|  |
| --- |
| **User** |
| **-userId: string**  **-password: string**  **-loginStatus: string**  **-accountType: string** |
| **+verifyLogin(): bool** |

* verifyLogin(userId,: string, password: string) retrieves the username and password from the user and verify the login based on the login status and account type whether the account is of customer or seller. After the retrived username , password and account type the login is verified based on Boolean type.

|  |
| --- |
| **Customer** |
| **-customerName: string**  **-address: string**  **-email: string**  **-userName: string**  **-password: string**  **-customerId: int**  **-shippingAddress: string**  **-accountType: string** |
| **+createAccount()**  **+login()**  **+updateProfile()**  **+logout()**  **+accountInfo()**  **+verifyAccount(): string** |

* createAccount ( customerName: string, address: string, email: string, userName: string, password: string, customerId: int, shippingAddress: string, accountType: string ): This method generates the account from the attributes of customerName, address, email, username, password, customerId, shippingAddress and accountType and saves it to database of our system.
* Login( userName: string, password: string, ) : The login method takes the userName and password and authenticates if the user is in the system or not.
* updateProfile(customerName: string, address: string, email: string, userName: string, password: string, customerId: int, shippingAddress: string, accountType: string): Updates the customer information if they want to change it. It would be userName, password, customerId and shippingAddress. )
* Logout() : redirects the customer to the homepage of the system once the method is run.
* verifyAccount() : This method checks whether the account is valid or invalid based on what the user created.

|  |
| --- |
| Seller |
| -SellerId: String  -AccountBalance: float  -totalAddedProduct: int |
| +addProduct():void  +productListed():void  +removeProduct(): void  +modifyProduct: void |

addProduct(): this method adds the product in the inventory of the system by the seller

productListed (addProduct(): void) : It retrieve the product added on the inventory and list out all the product.

removeProduct(totalAddedProduct: int) : This method modifies the product. It takes the parameter totalAddedProduct.

modifyProduct() : It modifies the product based on the attributes.

|  |
| --- |
| **Administrator** |
| **-adminName: string**  **-email: string** |
| **+viewComplaints(): void**  **+verifySellerAccount(): string** |

viewComplaints(adminName: string, email:string) : this method takes the attributes adminName and email to login as a Administrator and viewComplaints made by customer associated with the seller account.

verifySellerAccount(adminName: string, email: string) : verifies the seller account …..

|  |
| --- |
| Product |
| -productName: string  -productId: int  -productUnitCost: float  -productType: string  -sellerId-string  -dateAdded: Date  -productRentPrice: float |
|  |
|  |

Product class has the above attributes with the data types to ensure the administrator has added the product in the inventory and all the information is stored in a database calling method associated with the seller class. (addProduct() )

|  |
| --- |
| **Search** |
| -productId: int  -productType: string  -dateAdded: date |
| +searchProductByName: void  +searchProductByType: void  +searchProductByDateAdded: void  +searchProductByPrice():void |

|  |
| --- |
| **Shopping Cart** |
| -cartId: int  -productID: int  -quantity: int  -dateAdded: int |
| +addCartItem()  +updateQuantity()  +viewCartDetails()  +checkout() |

+addCartItem(productid: int, quantity:int ) : add the item to the shopping cart based on the cartId and the productid.

+updateQuantity() : updates the quantity in the shopping cart.

+viewCartDetails(quantity:int, dateAdded:int, productid: int) : Displays the cart details based on how many quantity the shopping cart has, the details of the product including productid and date added to the shopping cart.

+checkout(quantity:int , cartId:int) : checkout the products in the shopping cart.

|  |
| --- |
| **Order** |
| **-orderId: int**  **-dateCreated: string**  **-date Shipped: string**  **-customerName: string**  **-customerId: string**  **-status:string**  **-shipping:string** |
| +placeOrder() |

* placeOrder ( customerName: string, customerId: string, shipping:string, status:string) : Places the order based on the added quantity on the cart. It takes the following parameter when the order is placed.

|  |
| --- |
| **Payment** |
| -cardType: String  -cardName: string  -cardNumber: int  -expDate: date |
| +verifyPayment(): void |

* verifyPayment(cardType: string, cardName:string, cardNumber, expDate: date) : This method takes the parameter like cardType, cardName, cardNumber , expDate to ensure if the card verification is valid or not.

|  |
| --- |
| **Order Details** |
| -orderId: int  -productId: int  -productName: string  -quantity:int  -productUnitCost: float  -productRentPrice: float  -rentDuration: int  -subTotal:float |
| +calcPrice |

* +calcPrice ( productUnitCost: float, productRentPrice: float, rentDuration: int, subtotal:float ) : Calculates the price based on the unit cost of the product, unit rent price and the time duration the order is processed. Thus, it calculates the total price of the product while renting.