Pass Task 5.2 Stateful business logic

Design justification:

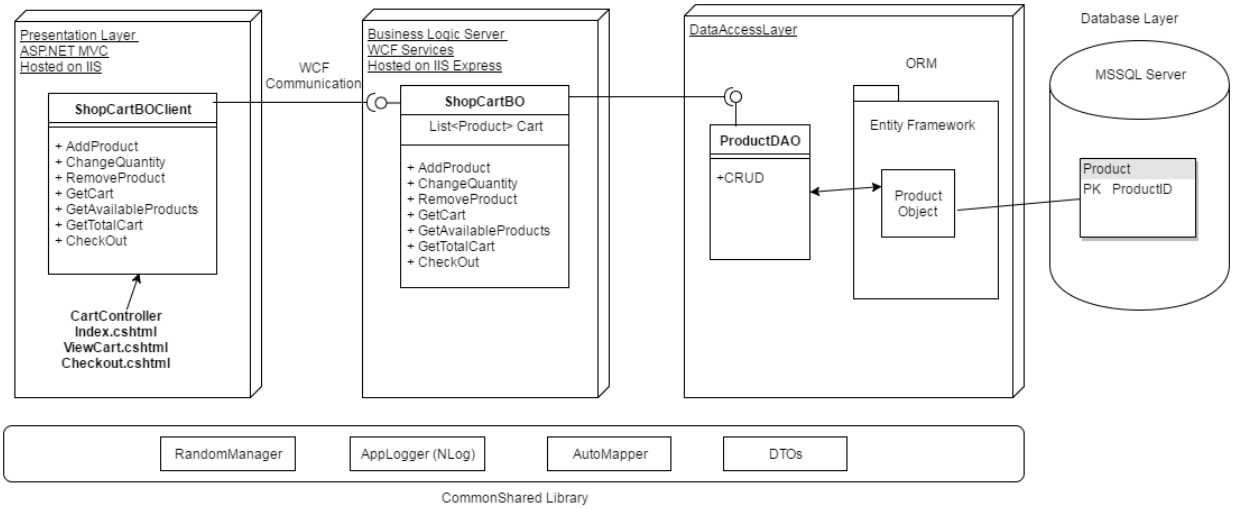


Figure 1. Architecture diagram.

Database Layer, DAL: This layer is similar to the design on PassTask 4.2: A MSSQL Server database, effectively SQL Express server which contain the data table Product. DAL contains one DAO object: ProductDao that implemented using Entity Framework to communicate with the Product data table.

Business Logic Server: This layer is similar to the design on PassTask 4.2: implemented using WCF Services. To handle the cart logic, it uses the ShopCartBO with an instant variable *Cart.* This service expose an interface IShopCartBO that can be access by the upper layer. This layer will handle tasks related to the business rules on customer cart such as: add product, change quantity of products on cart, remove item from cart, list all products from cart, calculate cart total, get all available products and cart checkout. The Business Logic Service is deployed to an IIS hosting service with a specific port number and URL address. In order to make this service stateful, SessionMode have to be enabled for this ServiceContract by decorating the IShopCartBO with option: SessionMode.Required; the ShopCartBO must also be decorated with a ServiceBehavior configuration to specify that the life span of this object will be per session.

```
[ServiceContract(SessionMode = SessionMode.Required)]
public interface IShopCartBO
{……}

[ServiceBehavior(InstanceContextMode = InstanceContextMode.PerSession)]
public class ShopCartBO : IShopCartBO
{……}
```

Presentation Layer: This layer use a proxy class ShopCartBOClient to handle the communication with the equivalent business object ShopCartBO in the Business Logic Server. This class is generated using the svcutil.exe tool provided by the .Net Framework. The presentation provides 3 web pages:

1. Index.cshtml : Listing of all available products so that customer can add an item to cart
2. ViewCart.cshtml : Show all the products currently on the cart and cart total.
3. Checkout.cshtml : Process cart checkout.

Common Shared Library: This is a shared library between all layers in this architecture which contains DTO, helper classes, logging tools that can be used by all layers.

Design justification:
In this design, we use a stateful business logic to implement the logic related to cart. The benefit of this design is that, the Cart instance will be maintained during the course of the whole transaction. This cart will only be cleared after user checkout and an order is created for it. Another benefit is that when user abandon the shopping cart. The BO object will be destroyed when the session is closed and the Cart instance will automatically be destroyed without any extra code. An equivalent implementation could be using a stateless business object with a Cart table to store the Cart information for each user. This method would introduce more overhead and complexity to the program such as: maintaining a session id for each user so that the BO know which cart to pull out from the database, clean up the cart information in the database if the cart is abandoned, identify when the cart can be marked as abandoned. It is clear that stateful business object would be a reasonable choice to implement this behavior.

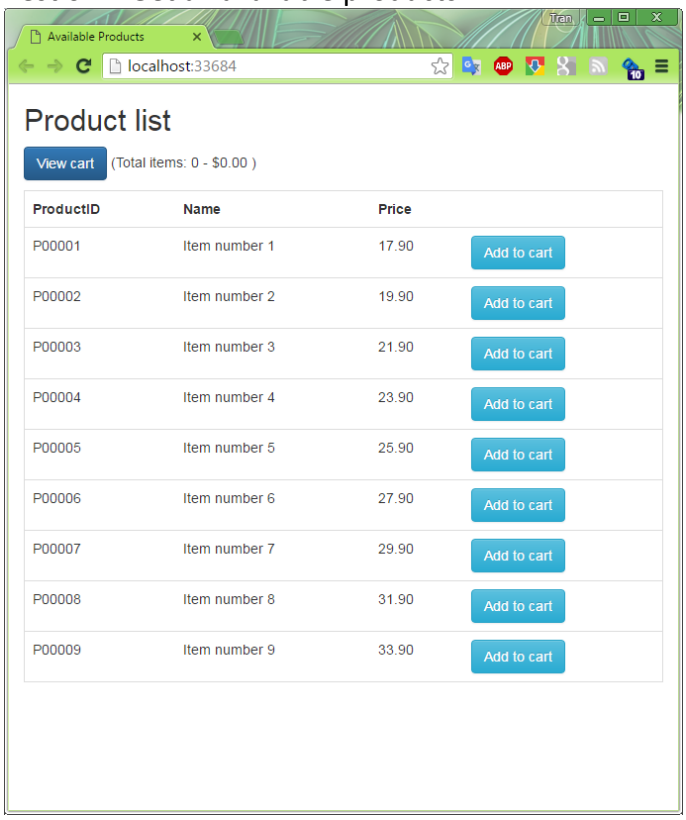
Test cases:
The following test cases have been taken out to make sure that the whole application works as expected:

ID	Test Case Name	Step	Expected Output	Result
01	Get all products	View the product pages	A list of products available is displayed on the screen	Pass
02	Add a product to cart	Click to the Add to cart link next to a product in the product list.	The cart total is changed reflecting new item is added to cart.	Pass
03	Add another product to cart	With one product currently on the cart, add another product to the cart	The cart total is calculated correctly for all items in the cart.	Pass
04	View cart list	Click to View cart link	All products on the cart should be displayed with a cart total information	Pass
05	Change product quantity on cart	Change quantity of a product on the cart and click Update Cart button	The quantity of that product is updated and the cart total is re-calculated	Pass
06	Checkout cart	Click to the Checkout button	A success message will display The cart is cleared as order is created for that cart.	Pass

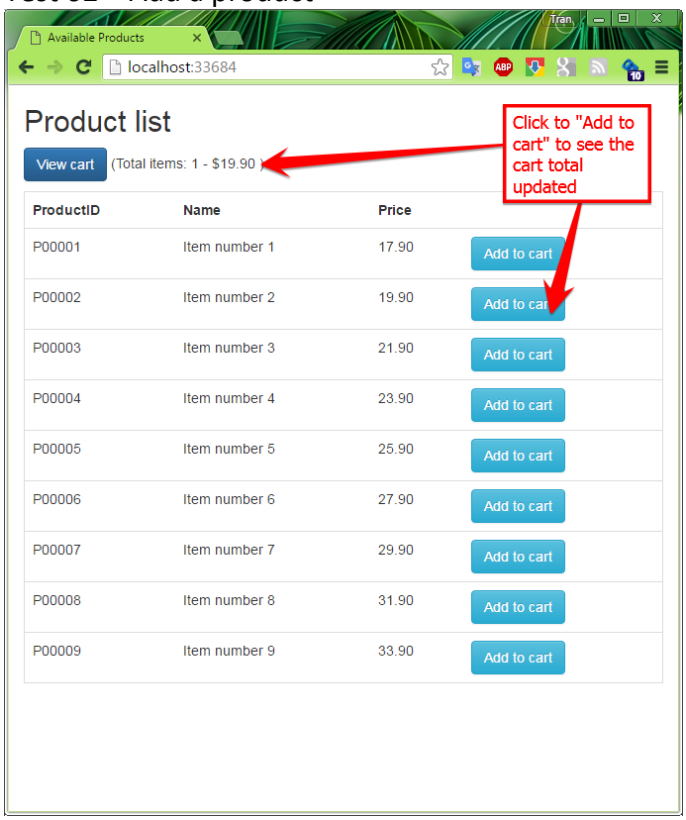
Table 1. Test cases and test results

Test case screen shots:

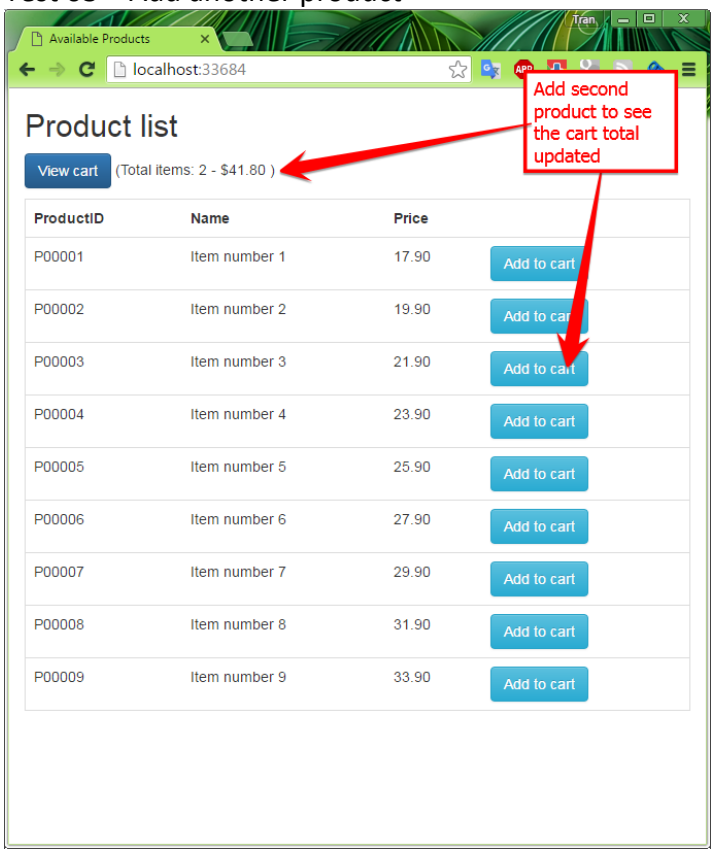
Test 01 – Get all available products



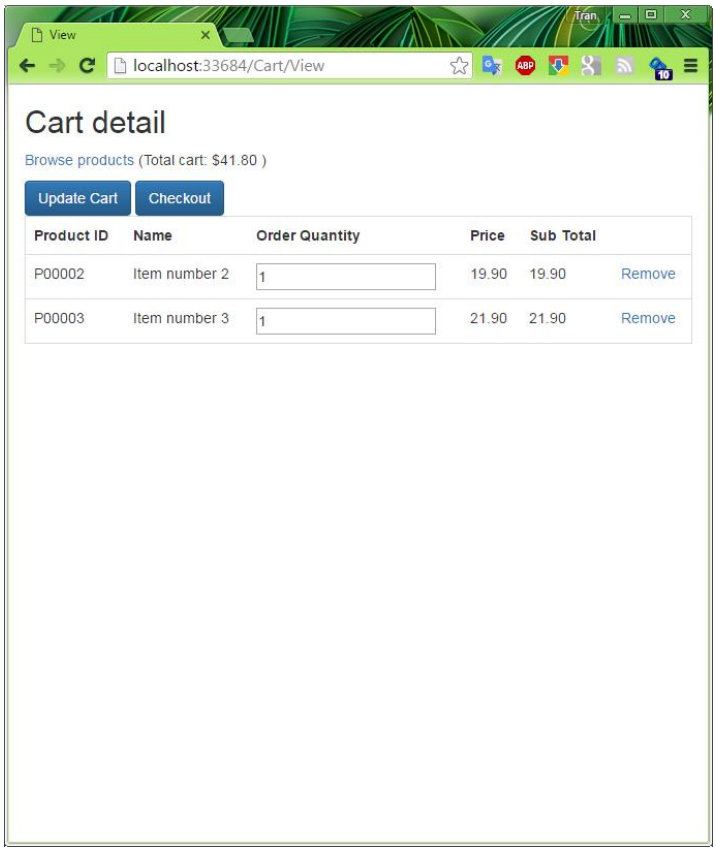
Test 02 – Add a product



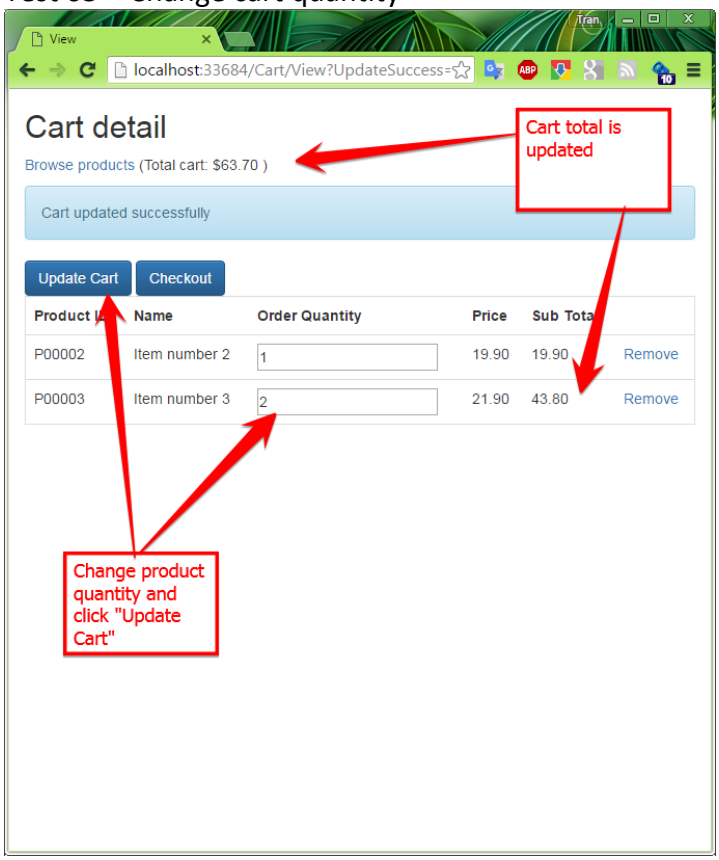
Test 03 – Add another product



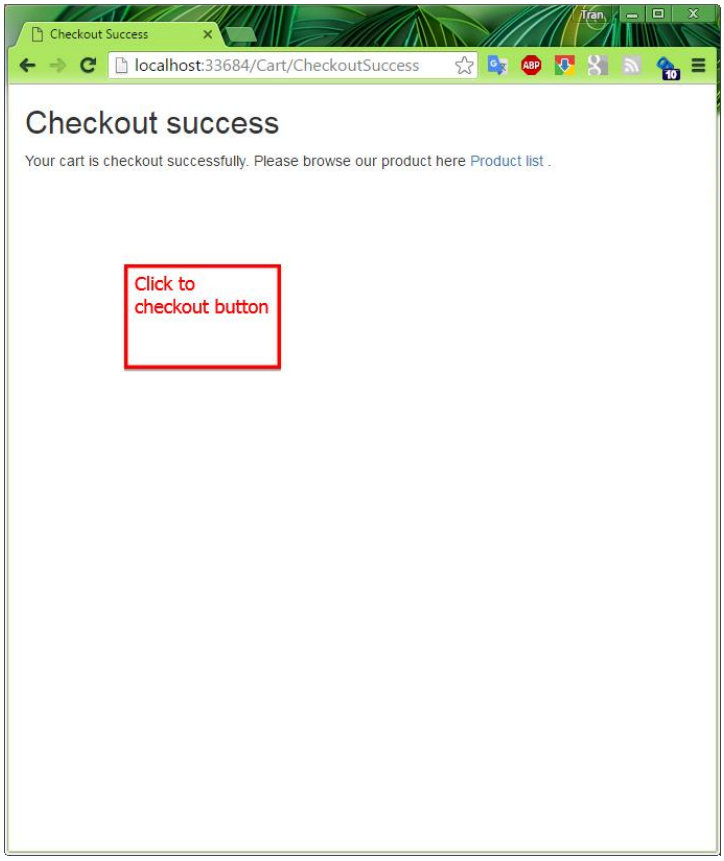
Test 04 – View cart



Test 05 – Change cart quantity



Test 06 – Checkout



References:

(1) Green, R 2016, *Sessions, Instancing and Concurrency in WCF Services*, Microsoft MSDN, viewed 10th May 2016, < https://msdn.microsoft.com/en-us/library/ff183865.aspx>

(2) BitBucket source code: https://bitbucket.org/werynguyen/swinschool/src/68cb7f9a84a508ef1263b72d2743ac9136c9e395/?at=PT52