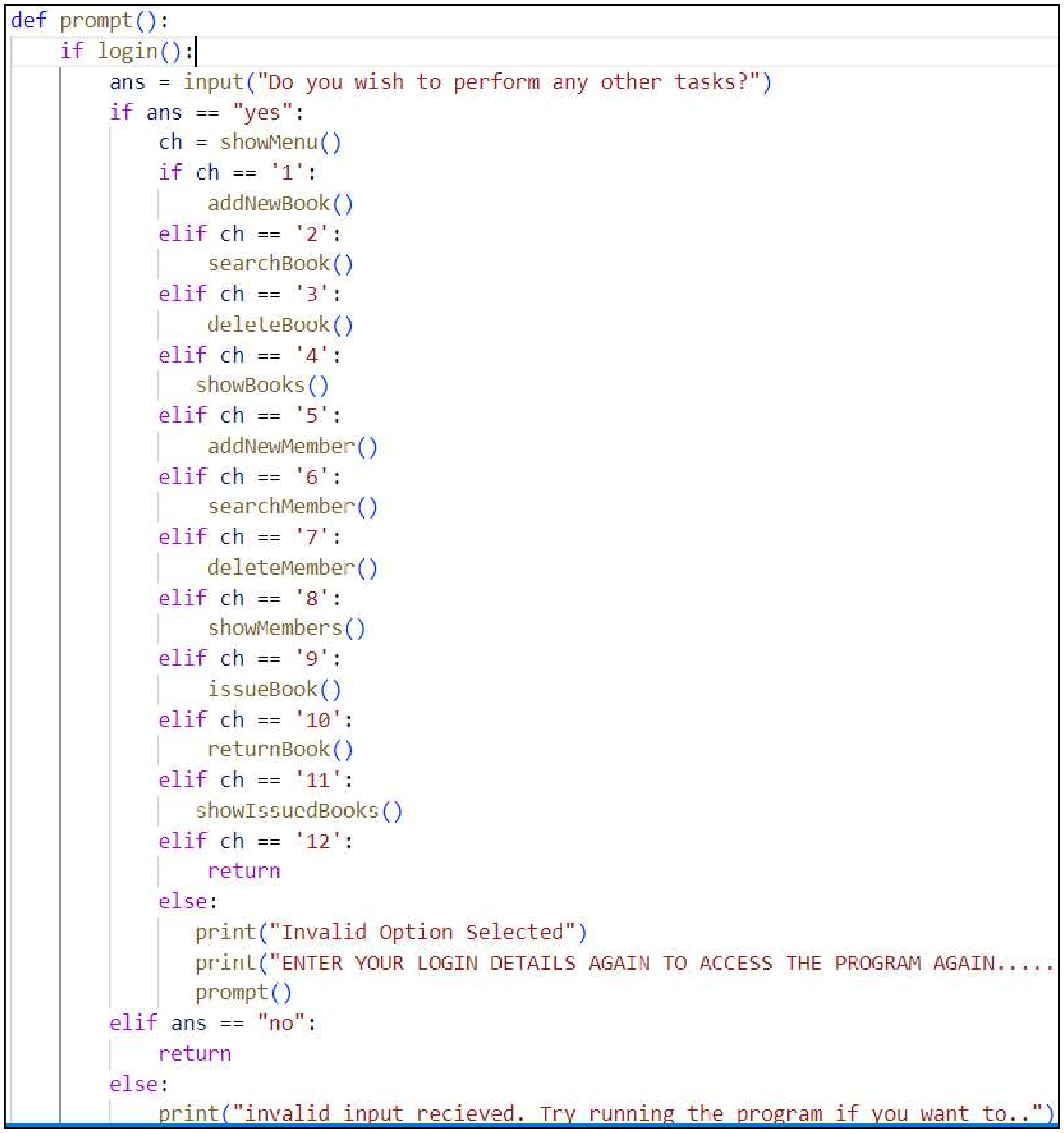
The main menu is presented using the showMenu() function as shown below:



```
def showMenu():
    print("-----------------------------")
    print("         MPS LIBRARY         ")
    print("-----------------------------")
    print("Press 1 - Add a New Book")
    print("Press 2 - Search for a Book")
    print("Press 3 - Delete a Book")
    print("Press 4 - Show All Books")
    print("Press 5 - Add a New Member")
    print("Press 6 - Search for a Member")
    print("Press 7 - Delete a Member")
    print("Press 8 - Show All Members")
    print("Press 9 - Issue a Book")
    print("Press 10 - Return a Book")
    print("Press 11 - Show All Issued Books")
    print("Press 12 - To Exit")
    choice = input("Enter your choice : ")
    return choice
```

14. Logic behind the Menu:

This function is basically connecting different functions with the menu.



```
def prompt():
    if login():
        ans = input("Do you wish to perform any other tasks?")
        if ans == "yes":
            ch = showMenu()
            if ch == '1':
                addNewBook()
            elif ch == '2':
                searchBook()
            elif ch == '3':
                deleteBook()
            elif ch == '4':
              showBooks()
            elif ch == '5':
                addNewMember()
            elif ch == '6':
                searchMember()
            elif ch == '7':
                deleteMember()
            elif ch == '8':
                showMembers()
            elif ch == '9':
                issueBook()
            elif ch == '10':
                returnBook()
            elif ch == '11':
              showIssuedBooks()
            elif ch == '12':
                return
            else:
              print("Invalid Option Selected")
              print("ENTER YOUR LOGIN DETAILS AGAIN TO ACCESS THE PROGRAM AGAIN.....
              prompt()
        elif ans == "no":
            return
        else:
            print("invalid input recieved. Try running the program if you want to..")
```

How to Use the Program

1. **Login**: When you run the program, you'll first be prompted to log in with a valid username and password. If authentication is successful, the system will allow access to the menu.
2. **Main Menu**: After logging in, you will see a menu with various options to manage books, members, and issued books. Here's a summary of the options:
    a. Press **1** to add a new book.
    b. Press **2** to search for a book by title.
    c. Press **3** to delete a book by title.
    d. Press **4** to display all books.
    e. Press **5** to add a new member.
    f. Press **6** to search for a member.
    g. Press **7** to delete a member.
    h. Press **8** to display all members.
    i. Press **9** to issue a book to a member.
    j. Press **10** to return a book.
    k. Press **11** to view all issued books.
    l. Press **12** to exit the program.
3. **Perform Operations**: Choose the desired option and follow the on-screen prompts to interact with the system (such as entering book titles, member names, etc.).
4. **Data Storage**: The program uses CSV files (books.csv, members.csv, issuedbooks.csv) to store data persistently. Ensure these files exist in the same directory as the Python script.

Conclusion

The Library Management System project offers a practical and effective solution for automating and streamlining various library operations, including book management, member management, and the tracking of issued books. By automating these essential tasks, the system significantly reduces the time and effort required for manual record-keeping, ensuring greater efficiency and accuracy in library management. This not only enhances the overall operation of the library but also improves the experience for both staff and members, allowing them to focus on more important activities rather than being bogged down by administrative work.

Furthermore, the system can serve as a foundation for future enhancements. Additional features, such as fine calculation for overdue books, advanced search filters for quicker book retrieval, and the integration of a graphical user interface (GUI), can be added to improve the user experience and increase the system's overall functionality. With these potential upgrades, the Library Management System can evolve into a more comprehensive tool that supports all aspects of library management, making it even more useful and adaptable to modern library needs.

Bibliography

1. **Python Documentation**:
   Official Python website for understanding Python libraries and their functionalities. https://docs.python.org
2. **Pandas Documentation**:
   Brilliant reference for the Pandas library, used for data manipulation and CSV file handling.
   https://pandas.pydata.org/docs
3. **Stack Overflow:** A platform that provided solutions and insights into common coding issues and challenges encountered during the project. https://stackoverflow.com

   Also, Class 12 NCERT of INFORMATICS PRACTICES WAS ALSO USED IN REFERENCE FOR PYTHON AND PANDAS BASICS.