Practical 6: Implement 3-Tier Web Application (Part 3)

Objectives

- Consume web services and display product to web form from web service
- Complete purchase order to Supplier

From B2C to B2B using Web Service

Scenario

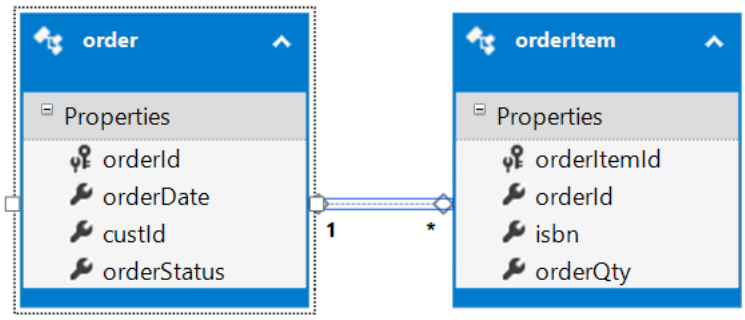
Best Book Supplier (BBS) extend their web service to provide an operation that allows the bookshop to place their inventory ordering. In this practical, you will create a new operation contract named **CreateOrders**. It will take the order items of the bookshop and create the record in the order table.

For this exercise, we have another customer of BBS named ABCHealth. ABC Health in an online shopping company. ABC Health wishes to implements a cost effective inventory management **which will automatically replenish their stock when inventory in running low**. Knowing that BBS providing web service that support online stock-ordering, they will like to extend their online

Exercise 1 : Product reordering Web Service for retailer

In this exercise, you will implement a new CreateOrder operation for BBS. Where BBS receives the customer order, stores it into order table and order item table.

1. Download IT2605Prac05_BBS resource to open IT2605Prac05_BBS.sln.
2. Open the database, you will find two new tables.



Order table contains the data about the order id, the date of order, customer id of the order.

Order item table contains the order item and the order quantity of each order id

3. Open DAL and you will find two new class **Order.cs** and DalOrder.cs
4. The Order object composes of book ISBN and the order quantity of the particular order item.
5. Open DalOrders.cs from DAL folder of IT2605Prac05_BBS. Read and understand the code.ly
6. DalOrder class contains CreateOrder method
    a. The method requires two input arguments
        i. Customer id
        ii. Order items
    b. The method inserts the ordering data into Order table and order item table
    c. It returns order id to the customer

Figure 1

```
public class DalOrders
{
    private String errMsg;
    DalDbConn dbConn = new DalDbConn();

    public int CreateOrder(String customerid,   List<Order> obj )
    {
        StringBuilder sql;
        SqlCommand sqlCmd;
        int result = 0;
        int newOrderId = 0;
        // create order header
        SqlConnection conn = dbConn.GetConnection();
        sql = new StringBuilder();
        sql.AppendLine("insert into orders (orderDate,custId,orderStatus)");
        sql.AppendLine("VALUES (@paraordate, @paraordcust, @paraordstatus)");
        sql.AppendLine("SELECT CAST(scope_identity() AS int)");
        sql.AppendLine(" ");
        sqlCmd = new SqlCommand(sql.ToString(), conn);
        sqlCmd.Parameters.AddWithValue("@paraordate", DateTime.Now);
        sqlCmd.Parameters.AddWithValue("@paraordcust", customerid);
        sqlCmd.Parameters.AddWithValue("@paraordstatus", "PO");
       try
       {
            conn.Open();
            newOrderId = (Int32)sqlCmd.ExecuteScalar();


            StringBuilder sqlItem ;

            foreach (Order objitem in obj)
            {
                sqlItem = new StringBuilder();
                sqlItem.AppendLine("insert into orderItem (orderId, isbn, orderQty)");
                sqlItem.AppendLine("VALUES (@paraordId, @paraisbn, @paraordqty)");
                sqlItem.AppendLine(" ");
                sqlCmd = new SqlCommand(sqlItem.ToString(), conn);
                sqlCmd.Parameters.AddWithValue("@paraordId", newOrderId);
                sqlCmd.Parameters.AddWithValue("@paraisbn", objitem.ordBookisbn);
                sqlCmd.Parameters.AddWithValue("@paraordqty", objitem.ordBookQty);
                result = sqlCmd.ExecuteNonQuery();

            }
        }
        catch (Exception ex)
        {
            errMsg = ex.Message;
        }
        finally
        {
            conn.Close();
        }
```

The code in the first rectangle, insert an order record to order table with customer id, today date and order status. An auto number is assigned to order id. The SELECT CAST(scope_identity) return the new order id to be used as reference key for the subsequent insert query for order item table.

The second rectangle, iterate the list of orders and insert each order item into order item table.

7. Right click **BLL** folder in solution explorer to create a new class file named **ProcessOrders.cs.**
   This business layer invokes CreateOrder in DalOrder to perform the order insertion in BBS order and order item tables.

   - Add the using statements

```
using System.Collections.Generic;
using IT2605Prac05_BBS.DAL;
```

   - Create **VerifyOrders** method. It invokes CreateOrder in DalOrder to perform the order insertion in BBS order and order item tables.

Figure 2

```
using System;
using System.Collections.Generic;
using System.Collections;
using System.Linq;
using System.Web;
using IT2605Prac05_BBS.DLL;

namespace IT2605Prac05_BBS.BLL
{
    public class ProcessOrders
    {
       public int VerifyOrders(String customerid, List<Order> obj)

        {
            DalOrders dataLayerOrders = new DalOrders();

            return dataLayerOrders.CreateOrder(customerid, obj);
        }
    }
}
```

6. Open WsBookCatalog.svc.cs, add a method CreateOrders .

Figure 3

```
public int CreateOrders(String customerid, List<Order> eachorder)
{

     IT2605Prac05_BBS.BLL.ProcessOrders bizLayerOrder ;
     bizLayerOrder = new IT2605Prac05_BBS.BLL.ProcessOrders();
     return bizLayerOrder.VerifyOrders(customerid, eachorder);

 }
```

8. Open **IWsBookCatalog.cs** in IT2605Prac05_BBS.sln
   - Add the using statements

```
using System.Collections.Generic;
using IT2605Prac05_BBS.DAL;
```

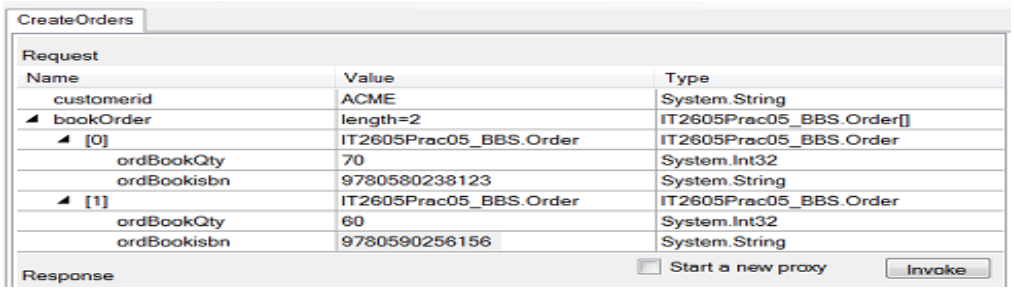
   - **Declare a new operation contract for CreateOrders.**

```
[OperationContract]
int CreateOrders(String customerid, List<Order> bookOrder);
```

   The operation requires two input arguments:
   -Customer id of its retail shop and
   -List of books ordered by the retail shop.

7. You have completed the CreateOrder operation. Test the result.
   - Enter any value for customerid. Ex ACME
   - Prepare the test data for ISBN from the books table
   - Enter any order quantity.
   - Verify that order and order item are inserted in order and orderitem table.

Figure 4



Exercise 2 : Consume the supplier web service to complete B2C to B2B

ABCHealth provides an online shopping cart. For every customer order, the system will check the remaining stock level, if it is below the reordering point, system will automatically place the purchase order to BBS supplier.

1. Download and unzip ABCHealth - ShoppingCart folder.
2. Open ABCHealth project solution.
3. Study the database of ABCHealth . You will implement the code to derive the remaining stock level by subtracting the existing stock level from customer order quantity. Check the reordering status of the product, if it is null and the remaining stock is less than reorder point, invoke BBS CreateOrder operation to order the stock using reordering quantity.
4. Right click service reference, copy the BBS web service url to address in Add Service Reference dialog and named the namespace as bbs_svc.
5. Open **Product.cs** in *DataAccessLayer*. Create a **updateProductRO** method as in Figure 5

Figure 5

```
public void updateProductRO(decimal prodID, int cartqty)
{
    String queryStr = "Update Product Set ReorderStatus='Pending DO' ";
    queryStr += ", Stock_Level = Stock_Level - @cartqty ";
    queryStr += "Where Product_ID = @ProdID";
     SqlConnection conn = new SqlConnection(_connStr);
     SqlCommand cmd = new SqlCommand(queryStr, conn);
     cmd.Parameters.AddWithValue("@ProdID", prodID);
     cmd.Parameters.AddWithValue("@cartqty", cartqty);
     conn.Open();
     cmd.ExecuteNonQuery();
}
```

6. Open **ProductBLL** in *BusinessLogicLayer.* Create updateProdRO method as in Figure 6.

Figure 6

```
public void updateProdRO(decimal prodID, int cartQty){
     Product prodDetail = new Product();
     prodDetail.updateProductRO(prodID, cartQty);

 }
```

7. Open **ShoppingCart.cs** in *ShoppingCart folder*.

Create a new method **CheckStock()** . This method will access each cart item against the product table. If the reordering status is null, (ie system has not reorder the stock yet) , and the remaining stock level is below the reordering point, then invoke CreateOrder() service to replenish stocks.

Figure 7

```
public void CheckStock()
 {
     List<Order> PurchaseOrder = new List<Order>();
     foreach (ShoppingCartItem item in Items)
     {
         ABCHealth.bbs_svc.Order cusOrd = new ABCHealth.bbs_svc.Order();

         ProductBLL prodBLL = new ProductBLL();

         // Create a product object
         Product prod = null;

         // Call getProdDetail() method
         prod = prodBLL.getProdDetail(item.ItemID);

         if ((prod.Stock_Level - item.Quantity < prod.Reorder_Point) &&
             (prod.Reorder_Status.Equals(String.Empty)))
         {
             cusOrd.ordBookisbn = prod.Product_ISBN;
             cusOrd.ordBookQty = prod.Reorder_Qty;
             // continue to add PO item
             PurchaseOrder.Add(cusOrd);
             prodBLL.updateProdRO(item.ItemID, item.Quantity);
         }
     }
     if (PurchaseOrder.Count > 0)
     {
         WsBookCatalogClient ws = new WsBookCatalogClient();
         Order[] PO = PurchaseOrder.ToArray();
         int result = ws.CreateOrders("ABCH", PO);
     }
```

Retrieve reordering information and stock level of the product

If reordering is required , formulate the reordering item for purchase order.

Invoke BBS service to send ABC Health Id and list of PO to supplier.

8. Open **ViewCart.aspx**, double click the order button . In the order button click method, you will retrieve the shopping cart instance, and then activate the action to call the checkStock () created earlier.

   Note that the practical only focus on completing B2B process, in the real scenario, the system should create the ordering information in the company database.

```
protected void btn_Order_Click(object sender, EventArgs e)
{
    if (ShoppingCart.ShoppingCart.Instance.Items.Count > 0)
    {
        // check the product inventory
        ShoppingCart.ShoppingCart.Instance.CheckStock();
    }
        lb_error.Text = "Thank you for your patronised! ";
}
```

9. Click run or press F5 to test the result.
   - Complete the ordering cart.
   - To activate the reordering, you need to input the quantity which will cause the stock level falls below reordering point.
   - Verify the reordering from the supplier database.

~~End ~~