Design Document

Methods and Tools in SW Development

**I. Group Information**

Group Number: 23

Group Member names/netIDs:

- John Thomas – jat710

- JoJo Kaler jk1885

- Franck Di Sanza fd210

- Charles Saffold cws330

What classes are you going to have? Explain why.

User Class – need class to identify users and store data for logging into and access accounts. Used to track users, their payment methods and addresses.

Inventory Class - Keep track of what items are still available on the store so the user knows what they can purchase.

Cart Class - tracks stored items users wish to purchase and links to which account is connected.

Books Class - identify and track books stored in inventory and number of each as well as author, ESRB rating, published date, and genre.

**II. Detailed Class Diagrams**

| Inventory Class |
| --- |
| -ISBN: int  -stock: int  -retailPrice: int  -purchaseCost: int |
| -addBook(ISBN: int, stock: int, retailPrice: int, purchaseCost: int): void  -removeBook(ISBN: int): void  -updateStock(ISBN: int): void  -updateRetailPrice(ISBN: int, retailPrice: int): void  -updatePurchasePrice(ISBN: int, retailPrice: int): void |

* addBook - Adds book to the inventory with its ISBN, available, purchase price and retail price.
* removeBook - Removes book from the inventory and all its data.
* updateStock - Updates the stock available for the book.
* updatePurchasePrice - Updates the purchase price of the book.
* updateRetailPrice - Updates the retail price of the book.

| User Class |
| --- |
| -username: string  -password: string  -email: string  -displayName: string  -address: string  -zip: int  -preferredPayment: string  -cartHistory: cart |
| +User(username,password,displayName,address,zip): void  +deleteAccount(username): void  +login(username,password): void  +logout(): void  +changeAddress(): void  +changePassword(): void  +changeUsername(): void  +changeEmail(): void  +changePreferredPayment (): void  +changeZip(): void  +changedisplayName(): void |

* User - Used to create an instance of the user class and initialize all of its data.
* deleteAccount - Used to delete this instance of the class and thus deleting the user’s account
* login - If the user has an existing account, will allow the user to login with their username and password.
* logout - The user will logout of the system.
* changeAddress - will change the user’s address
* changePassword - will change the user’s password
* changeUsername - will change the user’s username
* changeEmail - will change the user’s email
* changePreferredPayment - will change the user’s preferred payment
* changeZip - will change the user’s ZIP

| Cart Class |
| --- |
| -priceTotal: int  -ISBN: int  -retailPrice: int  -stock: int  -orderId: int |
| +checkInventory(stock: int): bool  +addItem(ISBN: int): void  +removeitem(ISBN: int): void  +calcTotal(retailPrice: int, priceTotal: int): int  +checkout(orderId: int, priceTotal): void  +updateInventory(stock: int): void |

* checkInventory(stock) - check the availability of the item in stock
* addItem(ISBN) - used to add an item to the cart
* removeitem(ISBN) - used to remove an item from the cart
* calcTotal(retailPrice, priceTotal) - used to calculate the total price of the purchase
* checkout(orderId, priceTotal) - checks the user out and completes the purchase
* updateInventory - used to update the inventory when a purchase is made

| Book Class |
| --- |
| + ISBN: int  -bookName: string  -bookGenre: string  -bookAuthor: string  -bookDate: dateTime |
| -newBook(ISBN, bookName, bookGenre, bookDate): void  -setBookName(ISBN): void  -setBookGenre(ISBN): void  -setBookDate(ISBN): void  -setBookAuthor(ISBN): void  -getBookGenre(ISBN): string  -getBookAuthor(ISBN):string  -getBookName(ISBN): string  -getBookDate(ISBN): dateTime |

* newBook(ISBN, bookName, bookGenre, bookDate) - called when needed to create a new book and set all according values for the book within the system. Requires all values to be non-null ISBN or returns error.
* setBookName(ISBN) - function for altering a pre-existing book’s name if needed. Uses the ISBN to alter the associated book’s name, returning nothing on success. Fails if no ISBN is entered or an ISBN unconnected to a book is entered.
* setBookGenre(ISBN) - function for altering a pre-existing book’s genre if needed. Uses the ISBN to alter the associated book’s genre, returning nothing on success. Fails if no ISBN is entered or an ISBN unconnected to a book is entered.
* setBookAuthor(ISBN) -function for altering a pre-existing book’s author if needed. Uses the ISBN to alter the associated book’s author, returning nothing on success. Fails if no ISBN is entered or an ISBN unconnected to a book is entered.
* setBookDate(ISBN) - function for altering a pre-existing book’s date if needed. Uses the ISBN to alter the associated book’s date, returning nothing on success. Fails if no ISBN is entered or an ISBN unconnected to a book is entered.
* getBookName(ISBN) - retrieves the name of a book, outputting the name if successful. Fails if ISBN does not exist or if no ISBN is entered.
* getBookGenre(ISBN): function for returning the book’s genre as a string based off the ISBN; throws error if no value/no ISBN in system after search.
* getBookAuthor(ISBN) - function returns the name of the book referred by the ISBN associated with it, returns error if no ISBN input or no ISBN in system matching search
* getBookDate(ISBN) - returns the date the book of the ISBN was published, returning an error if no ISBN input or no ISBN in system matching search.

**III. Menu Information**

Before login:

● Login

● Create Account

● Exit Program

After login:

* Check all books
  + Next page
  + Previous page
  + Add to cart
  + Go back
* Search for Book
  + Search for author
  + Search for date
  + Search for ISBN
  + Search for name
  + Search for genre
  + Go Back
* Cart Information
  + View Cart
  + Remove Item from Cart
  + Check total price
  + Go back
* Checkout
  + Checkout
  + Go back
* Edit account
  + Edit username
  + Edit password
  + Edit displayed name
  + Edit address
  + Edit zip
  + Edit preferred payment method
  + Go back
* View order History
  + Go back
* Logout
  + Confirm
  + Go back
* Exit Program
  + Confirm
  + Go back

Does your menu cover all requirements given? If not, explain why certain requirements don’t have a distinct menu option?

Yes

**IV. Information Storage**

How is your group storing information?

Include one of these lines of questioning based on your storage schema:

● If a database, what kind of database?

We will be using databases primarily. We will be using lists to store data in tables. This makes it easier to access within the functions and classes. These tables will consist of instances of the other classes. For example, the inventory table will contain instances of the book class.

What information are you going to store in each (table / file depending on schema)?

* *Users*
  + *username*
  + *Displayname*
  + *Password*
  + *Email*
  + *Address*
  + *Zip*
  + *Preferred payment*
* *Cart*
  + *ISBN*
  + *Retail price*
  + *Total price*
  + *Order Number*
* *Inventory*
  + *ISBN*
  + *Stock*
  + *Retail price*
  + *Purchase price*
* *Books*
  + *ISBN*
  + *title*
  + *Genre*
  + *Author*
  + *Date Published*