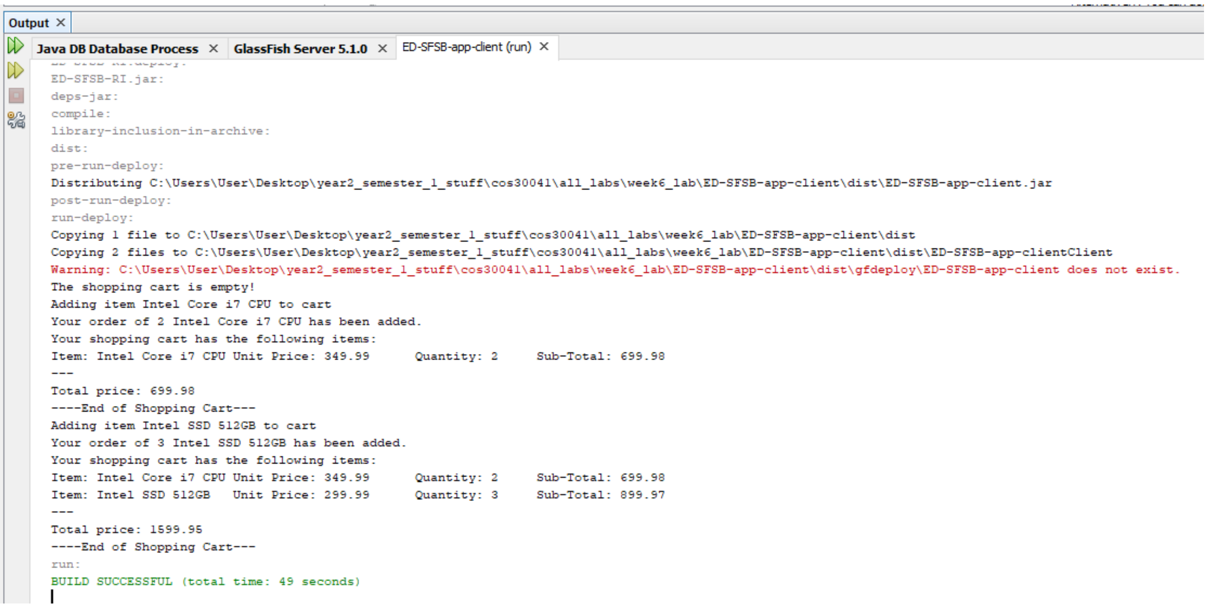
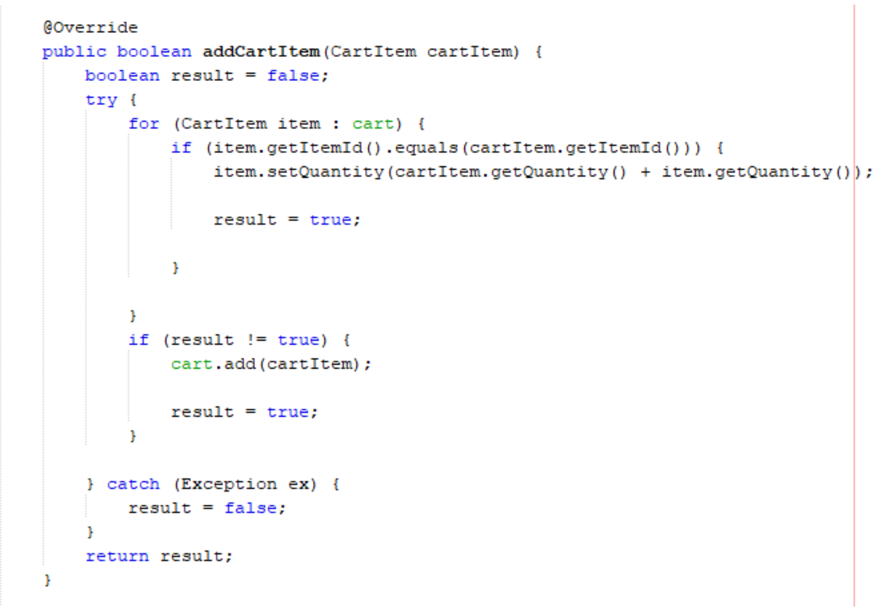
Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai

Step 1 ending part:



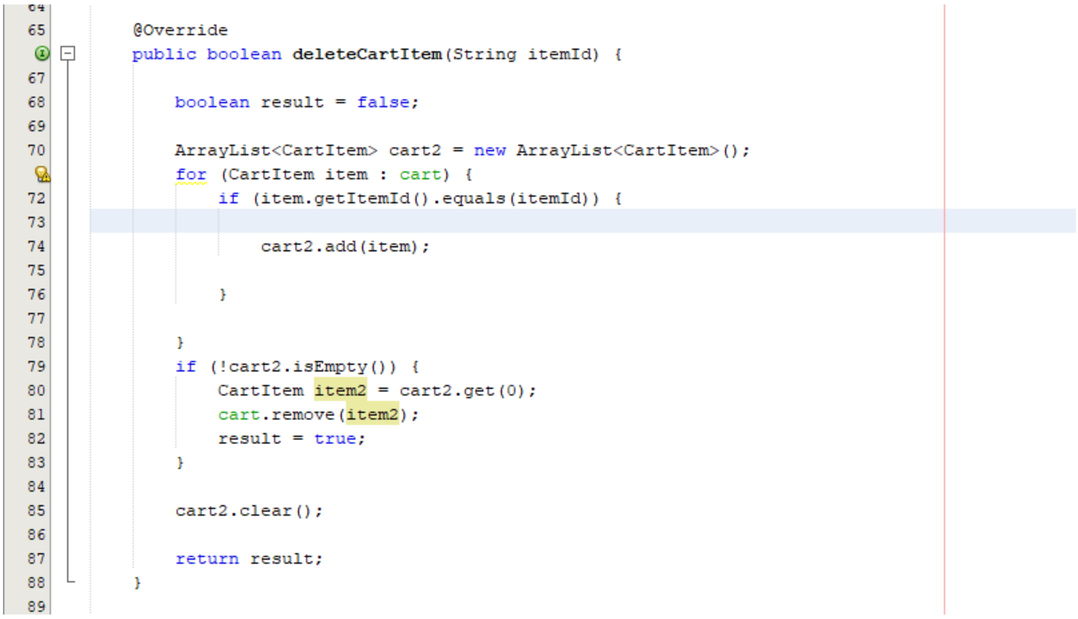
```
ED-SFSB-RI.jar:
deps-jar:
compile:
library-inclusion-in-archive:
dist:
pre-run-deploy:
Distributing C:\Users\User\Desktop\year2_semester_1_stuff\cos30041\all_labs\week6_lab\ED-SFSB-app-client\dist\ED-SFSB-app-client.jar
post-run-deploy:
run-deploy:
Copying 1 file to C:\Users\User\Desktop\year2_semester_1_stuff\cos30041\all_labs\week6_lab\ED-SFSB-app-client\dist
Copying 2 files to C:\Users\User\Desktop\year2_semester_1_stuff\cos30041\all_labs\week6_lab\ED-SFSB-app-client\dist\ED-SFSB-app-clientClient
Warning: C:\Users\User\Desktop\year2_semester_1_stuff\cos30041\all_labs\week6_lab\ED-SFSB-app-client\dist\gfdeploy\ED-SFSB-app-client does not exist.
The shopping cart is empty!
Adding item Intel Core i7 CPU to cart
Your order of 2 Intel Core i7 CPU has been added.
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
---
Total price: 699.98
----End of Shopping Cart---
Adding item Intel SSD 512GB to cart
Your order of 3 Intel SSD 512GB has been added.
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
Item: Intel SSD 512GB   Unit Price: 299.99      Quantity: 3     Sub-Total: 899.97
---
Total price: 1599.95
----End of Shopping Cart---
run:
BUILD SUCCESSFUL (total time: 49 seconds)
```

Step 2 all code parts:

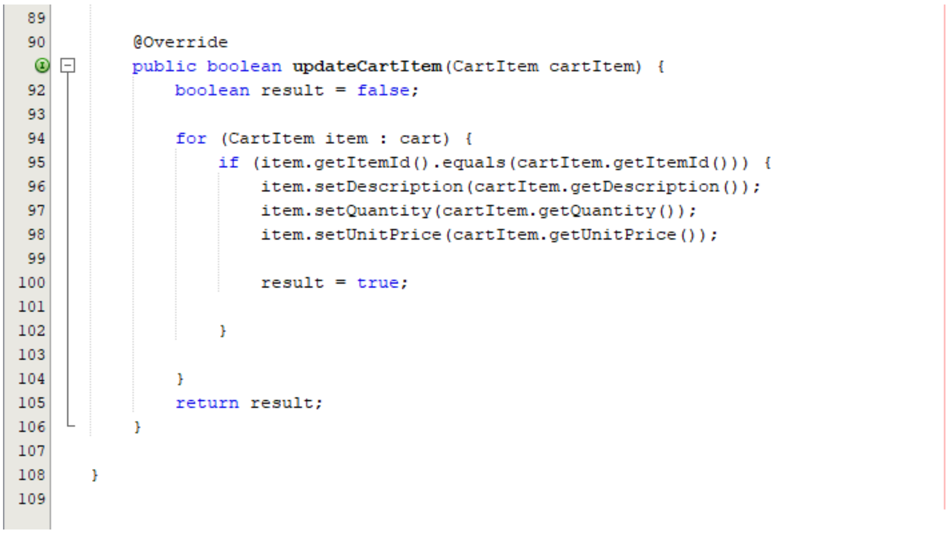


```java
@Override
public boolean addCartItem(CartItem cartItem) {
    boolean result = false;
    try {
        for (CartItem item : cart) {
            if (item.getItemId().equals(cartItem.getItemId())) {
                item.setQuantity(cartItem.getQuantity() + item.getQuantity());

                result = true;

            }

        }
        if (result != true) {
            cart.add(cartItem);

            result = true;
        }

    } catch (Exception ex) {
        result = false;
    }
    return result;
}
```

Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai



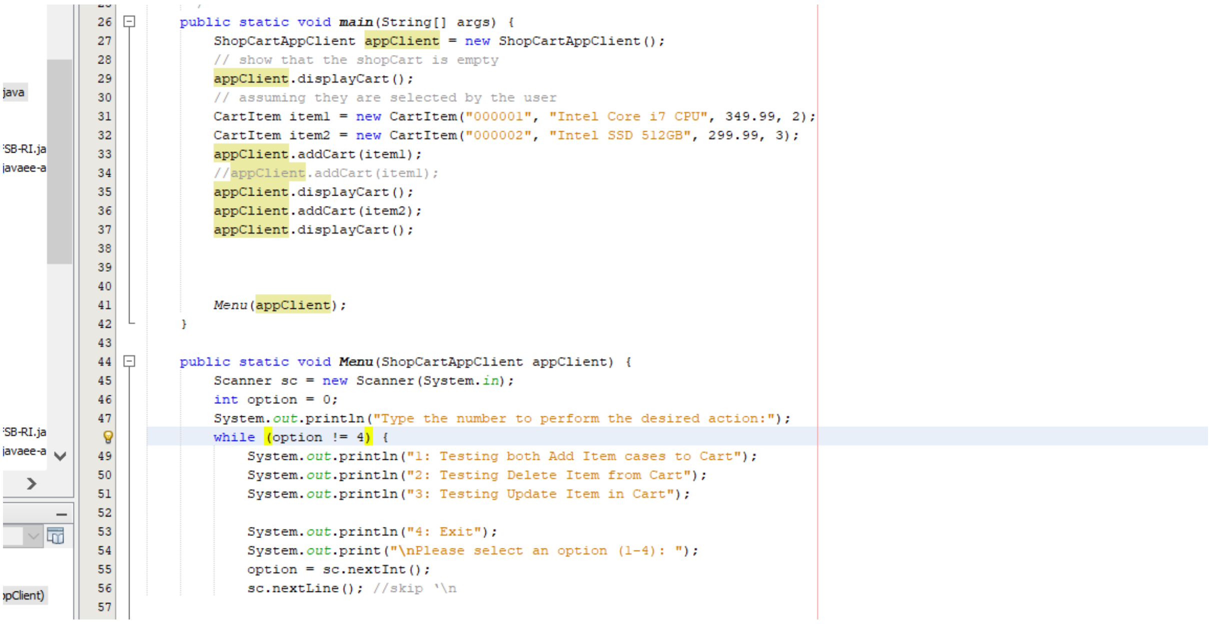
```java
    @Override
    public boolean deleteCartItem(String itemId) {

        boolean result = false;

        ArrayList<CartItem> cart2 = new ArrayList<CartItem>();
        for (CartItem item : cart) {
            if (item.getItemId().equals(itemId)) {

                cart2.add(item);

            }

        }
        if (!cart2.isEmpty()) {
            CartItem item2 = cart2.get(0);
            cart.remove(item2);
            result = true;
        }

        cart2.clear();

        return result;
    }

```



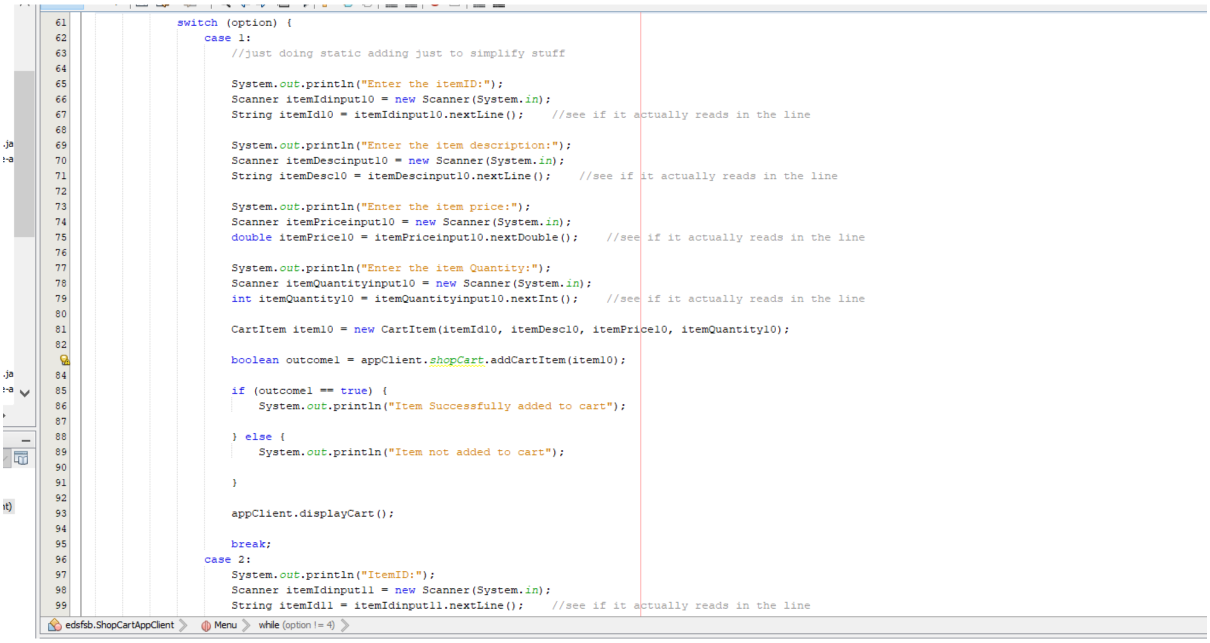
```java

    @Override
    public boolean updateCartItem(CartItem cartItem) {
        boolean result = false;

        for (CartItem item : cart) {
            if (item.getItemId().equals(cartItem.getItemId())) {
                item.setDescription(cartItem.getDescription());
                item.setQuantity(cartItem.getQuantity());
                item.setUnitPrice(cartItem.getUnitPrice());

                result = true;

            }

        }
        return result;
    }

}

```

Task 3 code parts:

Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai

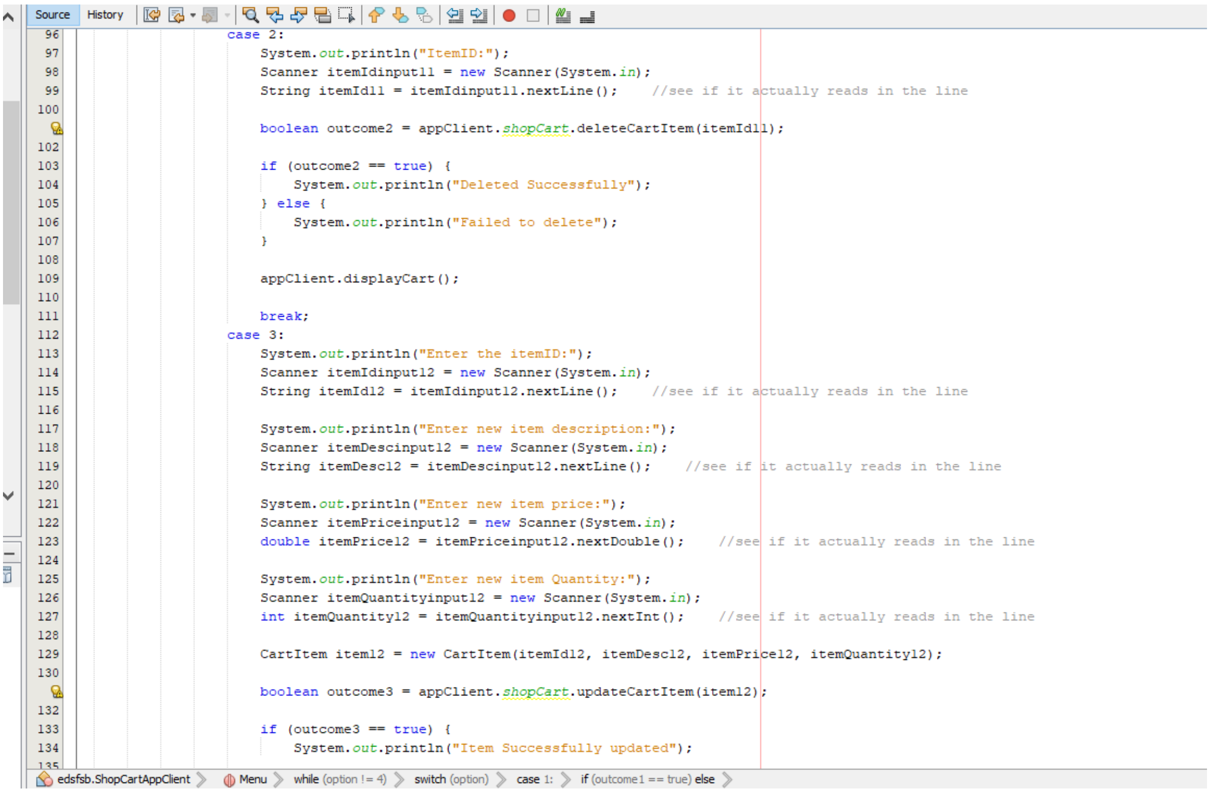


```java
    public static void main(String[] args) {
        ShopCartAppClient appClient = new ShopCartAppClient();
        // show that the shopCart is empty
        appClient.displayCart();
        // assuming they are selected by the user
        CartItem item1 = new CartItem("000001", "Intel Core i7 CPU", 349.99, 2);
        CartItem item2 = new CartItem("000002", "Intel SSD 512GB", 299.99, 3);
        appClient.addCart(item1);
        //appClient.addCart(item1);
        appClient.displayCart();
        appClient.addCart(item2);
        appClient.displayCart();



        Menu(appClient);
    }

    public static void Menu(ShopCartAppClient appClient) {
        Scanner sc = new Scanner(System.in);
        int option = 0;
        System.out.println("Type the number to perform the desired action:");
        while (option != 4) {
            System.out.println("1: Testing both Add Item cases to Cart");
            System.out.println("2: Testing Delete Item from Cart");
            System.out.println("3: Testing Update Item in Cart");

            System.out.println("4: Exit");
            System.out.print("\nPlease select an option (1-4): ");
            option = sc.nextInt();
            sc.nextLine(); //skip '\n
```

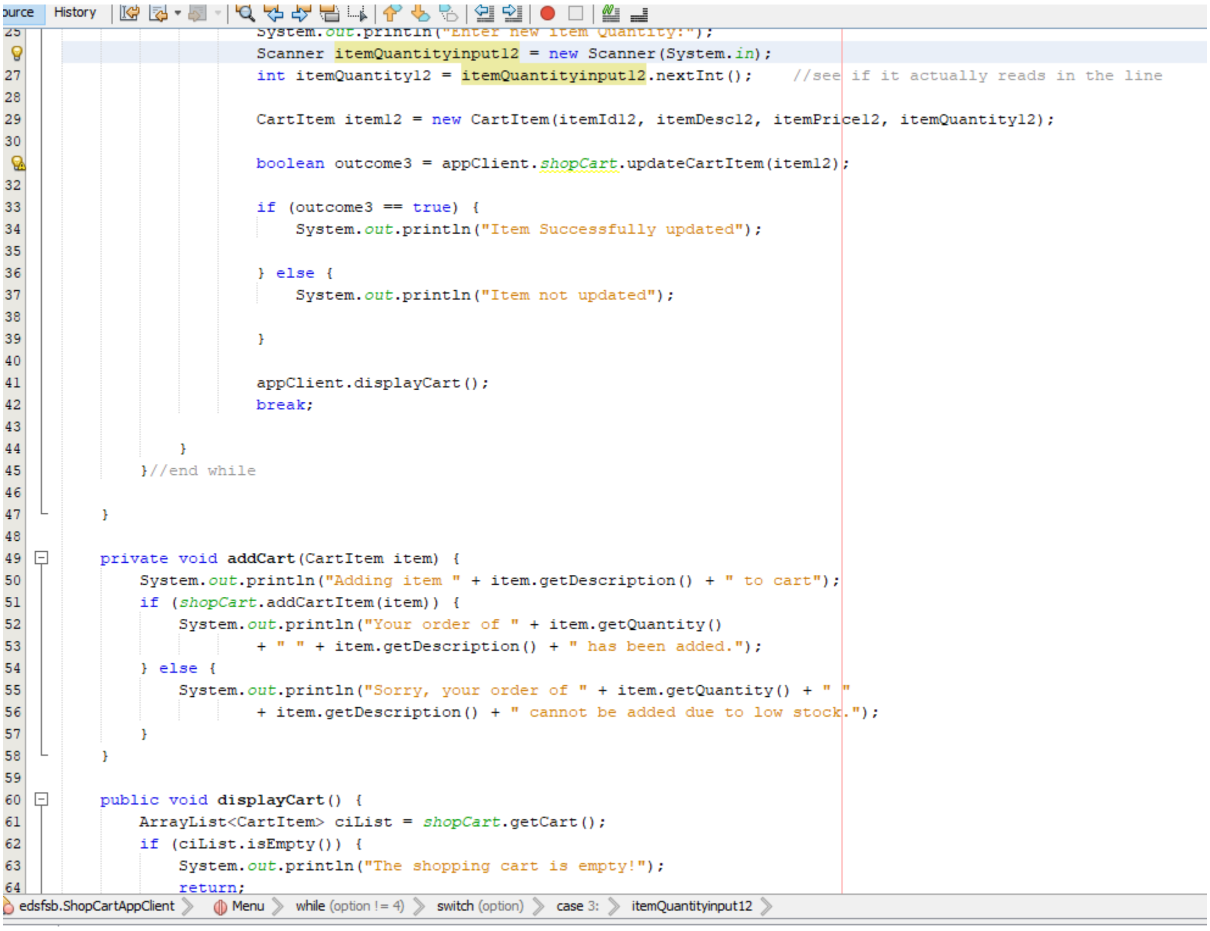


```java
            switch (option) {
                case 1:
                    //just doing static adding just to simplify stuff

                    System.out.println("Enter the itemID:");
                    Scanner itemIdinput10 = new Scanner(System.in);
                    String itemId10 = itemIdinput10.nextLine();    //see if it actually reads in the line

                    System.out.println("Enter the item description:");
                    Scanner itemDescinput10 = new Scanner(System.in);
                    String itemDesc10 = itemDescinput10.nextLine();    //see if it actually reads in the line

                    System.out.println("Enter the item price:");
                    Scanner itemPriceinput10 = new Scanner(System.in);
                    double itemPrice10 = itemPriceinput10.nextDouble();    //see if it actually reads in the line

                    System.out.println("Enter the item Quantity:");
                    Scanner itemQuantityinput10 = new Scanner(System.in);
                    int itemQuantity10 = itemQuantityinput10.nextInt();    //see if it actually reads in the line

                    CartItem item10 = new CartItem(itemId10, itemDesc10, itemPrice10, itemQuantity10);

                    boolean outcome1 = appClient.shopCart.addCartItem(item10);

                    if (outcome1 == true) {
                        System.out.println("Item Successfully added to cart");

                    } else {
                        System.out.println("Item not added to cart");

                    }

                    appClient.displayCart();

                    break;
                case 2:
                    System.out.println("ItemID:");
                    Scanner itemIdinput11 = new Scanner(System.in);
                    String itemId11 = itemIdinput11.nextLine();    //see if it actually reads in the line
```

Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai



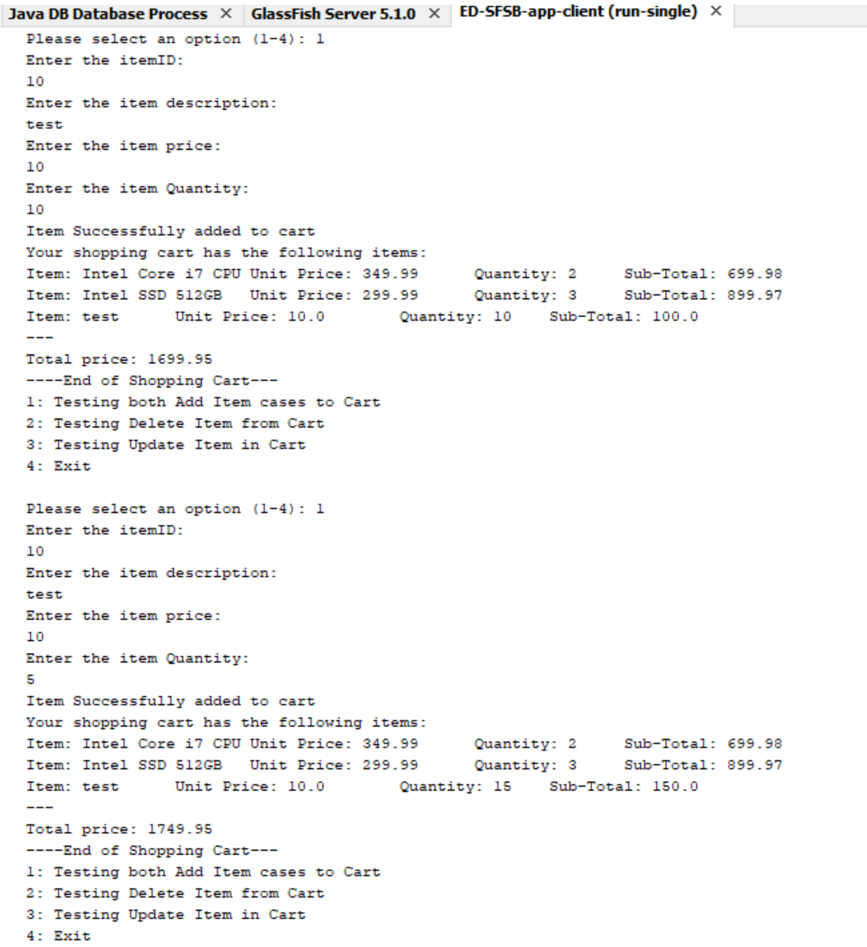
```
                case 2:
                    System.out.println("ItemID:");
                    Scanner itemIdinput11 = new Scanner(System.in);
                    String itemId11 = itemIdinput11.nextLine();    //see if it actually reads in the line

                    boolean outcome2 = appClient.shopCart.deleteCartItem(itemId11);

                    if (outcome2 == true) {
                        System.out.println("Deleted Successfully");
                    } else {
                        System.out.println("Failed to delete");
                    }

                    appClient.displayCart();

                    break;
                case 3:
                    System.out.println("Enter the itemID:");
                    Scanner itemIdinput12 = new Scanner(System.in);
                    String itemId12 = itemIdinput12.nextLine();    //see if it actually reads in the line

                    System.out.println("Enter new item description:");
                    Scanner itemDescinput12 = new Scanner(System.in);
                    String itemDesc12 = itemDescinput12.nextLine();    //see if it actually reads in the line

                    System.out.println("Enter new item price:");
                    Scanner itemPriceinput12 = new Scanner(System.in);
                    double itemPrice12 = itemPriceinput12.nextDouble();    //see if it actually reads in the line

                    System.out.println("Enter new item Quantity:");
                    Scanner itemQuantityinput12 = new Scanner(System.in);
                    int itemQuantity12 = itemQuantityinput12.nextInt();    //see if it actually reads in the line

                    CartItem item12 = new CartItem(itemId12, itemDesc12, itemPrice12, itemQuantity12);

                    boolean outcome3 = appClient.shopCart.updateCartItem(item12);

                    if (outcome3 == true) {
                        System.out.println("Item Successfully updated");
```



```
                    System.out.println("Enter new item Quantity:");
                    Scanner itemQuantityinput12 = new Scanner(System.in);
                    int itemQuantity12 = itemQuantityinput12.nextInt();    //see if it actually reads in the line

                    CartItem item12 = new CartItem(itemId12, itemDesc12, itemPrice12, itemQuantity12);

                    boolean outcome3 = appClient.shopCart.updateCartItem(item12);

                    if (outcome3 == true) {
                        System.out.println("Item Successfully updated");

                    } else {
                        System.out.println("Item not updated");

                    }

                    appClient.displayCart();
                    break;

            }
        }//end while

    }

    private void addCart(CartItem item) {
        System.out.println("Adding item " + item.getDescription() + " to cart");
        if (shopCart.addCartItem(item)) {
            System.out.println("Your order of " + item.getQuantity()
                    + " " + item.getDescription() + " has been added.");
        } else {
            System.out.println("Sorry, your order of " + item.getQuantity() + " "
                    + item.getDescription() + " cannot be added due to low stock.");
        }
    }

    public void displayCart() {
        ArrayList<CartItem> ciList = shopCart.getCart();
        if (ciList.isEmpty()) {
            System.out.println("The shopping cart is empty!");
            return;
```

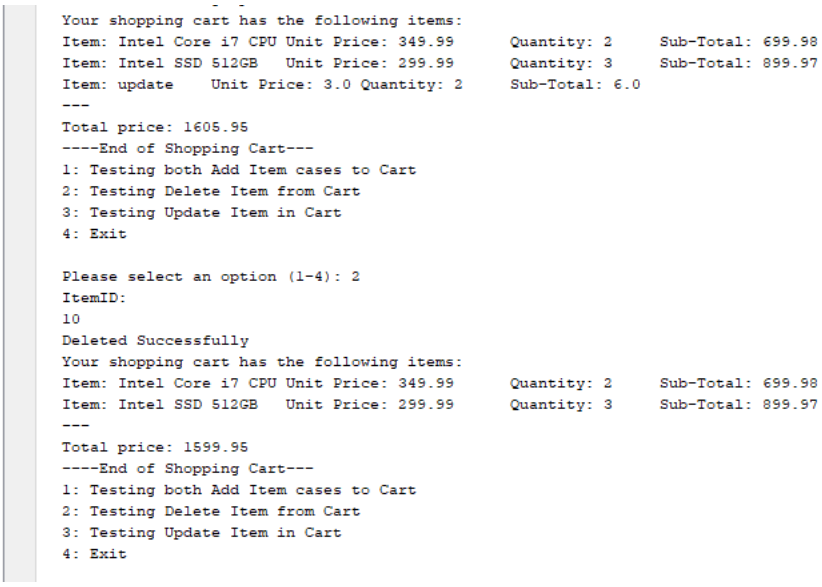
Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai
Task4:

Output for adding items (both new item and existing item)



```
Please select an option (1-4): 1
Enter the itemID:
10
Enter the item description:
test
Enter the item price:
10
Enter the item Quantity:
10
Item Successfully added to cart
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
Item: Intel SSD 512GB   Unit Price: 299.99      Quantity: 3     Sub-Total: 899.97
Item: test      Unit Price: 10.0        Quantity: 10    Sub-Total: 100.0
---
Total price: 1699.95
----End of Shopping Cart---
1: Testing both Add Item cases to Cart
2: Testing Delete Item from Cart
3: Testing Update Item in Cart
4: Exit

Please select an option (1-4): 1
Enter the itemID:
10
Enter the item description:
test
Enter the item price:
10
Enter the item Quantity:
5
Item Successfully added to cart
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
Item: Intel SSD 512GB   Unit Price: 299.99      Quantity: 3     Sub-Total: 899.97
Item: test      Unit Price: 10.0        Quantity: 15    Sub-Total: 150.0
---
Total price: 1749.95
----End of Shopping Cart---
1: Testing both Add Item cases to Cart
2: Testing Delete Item from Cart
3: Testing Update Item in Cart
4: Exit
```

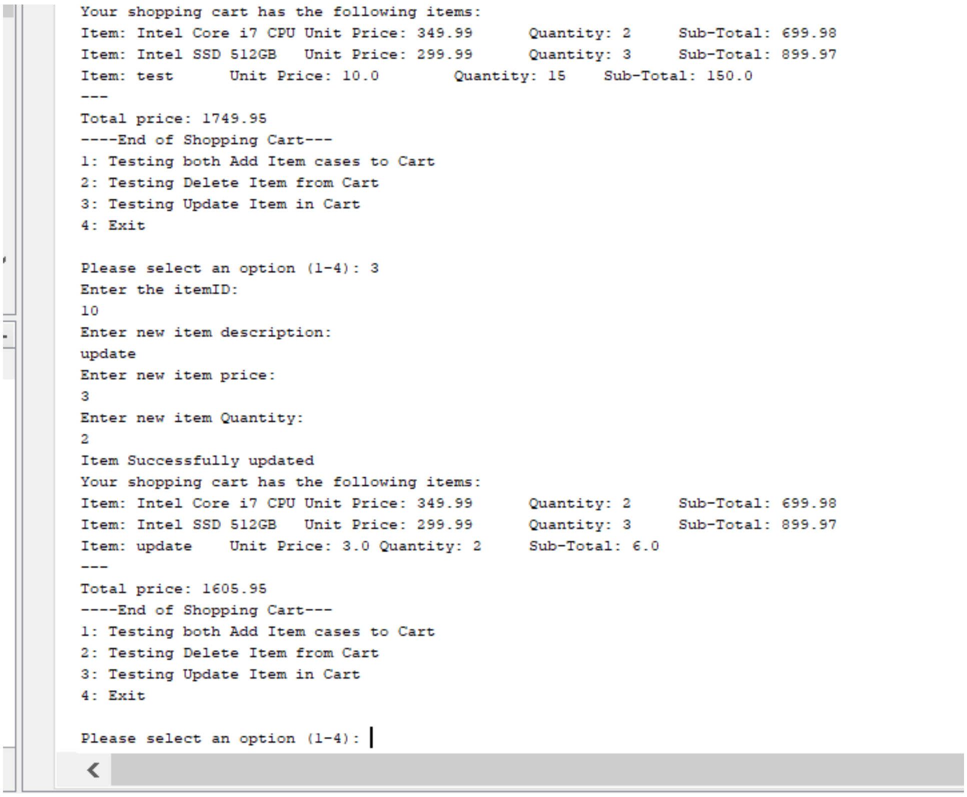
Output for deleting item:



```
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
Item: Intel SSD 512GB   Unit Price: 299.99      Quantity: 3     Sub-Total: 899.97
Item: update    Unit Price: 3.0 Quantity: 2     Sub-Total: 6.0
---
Total price: 1605.95
----End of Shopping Cart---
1: Testing both Add Item cases to Cart
2: Testing Delete Item from Cart
3: Testing Update Item in Cart
4: Exit

Please select an option (1-4): 2
ItemID:
10
Deleted Successfully
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
Item: Intel SSD 512GB   Unit Price: 299.99      Quantity: 3     Sub-Total: 899.97
---
Total price: 1599.95
----End of Shopping Cart---
1: Testing both Add Item cases to Cart
2: Testing Delete Item from Cart
3: Testing Update Item in Cart
4: Exit
```

Output for updating item:

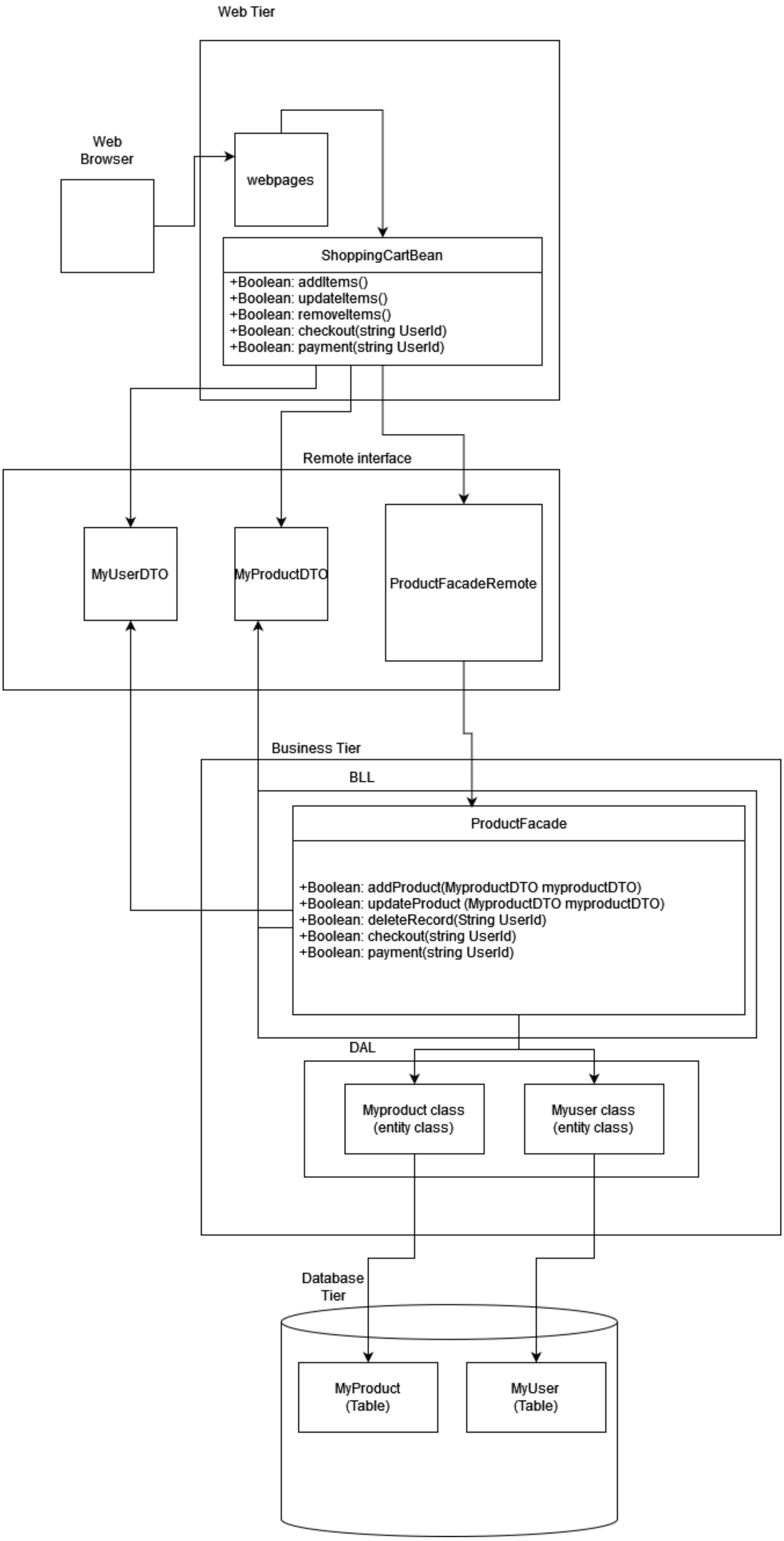
Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai



```
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
Item: Intel SSD 512GB   Unit Price: 299.99      Quantity: 3     Sub-Total: 899.97
Item: test      Unit Price: 10.0        Quantity: 15    Sub-Total: 150.0
---
Total price: 1749.95
----End of Shopping Cart---
1: Testing both Add Item cases to Cart
2: Testing Delete Item from Cart
3: Testing Update Item in Cart
4: Exit

Please select an option (1-4): 3
Enter the itemID:
10
Enter new item description:
update
Enter new item price:
3
Enter new item Quantity:
2
Item Successfully updated
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349.99      Quantity: 2     Sub-Total: 699.98
Item: Intel SSD 512GB   Unit Price: 299.99      Quantity: 3     Sub-Total: 899.97
Item: update    Unit Price: 3.0 Quantity: 2     Sub-Total: 6.0
---
Total price: 1605.95
----End of Shopping Cart---
1: Testing both Add Item cases to Cart
2: Testing Delete Item from Cart
3: Testing Update Item in Cart
4: Exit

Please select an option (1-4):
```

Task 5:

Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai


Web Tier
Web Browser
webpages
ShoppingCartBean
+Boolean: addItems()
+Boolean: updateItems()
+Boolean: removeItems()
+Boolean: checkout(string UserId)
+Boolean: payment(string UserId)
Remote interface
MyUserDTO
MyProductDTO
ProductFacadeRemote
Business Tier
BLL
ProductFacade
+Boolean: addProduct(MyproductDTO myproductDTO)
+Boolean: updateProduct (MyproductDTO myproductDTO)
+Boolean: deleteRecord(String UserId)
+Boolean: checkout(string UserId)
+Boolean: payment(string UserId)
DAL
Myproduct class (entity class)
Myuser class (entity class)
Database Tier
MyProduct (Table)
MyUser (Table)

Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai
Webpage: It is the UI (basically whatever the user sees)

ShoppingCartBean: This is the stateful managed bean. It basically keeps track of users and contains all the methods needed for the business logic to function

MyuserDTO: It is a DTO class that I would task to keep track of userID so that when customer comes again (assuming after certain period of time by which idle time has been triggered and browser has locked the user out), they would just input their user id and then the myuserDTO will pass it to the relevant objects (in BLL) to restore the user's session (in UI) and bring him back to his previous page.

MyproductDTO: It is a DTO class that will keep track of all data transfer from web UI to BLL

ProductFacadeRemote: it is a remote interface class. It is used to call the ProductFacade class from ShoppingCartBean

ProductFacade: It is a Facade Class. It serves as an interface to simplify the codes and contains most of the basic logic being used in the ShoppingCartBean (except the basic SQL ones)

Myproduct class: It is an entity class. It contains the most basic knowledge needed to transfer data to and from the MyProduct table in Database

Myuser class: It is an entity class. It contains the most basic knowledge needed to transfer data to and from the MyUser table in Database

MyProduct: It is a database table that will store the product details for business purposes

MyUser: It is a database table that will store the user details for session purposes