|  |  |  |
| --- | --- | --- |
| https://lh7-us.googleusercontent.com/docsz/AD_4nXfypZyEB7KRYNRdUlSYALKHjxyOp80JUWM2u_0F4Rgz7WM89jUKZKPtRDYo56gYBM6UlTXOurdkjG2byKAjfas_DXmi4dDN2fN-lUvioQS3MP2WaosaVQdwDcVmFcimSuWya0K1Ftuj2TMDqsqaUNwVV7wWQxEH0Uq-eq0HGQ?key=Uio3m22x4g6CR9uyLgFNBw | **Charotar University of Science and Technology**  **Devang Patel Institute of Advance Technology and Research**  **Department of Computer Engineering** | https://lh7-us.googleusercontent.com/docsz/AD_4nXchK2Jf1VxjdZzew6f7QiMBTrw2SCcoCO6NJmgUE1cl0XtFC93Ng-5zR80E8RWnj1YG883YG4FasYLh7Qa9Vknpxb8_ePwVRHOnO1h0q7zcdX20j5kkld1MzYz6dYCO6v7PVyaOOo8UtQOzx2-DQMkbMA2g-1xUwfHaeus6IA?key=Uio3m22x4g6CR9uyLgFNBw |

**JAVA MINI PROJECT**

**LIBRARY MANAGEMENT SYSTEM**

**Main class:**

In the Library Management System, the flow begins from the main() method in the Library\_Management class.

**Flow:**

* main() initializes the Library object.
* The User object is created and associated with the Library.
* The system enters the interaction phase by calling user.start(), where the User interacts with the Library.

**User class:**

User class represents a user interacting with a library system. The user can perform various actions like adding books, borrowing, returning books, and displaying the list of books.

**Flow:**

1. Class Initialization:

* When a User object is created, it is passed an instance of the Library class (through the constructor User(Library library)), which it will use to perform operations on the library's collection.
* A Scanner object is initialized to handle input from the user.

1. User Interaction (start() method):

* The start() method is the central loop that continuously prompts the user to choose between various actions.

1. Menu Options: The user is presented with five options:

* Add a book.
* Borrow a book.
* Return a book.
* Display the list of available books.
* Exit the program.
* Based on the user's input, the program switches to the appropriate action (using a switch statement).
* The OUTER label is used to allow breaking out of the loop cleanly when the user chooses to exit the system by selecting option 5.

1. Option Handling:

* Add Book: Calls the addBook() method, which gathers input from the user (title, author, price) and creates a Book object that is added to the library by calling library.addBook().
* Borrow Book: Calls the borrowBook() method, which asks for the title of the book and calls library.borrowBook() to mark the book as loaned out.
* Return Book: Calls the returnBook() method, which asks for the title of the book and calls library.returnBook() to mark the book as returned.
* Display Books: Calls library.displayBooks() to show all the books available in the library.
* Exiting: When the user chooses to exit (option 5), the program breaks out of the loop using the OUTER label and displays an exit message.

**Book class:**

The Book class represents a book in the library with attributes like title, author, availability status, and price. This class encapsulates the basic operations related to a single book, such as checking availability, borrowing, and returning the book.

**Flow:**

1. Book Creation (Book() Constructor):

* When a new Book object is instantiated, it initializes the book's title, author, price, and availability status.
* By default, the isAvailable flag is set to true, meaning the book is available for borrowing at the time of creation.

1. Get Book Title (getTitle() method):

* The getTitle() method simply returns the title of the book, allowing other parts of the program (such as the User class or the Library class) to access the title.

1. Check Availability (isAvailable() method):

* This method returns the current availability status of the book (true or false).
* If isAvailable is true, the book is available to borrow; if false, the book is currently borrowed.

1. Borrow a Book (borrowBook() method):

* This method allows a book to be borrowed. If the book is available (i.e., isAvailable == true), it changes the availability status to false, meaning the book is no longer available for borrowing.
* If the book is already borrowed (isAvailable == false), it prints a message indicating that the book is not available.

1. Return a Book (returnBook() method):

* This method is used to return a book to the library. It sets the isAvailable flag back to true, meaning the book can be borrowed again.

1. Display Book Details (toString() method):

* The toString() method is overridden to provide a string representation of the book's details, including its title, author, price, and availability status.
* This method is useful for displaying book information when listing books in the library.

**Library class:**

The Library class represents a collection of books and provides methods for adding, borrowing, returning, and displaying books. It uses an ArrayList<Book> to manage the collection of book.

**Flow:**

1. Library Initialization (Library() Constructor):

* When a Library object is instantiated, it initializes the books list as an ArrayList<Book>.
* After that, the initializeDefaultBooks() method is called to populate the library with some default books.

1. Initialize Default Books (initializeDefaultBooks() method):

* This method adds a set of default books to the library's collection.
* Each book is instantiated using the Book class and added to the books list.
* This ensures that the library starts with a collection of predefined books.

1. Add a Book (addBook() method):

* This method allows a user to add a new book to the library collection.
* The Book object is passed as an argument and added to the books list using the add() method of ArrayList.

1. Borrow a Book (borrowBook() method):

* The borrowBook() method allows a user to borrow a book from the library by searching for the book by its title.
* It iterates over the books list, checking each book's title. If the title matches, it calls the borrowBook() method of the Book class to mark the book as borrowed.
* If the book is not found in the list, it prints a message indicating that the book was not found.

1. Return a Book (returnBook() method):

* Similar to borrowBook(), this method allows a user to return a borrowed book by searching for the book by its title.
* It iterates through the books list and, if the book is found, it calls the returnBook() method from the Book class to mark the book as available.
* If the book is not found, it prints a message saying the book was not found.

1. Display All Books (displayBooks() method):

* The displayBooks() method iterates over the books list and prints out the details of each book.
* It calls the toString() method of the Book class to display the formatted details of each book (title, author, price, and availability).