Cognizant Academy

truYum

ADO.NET Specification Document

Version 1.0

	Prepared By / Last Updated By	Reviewed By	Approved By
Name	Ramamoorthy Selvam	Vimalathithan Krishnan	Ramadevanahalli Lingachar, Shashidhara Murthy
Role	Learning Solution Designer	Learning Solution Architect	Learning Solution Lead
Signature			
Date			

Table of Contents

1.0 Introduction

1.1 Purpose of this document

The purpose of this document is to define the ADO.Net module implementation for truYum project.

1.2 Definitions & Acronyms

Definition / Acronym	Description
DAO	Data Access Object
ADO.NET	Activex Data Objects

1.3 Project Overview

Refer truYum-use-case-specification.docx to understand the functionality and features.

1.4 Scope

1. Creation of DAO classes and methods for reading and persisting data of truYum application.

1.5 Intended Audience

- Product Owner
- Scrum Master
- Application Architect
- Project Manager
- Test Manager
- Development Team
- Testing Team

1.6 Hardware and Software Requirement

1. Hardware Requirement:
    a. Developer PC with 8GB RAM
2. Software Requirement

a. IE or Chrome

b. .Net Framework 4.5

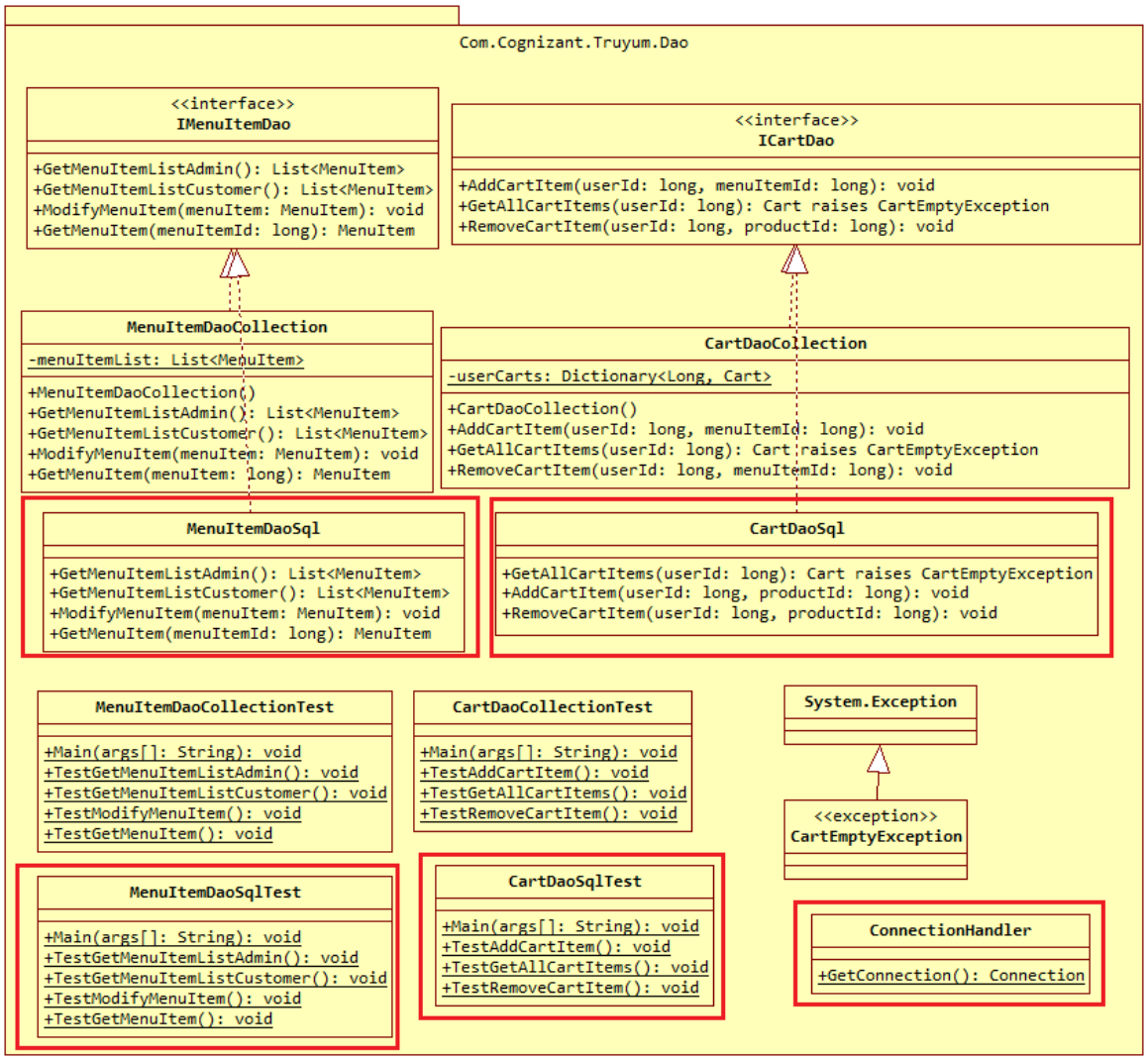
c. Visual Studio Professional Edition 2015

d. SQL Server enterprise edition 2014

2.0 Class Diagram

2.1 Data Access Layer

Refer the diagram below and create classes accordingly.



Dotted arrow represents implementation of an interface.

Make note that GetConnection is a static method.

ADO.Net class library needs to be referenced in the Class Library project created in the Console application as per the CSharp specification. The data fetched from database thru ADO.Net should be displayed in the console window of the TruyumConsole application.

Highlighted classes are the ones that needs to be implemented in this specification.

2.2 Helper.cs

This class will be used by each Dao implementation class for getting the database connection.

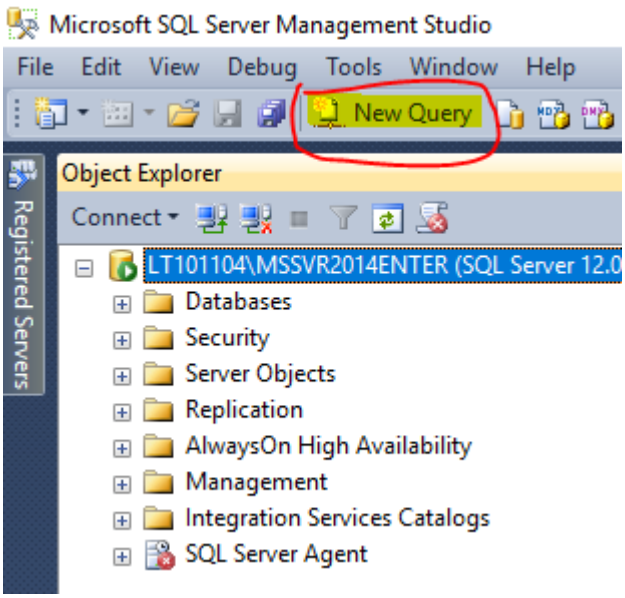
The connection details has to be stored in a web.config file. Find the details below:

```
<connectionStrings>
       <add name="connectionString"
       connectionString="Data Source=localhost;Initial Catalog=truYum;
       Integrated Security=True" providerName="System.Data.SqlClient" />
</connectionStrings>
```

A static class with a property `ConnectionString` that gets connection string from the web.config file.

Gets connection using `ConfigurationManager` Class by referring the name attribute in <connectionStrings> element present in Web.Config file

2.3 Base data creation

1. Open SQL Server Management Studio.
2. Click on the 'New Query' option
3. 

4. Execute the script available here
5. The scripts creates the base database with minimal data.

3.0 DAO for View Menu Item List Admin

(TYUC001)

3.1 MenuItemDaoSql.cs

GetMenuItemListAdmin() of return type List<MenuItem>

1. Get connection using `SqlConnection` object
2. Initialize a `List` of `MenuItem`
3. Using `SqlCommand` execute the select query that retrieves all the records from menu_item table
4. Iterate through the `ResultSet`
5. For each item in the `ResultSet` create a new `MenuItem` instance and add it to the `List` created in the step 2 and return the `List`

4.0 DAO for View Menu Item List Customer (TYUC002)

4.1 MenuItemDaoSql.cs

GetMenuItemListCustomer() of return type List<MenuItem>

1. Get connection using `SqlConnection` object
2. Initialize a `List` of `MenuItem`
3. Using `SqlCommand` execute the select query that retrieves the records from menu_item table applying the following filters:
    a. The menu item is in stock and
    b. The menu item is not past the expiry date
4. Iterate through the `ResultSet`
5. For each item in the `ResultSet` create a new `MenuItem` instance and add it to the `List` created in the step 2 and return the `List`

5.0 DAO for Edit Menu Item (TYUC003)

5.1 MenuItemDaoSql.cs

GetMenuItem(long menuItemId) of return type MenuItem

1. Get connection using `SqlConnection` object
2. Execute select query using `SqlCommand` that retrieves an item from `menuItem` table based on `menuItemId`.
3. Create a `MenuItem` instance and set the values for this menuItem instance from the first item of the `ResultSet`
4. Return the `menuItem` created in the previous step

EditMenuItem(MenuItem menuItem)

1. Get connection using `SqlConnection` object
2. Execute update statement using `SqlCommand` that modifies the values of menuItem table based on `menuItemId`.
3. Set the parameters of the `SqlCommand` and execute the statement.

6.0 DAO for Add a Menu Item to Cart (TYUC004)

6.1 CartDaoSql.cs

AddMenuItem(long userId, long menuItemId) of return type void

1. Get connection using `SqlConnection` object
2. Execute insert statement using `SqlCommand` for inserting data into `cart` table with `userId` and `menuItemId`.

7.0 DAO for View Cart (TYUC005)

7.1 CartDaoSql.cs

GetMenuItems(long userId) of return type List<MenuItem>

1. Get connection using `SqlConnection` object
2. Initialize a `List` of MenuItem
3. Execute update statement using `SqlCommand` that joins `Cart` and `MenuItem` table to retrieve the list of menu items associated with a specific user.
4. Iterate through the `ResultSet`
5. For each item in the `ResultSet` create a new `MenuItem` instance and add it to the `List` created in the step 2 and return the `List`

8.0 DAO for Remove Item from Cart (TYUC006)

8.1 CartDaoSql.cs

RemoveMenuItem(long userId, long menuItemId) of return type void

1. Get connection using `SqlConnection` object
2. Execute delete statement using `SqlCommand` for delete data into cart table based on `userId` and `menuItemId`.

9.0 Standards and Guidelines

9.1 DAO

1. All .Net coding standards are applicable
2. Closure of connection should be done within finally block or thru Using statement

10.0 Change Log

	Changes Made			
V1.0.0	Initial baseline created on 20-May-19 by Ramamoorthy Selvam			
Vx.y.z	<Please refer the configuration control tool / change item status form if the details of changes are maintained separately. If not, the template given below needs to be followed>			
	Section No.	**Changed By**	**Effective Date**	**Changes Effected**