Design Document

Methods and Tools in SW Development

1. **Group Information**

Group Number: 1

Group Member names/netIDs:

* Levi Gist
  + Lig17
* Joshua McCoy
  + Jdm1502
* David Doss
  + Dpd115
* Russell Smith
  + res565
* Ian Goforth
  + Img56

What classes are you going to have? Explain why.

* User
  + This will allow the drive file to manage multiple users with individual attributes. These include the users name, shopping cart, order history, address, and payment information.
* History
  + This class holds the attributes of the individual items in the order history. These attributes include the items, name, category, ID, and price.
* PayInfo
  + This class stores the individual user's payment information
* Cart
  + The cart class allows the user to add and remove things from the cart and eventually checkout. The cart keeps track of the total cost and also updates the users order history.
* Catalog
  + The catalog class allows to have as many items as needed for the inventory. The catalog stores attributes such as the item’s ID, price, name, and category. The catalog class allows things to be add and removed accordingly from the catalog.
* Shipping
  + This class contains the attributes for the shipping information. This includes the state, city ,zip, and a unique ID.

1. **Detailed Class Diagrams**

|  |
| --- |
| User |
| -username:string  -userID:int  -password:string  -email:string  -ship:Shipping  -payment\_info:PayInfo  -user\_cart:Cart  -user\_history:vector<History> |
| +~User():  +setShipID(id:int)void  +setuserID(id:int)void  +setName(n:string):void  +setPassword(p:string):void  +setEmail(e:string):void  +getName(): string  +getPassword(): string  +getEmail():string  +getUserID()int  +getShipID()int  +assignShip(ad:int,s:string,c:string,z:int)void  +assigninfo(cn:int,cvs:int,exd:int):void  +addItem(id:int,cat:vector<Catalog>)bool  +removeItem(id:int,cat:vector<Catalog>)bool  +viewCart():void  +checkout()void  +viewHis()void |

* ~User()-> Allows the current user to be deleted
* setShipID(id:int)->sets the unique shipping info ID for the user.
* setName(n:string)->sets the username attribute equal to desired username being passed in by reference.
* setPassword(p:string)->sets the password attribute equal to the desired password passed in by reference.
* setEmail(e:string)->sets the email attribute equal to the email being passed in by reference.
* setuserID(id:int)->this method sets the ID for the individual user.
* getName()->returns the value stored in username.
* getPassword()->returns the value stored in password.
* getEmail()->returns the value stored in email
* getUserID()->this method returns the ID for the individual user.
* getShipID()->returns the unique Shipping info ID for the user.
* assignShip(ad:int,s:string,c:string,z:int)->this method takes in the shipping information and stores it in the Shipping class
* assigninfo(cn:int,cvs:int,exd:int)->sets the Creditcard number, cvs, and exp date values into the Payment\_information class.
* viewCart()->Displays the users cart in the Cart class.
* addItem(id:int,cat:vector<Catalog)->This method adds the item with the corresponding ID to the cart through the cart class and updates the catalog. It returns false if the ID did not match any of the items.
* removeItem(id:int,cat:vector<Catalog>)->This method removes the item with the corresponding ID from the cart through the cart class and updates the Catalog. It returns false if the ID did not match any of the items in the cart.
* viewCart()-> This method displays the current contents of the cart.
* checkout()->this method checks out the users cart.
* viewHis()->this method displays the order history of the given user.  
  \*\* each detailed class diagram should be on its own page!

|  |
| --- |
| Shipping |
| -address: string  -ShipID:int  -state:string  -city:string  -zip:int |
| +setAd(a:string)void  +setID(id:int)void  +setState(s:string)void  +setCity(c:string)void  +setZip(z:int)void  +getAd()string  +getState()string  +getCity()string  +getZip()int  +getID() int |

* setAd(a:string)->sets the address attribute equal to the desired address passed in by reference .
* setCity(c:string)->sets the city attribute equal to the desired city being passed in by reference.
* setState(s:string)-> sets the state attribute equal to the desired state being passed in by reference.
* setZip(z:int)->sets the zip attribute equal to the desired zip being passed in by reference.
* setID(id:int)->this method sets the unique ID for the shipping info
* getAd()->returns the value stored in address.
* getCity()->returns the value stored in city.
* getState()->returns the value stored in state.
* getZip()->returns the value stored in zip.
* getID()->returns the shipping ID for the current information.

|  |
| --- |
| History |
| -itemName: string  -itemCategory:string  -itemPrice:double  -itemID:int |
| +setName(n:string)void  +setCat(c:string)void  +setPrice(p:double)void  +setID(id:int)void  +getName()string  +getCat()string  +getPrice()string  +getID()int |

* setName(n:string)->This method sets the name of the Item that is going into the order history.
* setCat(c:string)->This method sets the name of the category for the item that is going into the order history.
* setPrice(p:double)->This method sets the price of the item that is going in the order history.
* setID(id:int)->This method sets the id for the item going in the order history
* getName()->This method returns the name of the item in the order history.
* getCat()->This method returns the name of the category for the item in the order history.
* getPrice()->This method returns the price of the item in the order history.
* getID()->This method returns the ID of the item in the order history.  
  \*\* each detailed class diagram should be on its own page!

|  |
| --- |
| PayInfo |
| -card\_num:int  -exp\_date:int  -CSV:int  -payID:int |
| +setPayID(id:int)void  +set\_card\_num(cn:int)void  +set\_exp\_date(ed:int)void  +set\_CSV(sec:int)void  +get\_card\_num()int  +get\_exp\_date()int  +get\_CSV():int  +getPayID():int |

* setPayID(id:int)->This method sets the unique ID for the current payment information.
* set\_card\_num(cn:int) -> sets the card number to card\_num
* set\_exp\_date(ed:int) -> sets the expiration date to exp\_date
* set\_CSV(sec:int) -> sets the card verifation to CSV
* get\_card\_num() -> returns card\_num
* get\_exp\_date() -> returns exp\_data
* get\_CSV() -> returns CSV
* getPayID()->Returns the unique ID for the current payment information.  
  \*\* each detailed class diagram should be on its own page!

|  |
| --- |
| Cart |
| -cart:int[]  -cartID:int  -total\_cost:int  -orderHist:vector<History>  -checked\_out:bool |
| +setCartID(id:int)void  +getCardID()int  +add(id:int,cat:vector<Category>):bool  +remove(id:int,cat:vector<Category>):bool  +view\_cart():void  +checkout(oh:vector<History>):void |

* setCartID(id:int)->This method sets the ID for the current cart.
* getCartID(id:int)->This method returns the ID for the current cart.
* add(id:int,cat:vector<Category>) -> this method adds the item ID to the cart array. This method also updates the current total cost and updates the current inventory availability. Returns true if the item was successfully added false otherwise.
* remove(id:int,cat:vector<Catalog>) ->this methods removes the item ID from the cart array. This method also updates the current total cost and the inventory availability. Return true if it is added otherwise it returns false .
* view\_cart() -> this method traverses through the array displaying the id and the corresponding item name with the ID and the current total cost.
* checkout(oh:vector<History>) -> This method changes the bool to true and then displays the total and updates the order history before destroying the current cart.
* \*\* each detailed class diagram should be on its own page!

|  |
| --- |
| Catalog |
| -name:string  -itemID:int  -price:double  -category:string |
| +setName(n:string)void  +setID(id:int)void  +setPrice(p:double)void  +setCat(c:string)void  +getName()string  +getID()int  +getPrice()double  +getCat()string |

* setName(n:string)->This method sets the name of the item in the catalog.
* setID(id:int)->sets the unique Item id in the catalog
* setPrice(p:double)->This method sets the price for the item in the catalog.
* setCat(c:string)->this method sets the category name for the item in the catalog.
* getName()->returns the name of the item in the catalog.
* getID()->returns the unique ID to the item in the catalog.
* getPrice()->returns the price of the item in the catalog.
* getCat()->returns the category name of the item in the catalog.

1. **Menu Information**

Before Login:

* Sign In
* Sign Up
* Exit Program

After Login:

* View Books
* View Movies
* View Cart
* Add to cart
* Edit Shipping Info
* Edit payment Info
* View Order History
* Logout
* Exit Program

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

The menu covers all the requirements given. The user must login in order to start adding or removing things to their cart. Once in the menu the user can view the items available, then choose to add it to their cart. They can also edit their shipping and payment info. Then after viewing their cart, they can choose to checkout and their order history would then be updated. This order history can also be looked at from the menu.

1. **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?
  + How many database tables will you have?
  + SQL
  + 7

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

* User table
  + userID->primary
  + Username
  + Shipping ID->foreign
  + PaymentID->foreign
  + cartID->foreign
  + Order history->foreign
* PayInfo table
  + card\_num->primary
  + exp\_date
  + CSV
* Cart table
  + User ID->primary
  + Price
  + total
  + Item ID->foreign
* Catalog Table
  + Item ID->primary
  + Name
  + Price
  + Category
* History table
  + Item ID->primary
  + Name
  + Price
* Shipping table
  + Shipping ID->primary
  + State
  + City
  + Zip