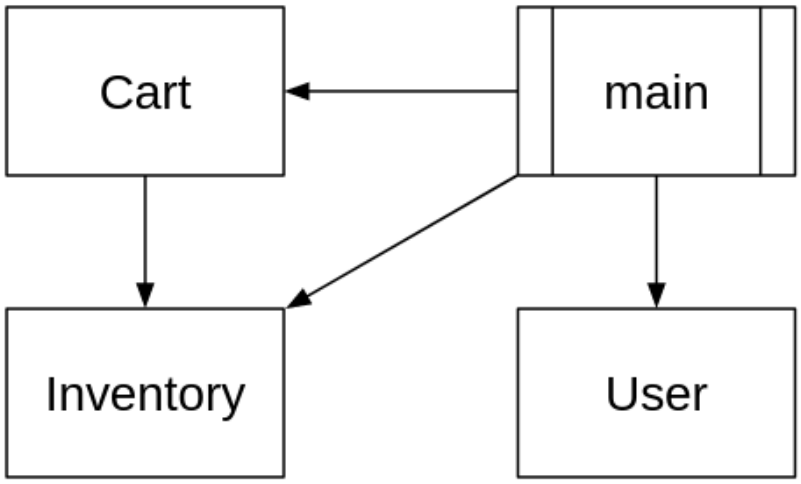
I. Project Overview Diagram



There will be 3 classes overall, along with a main program (with a main function included). Overall, the class interaction is minimal, primarily relying on the main to tie functionality together. However, the Cart class does have a couple items reliant on the Inventory class to finish completing. Details for this is given in the detailed class diagram for the Cart class following.

II. Detailed Class Diagrams

II.I. Inventory Class

Inventory
- databaseName: string - tableName: string
+ Inventory() + Inventory(string databaseName, string tableName) + viewInventory(): void + searchInventory(): void + decreaseStock(string ISBN): void

Description:

Class designed to handle everything related to the store inventory. The class will interface with an inventory database to handle cross session storage.

Assumptions Made:

The inventory class has minimal class variables for the entire class – data storage is handled by the database and because of that, we don't need anything enough to store it cross class. Individual functions can have variables within if necessary. What we *do* have (which aids in dynamic database allocation), is storing the database name and table name each function may use to access the database. Moreso important since SQLite3 databases are local to the machine.

Additionally, the functionality here is limited by the scope – there's no add/remove to inventory implemented so it limits what the class needs. Otherwise, using a data class for Books might be helpful. It's not so much required here, though.

Finally, input and output is utilized in this class.

Function Descriptions:

- Inventory(): Zero constructor. Nothing needs to be initialized here, but it allows for the functions to be used.

- Inventory(string databaseName, string tableName): Initially sets up the database / table names that will be used throughout the class to access the appropriate database and table
- viewInventory(): Displays all items in the inventory in some formatted way
- searchInventory(): Asks for a *title*, checks the database to see if a result is returned on that name. If so, display all results. If not, the user is informed their search failed
- decreaseStock(string ISBN): Called with a single ISBN parameter and decreases the stock number in the appropriate database for the appropriate ISBN
- Getters / Setters: Appropriate traditional getters and setters for the class variables can be added to supplement the code

II.II. User Class

User
- databaseName: string - tableName: string - loggedIn: bool - userID: string
+ User() + User(string databaseName, string tableName) + login(): bool + logout(): bool + viewAccountInformation(): void + createAccount(): void + getLoggedIn(): bool + getUserID(): string

Description:

This class is primarily used for the User data – handling the login, logout, account creation, etc. It handles all the interactions with the User database.

Assumptions Made:

Like with the Inventory class, this one has fairly minimal class variables, but it does have more than the inventory. You have the same database handlers to handle the locally stored database, but you end up with two extras. This class uses two to help validate whether someone is logged in or not: a boolean that's flagged true once someone's gone through the login process and one to track the current user's ID. This is returned to main to use in the Cart class, primarily. Note that the boolean logged in tracker is marked false initially and moved back to false once logout occurs.

Finally, input and output is utilized in this class.

Function Descriptions:

- User(): Zero constructor. Nothing needs to be initialized here, but it allows for the functions to be used. (userID should be empty and loggedIn should start at false)
- User(string databaseName, string tableName): Initially sets up the database / table names that will be used throughout the class to access the appropriate database and

table. Please note: the constructor is NOT used to log the user into the system. (userID should be empty and loggedIn should start at false)

- login(): Handles input, validation, and verification to allow someone to login. Updates the loggedIn boolean and returns true/false back to the original place of calling upon successful login. Sets the userID to the user who logged in for tracking
- logout(): Resets the userID back to blank and loggedIn back to false – returns false to be used in main
- viewAccountInformation(): Displays all the appropriate account information to the currently logged in user – query based on tracked userID in the class
- createAccount(): Contains all input and output related to creating an account – validation included. Once a correct person's inputs are gathered, handles database interaction appropriately
- getLoggedIn(): Traditional getter for loggedIn
- getUserID(): Traditional getter for userID
- Getters / Setters: Appropriate traditional getters and setters for the other class variables can be added to supplement the code

II.III. Cart Class

Cart
- databaseName: string - tableName: string
+ Cart(): + Cart(string databaseName, string tableName) + viewCart(string userID, string inventoryDatabase): void + addToCart(string userID, string ISBN): void + removeFromCart(string userID, string ISBN): void + checkOut(string userID): void

Description:

This handles all Cart interactions including interfacing with the Cart database defined below.

Assumptions Made:

Continuing from the previous classes, there's minimal class variables here. The Cart class has the user ID sent from the main (gathered from the User class) for a lot of the functionality – to make sure cart items are linked to the appropriate user. This is used as a parameter instead of a class variable to aid in limited changes needing to occur upon logout of a user. So, it makes for a more seamless process.

This also has the only instance of one class calling another – relying on the Inventory class for some of the functions.

Finally, input and output is utilized in this class.

Function Descriptions:

- Cart(): Zero constructor. Nothing needs to be initialized here, but it allows for the functions to be used.
- Cart(string databaseName, string tableName): Initially sets up the database / table names that will be used throughout the class to access the appropriate database and table
- viewCart(string userID, string inventoryDatabase): Displays all books in the logged in User's cart. Please note: this cooperates with the inventory database to display all the correct information on the inventory items – it just selectively shows the books in the User's cart

- addToCart(string userID, string ISBN): This relies on the user viewing the inventory previously – from the main. Once they select a book, this ISBN is used to add an item to the appropriate cart
- removeFromCart(string userID, string ISBN): This relies on the user viewing the cart previous – from the main. Once they select a book to remove, this ISBN is used to remove an item from the user's cart
- checkOut(string userID): The user checks out – this removes all their cart items. It also calls the Inventory class function to decrease the stock of the books by the correct amount the user bought (prior to removing them from the cart)

III. Menu Information

Please note that not all functions are going to be covered by the menu. Items such as the decreaseStock() and getUserID(), etc are used behind the scenes to aid other functions in the code.

It's worth mentioning that logout should link back to the pre-login menu options.

The Inventory and Cart Information menus don't need to loop until they're ready to leave – but, a go back option should be provided in case they don't want to perform an action in the consequent sub-menu. The Main Menu *should* loop until they're ready to leave. Error check for menu options is provided throughout to ensure a valid menu option is selected.

Before Login:

- Login
- Create Account
- Logout

After Login (Main Menu):

- Logout
- View Account Information
- Inventory Information
- Cart Information

After Login (Inventory Information)

- Go Back
- View Inventory
- Search Inventory

After Login (Cart Information)

- Go Back
- View Cart
- Add Items to Cart
- Remove an Item from Cart
- Check Out

IV. Information Storage

SQLite3 databases will be utilized throughout.

For the users table, payment is just expecting something like… PayPal, Discover card, etc. It's more a formality to get you used to a category existing more than anything.

Key:
* PRIMARY KEY
** FOREIGN KEY

Inventory Table

ISBN *
Title
Author
Genre
Pages
ReleaseDate
Stock

User Table

UserID *
Email
Password
FirstName
LastName
Address
City
State
Zip
Payment

Cart Table

UserID **
ISBN **
Quantity