Cognizant Academy

truYum

ASP.NET MVC Core Specification Document

Version 0.1

	Prepared By / Last Updated By	Reviewed By	Approved By
Name	Seshadri M R	Seshadri M R	Seshadri M R
Role	Learning Solution Designer	Learning Solution Designer	Learning Solution Designer
Signature			
Date			

Table of Contents

1.0 Introduction

1.1 Purpose of this document

The purpose of this document is to define the server side implementation of the truYum application.

1.2 Definitions & Acronyms

Definition / Acronym	Description
ASP.NET MVC Core	ASP.Net MVC Core is a Web development framework built on top of ASP.Net MVC. It is built on the request pipeline serving concept. Dependency injection is the core concept. Webhost builder is used to create the required builder to build the web application. This application can be deployed onto Linux environment too.

1.3 Project Overview

Refer Use Case specification document for understanding the functionality and features.

1.4 Scope

1. Creation of ASP.Net MVC Core web application for truYum application

1.5 Hardware and Software Requirment

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 2017
   d. SQL Server enterprise edition 2014

2.0 Design for View Menu Item List Admin (TYUC001)

2.1 Requirement flow

Steps Explanation

1. Application user launches the application. There should be a link to View the Menu items as the Admin of the application.

2. Create an Empty controller MenuItemsController, to contain Actions related to Menu items.

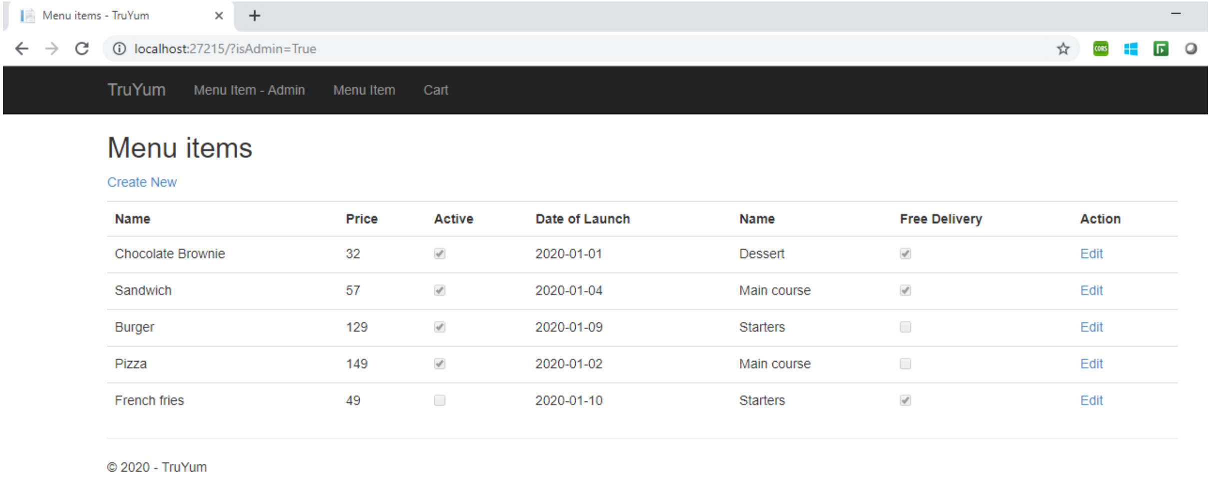
3. Use Entity Framework to get the data of the relevant entities and display it in the View.

4. Admin user would be able to view all the menu items.

5. On any exception, user should be directed to "**Error**" view with a message `Exception occurred in [Action] method of [controller].` Along with this, please append the exception message.

2.2 Menu items – Admin

List the menu items for admin.



Menu items

Create New

Name	Price	Active	Date of Launch	Name	Free Delivery	Action
Chocolate Brownie	32	☑	2020-01-01	Dessert	☑	Edit
Sandwich	57	☑	2020-01-04	Main course	☑	Edit
Burger	129	☑	2020-01-09	Starters	□	Edit
Pizza	149	☑	2020-01-02	Main course	□	Edit
French fries	49	□	2020-01-10	Starters	☑	Edit

© 2020 - TruYum

2.3 Technical guidelines

- Create class for Category with Id and Name data fields
  - Use data annotation **Key** for the Id field
- Create the MenuItem model with the required fields and data annotations

- Use data annotation **Key** for the Id field

- Use data annotation **Required** for the Name & Price fields. This data annotation will show the default message on the UI while creating a MenuItem.

- Use data annotation **Display** with name attribute for the fields, FreeDelivery and DateOfLaunch. This will ensure that the detail listed on the UI will be appropriate.

- Use the data annotation DataType as Date for DateOfLaunch. This will help to display the date field in the format that is expected.

  ```
  [DataType(DataType.Date)]
  [DisplayFormat(DataFormatString       =       "{0:yyyy-MM-dd}",
  ApplyFormatInEditMode = true)]
  ```

  The **ApplyFormatInEditMode** being set to true is used to ensure that the date is filled in correct manner in the text box

- Reference the CategoryId and the Category class reference as Foreign key in the MenuItem class

- Create the Cart model with the required fields and data annotations. Create a reference to the MenuItem class to create foreignkey reference

  - Use data annotation **Key** for the Id field

  - Foreign key reference

- Create truYumContext class which inherits DbContext class. Include namespace "System.Data.Entity"

- Create a class `TruYumContext` in the Models folder. Inherit it from `DbContext` class of the namespace System.Data.Entity

  - Create a constructor of the truYumContext. Specify the name of the database connection string element to be "`TruYumContext`"

  - Create a connection string in appSettings.json to connect to the database with the connection string name "TruYumContext". Refer the link here to know how to access appSettings key.

  - Add DbSets for MenuItems, Carts and Categories in the TruYumContext class

  - Override the "OnModelCreating" method to remove the Pluralizing table names

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
modelBuilder.Conventions.Remove<PluralizingTableNameConvention>();
}
```

The "OnModelCreating" method override ensures that the table that gets created thru the CodeFirst approach has a singular name. On

- Enabling migration
- Add migration

        - Update database

    By default, there will be DbSet of MenuItems, Carts based on which the EntityFramework will create a table with the name “MenuItems” & “Carts”. The OnModelCreating override will create tables with singular name

    - On completion of “Update database” please insert master data in “Categories” table in the database. Master data should be
        - Main Course
        - Starters
        - Snack

- Modify the _Layout file to point to the Home page link. Notice the presence of Tag helpers. Use tag helpers to create the appropriate tags.

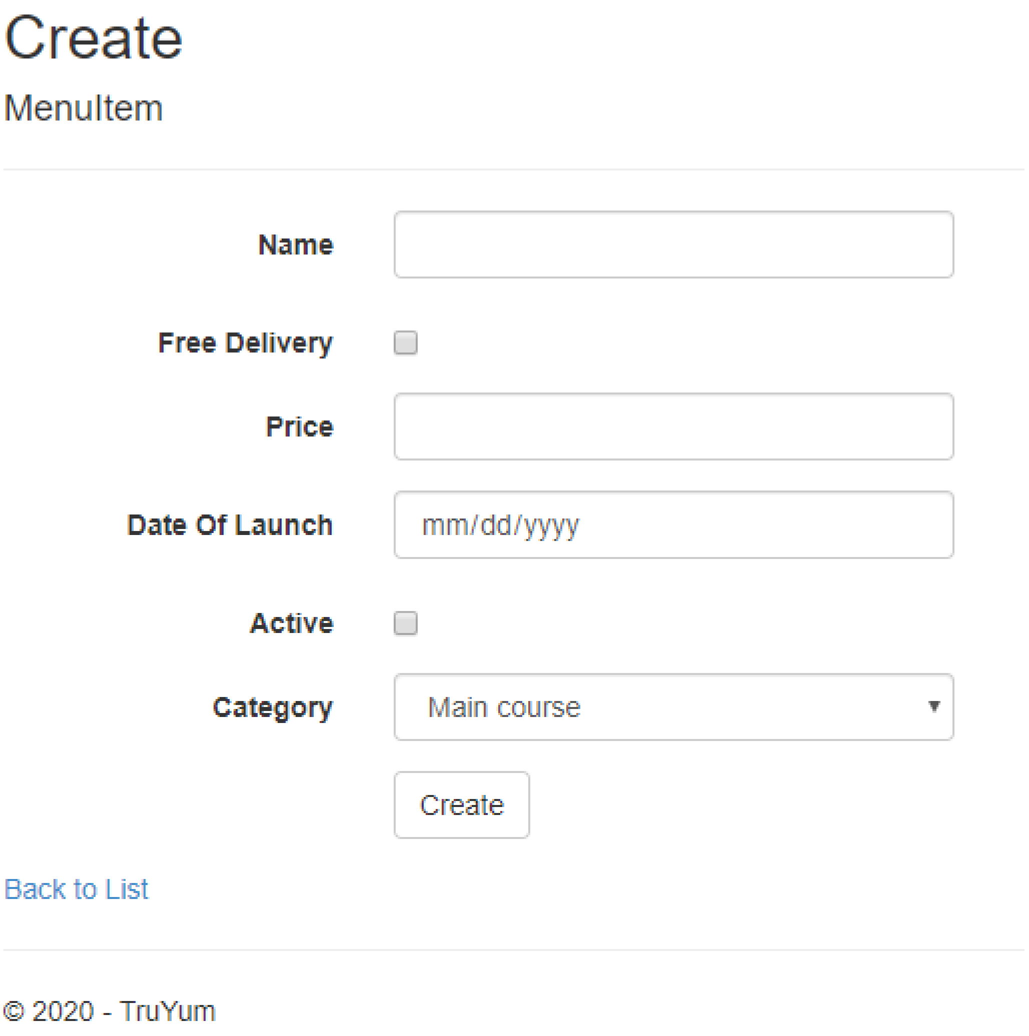
- Modify the RouteConfig for the Default route to point to the MenuItems controller

- Create Hyperlink for MenuItems listing for Admin and Customer. Use appropriate tag helpers
    - In the ActionLink for Menu Item for Admin “Menu Item – Admin”, send an object with a parameter “isAdmin” whose value should be True, to be output as querystring parameter. This will NOT be required for Customer role

- Create Hyperlink for Customer for Cart listing. Use appropriate tag helpers

- Create MenuItemsController thru EntityFramework scaffolding. Use the Index action method to list the Menu items
    - Create a boolean parameter **“isAdmin”** to the Index method and set the default value to false. This would be used to filter the menu item listing for Admin and Customer

- The “**Create New**” link in the menu item list page should be used to Create a new Menu Item by the admin user. Use appropriate tag helpers



Create

MenuItem

Name	
Free Delivery	□
Price	
Date Of Launch	mm/dd/yyyy
Active	□
Category	Main course

- Create a View “Error” which should be accessible from anywhere. This view should access error message from **TempData** to display on the UI

3.0 Design for View Menu Item List Customer (TYUC002)

3.1 Requirement flow

Steps Explanation

1. Application user launches the application. There should be a link to View the Menu items as the Customer of the application.

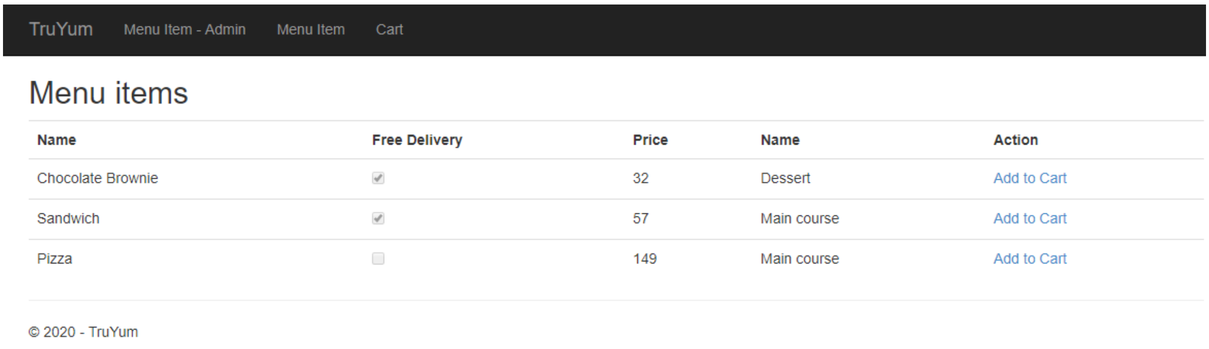
2. Customer user would be able to view the Active menu items and whose date of launch is in

the past compared to current date.

3. On any exception, user should be directed to “**Error**” view with a message `Exception occurred in [Action] method of [controller].` Along with this, please append the exception message.

3.2 Menu items – Customer

List the menu items for customer.



Menu items

Name	Free Delivery	Price	Name	Action
Chocolate Brownie	☑	32	Dessert	Add to Cart
Sandwich	☑	57	Main course	Add to Cart
Pizza	□	149	Main course	Add to Cart

© 2020 - TruYum

3.3 Technical guidelines

- The link added in _Layout is to load the menu items for the customer role user
- Modify the Index Action method of MenuItemsController to filter the Menu items as per the requirement
- Set the isAdmin value in ViewBag
- Modify the Index view to access the ViewBag to choose the columns to be listed
- Use the ViewBag data to toggle the **Action** column hyperlink

4.0 Design for Edit Menu Item (TYUC003)

4.1 Requirement flow

Steps Explanation

1. This feature is available for Admin user only
2. Admin user clicks on **Edit** link against every line item in the Menu items list. On click the edit page loads. Use appropriate tag helpers
3. The menu item Id should be sent to the Edit Action method of the MenuItemsController

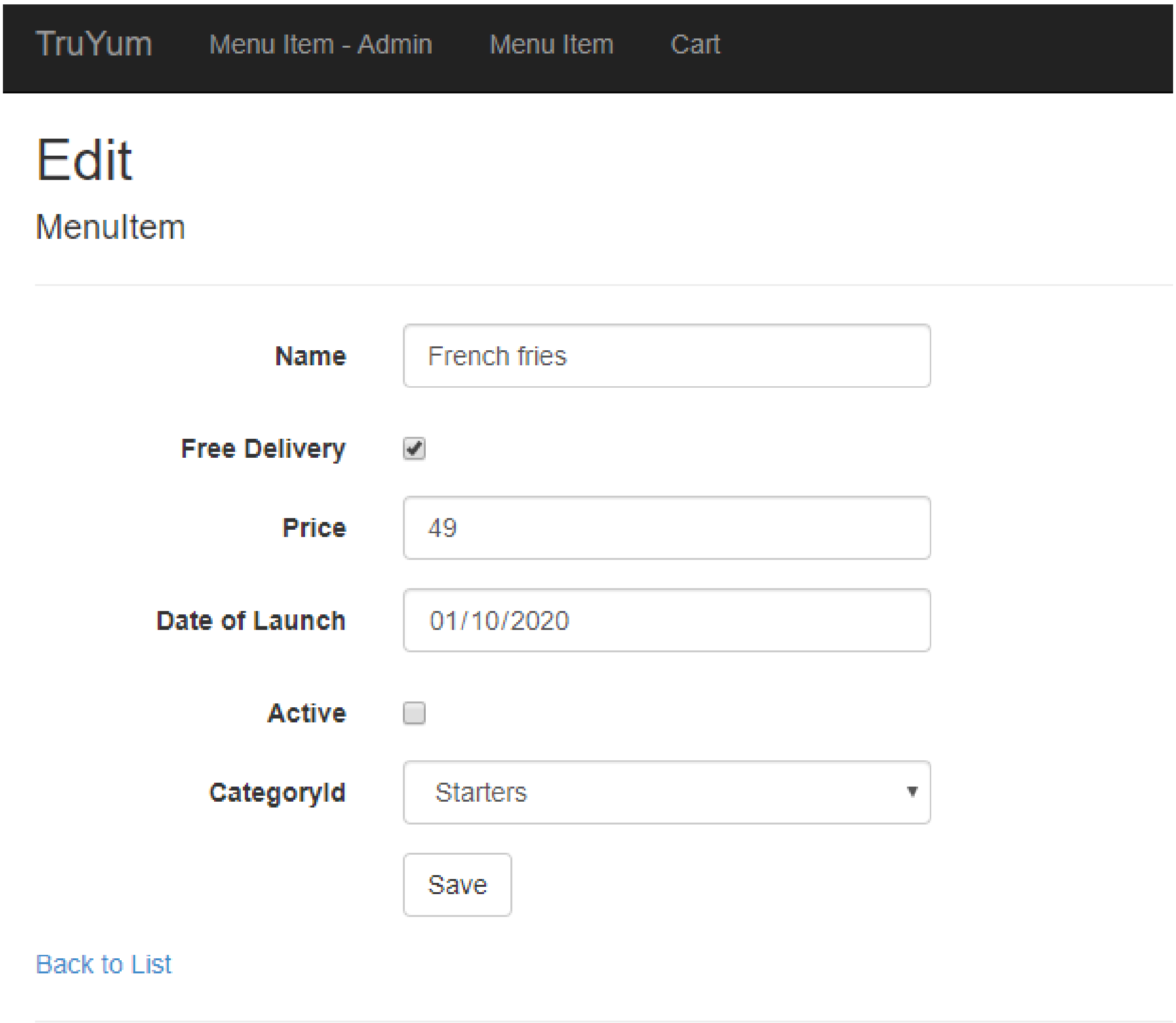
4. Ensure that the Category data is sent in ViewBag to be used to list in the dropdown. This should be used to set the data source to the DropDownList thru TagHelper

5. On editing the data, use Entity Framework to update the data

6. On any exception, user should be directed to “**Error**” view with a message `Exception occurred in [Action] method of [controller].` Along with this, please append the exception message.

4.2 Edit Menu Item

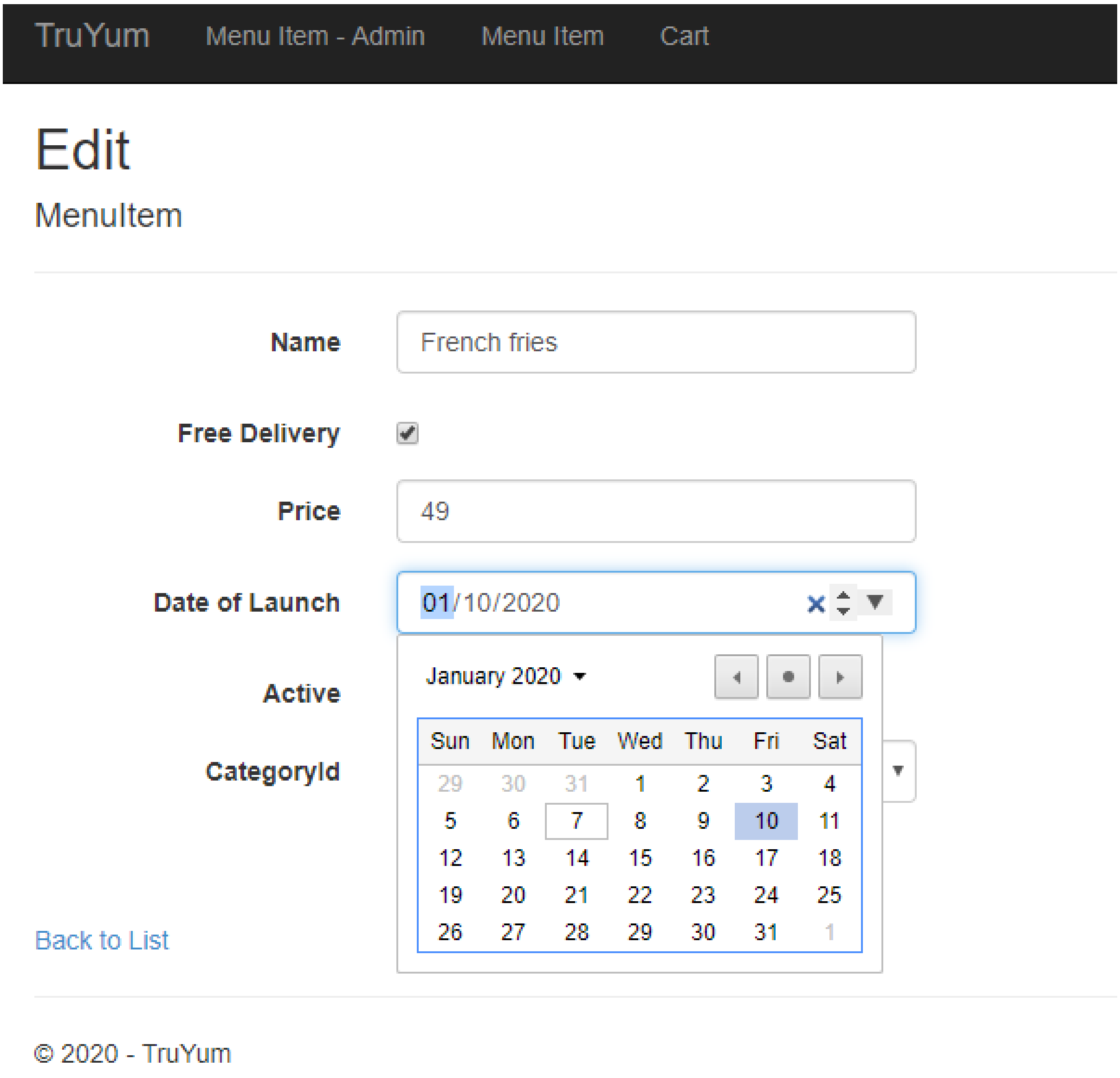
Explanation steps are not included for the above diagram as well as subsequent sample snapshots. Earlier explanation steps can be referred to arrive at the understanding.



Edit

MenuItem

Name	French fries
Free Delivery	☑
Price	49
Date of Launch	01/10/2020
Active	☐
CategoryId	Starters ▾



Edit

MenuItem

Name	French fries
Free Delivery	☑
Price	49
Date of Launch	01/10/2020
Active	
CategoryId	

January 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1

Back to List

© 2020 - TruYum

- On clicking the “**Save**” button, the data should be updated in database.
- The “Back to List” hyperlink(thru TagHelper) should navigate to the MenuItem listing page. The role of the user should be set while redirecting to Menu item list.

5.0 Design for Add Cart (TYUC004)

5.1 Requirement

Steps Explanation

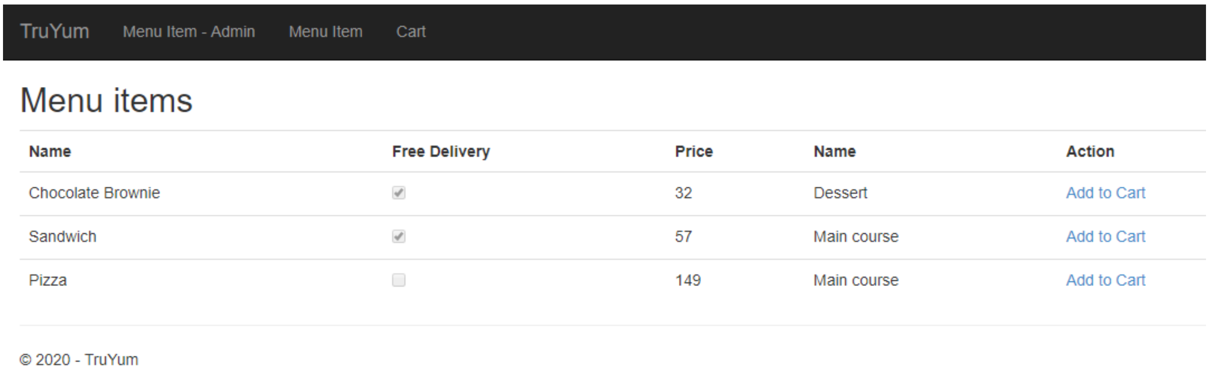
1. Customer clicks on “Add to Cart” link on a menu item from Menu Item List.

2. Customer is redirected to View Cart Url /Cart/AddToCart?menuItemId=3. Here 3 is the id of the chosen menu item id

3. The menu item should be added to the Cart table for the customer

4. On any exception, user should be directed to “**Error**” view with a message `Exception occurred in [Action] method of [controller].` Along with this, please append the exception message.

5.2 Add to Cart



Menu items

Name	Free Delivery	Price	Name	Action
Chocolate Brownie	☑	32	Dessert	Add to Cart
Sandwich	☑	57	Main course	Add to Cart
Pizza	□	149	Main course	Add to Cart

© 2020 - TruYum

5.3 Technical guidelines

- Use Entity Framework to add an entity to Cart table.

- Set the UserId to 1, a value to be picked from appSettings.

- Set the menuItemId to the querystring parameter

- On adding an item to cart, redirect the user to View the items in cart

6.0 Design for View Cart (TYUC005)

6.1 Requirement

Steps Explanation

1. Customer clicks on “Cart” link on the navigation bar.

2. Customer gets directed to the Url /Cart or /Cart/Index

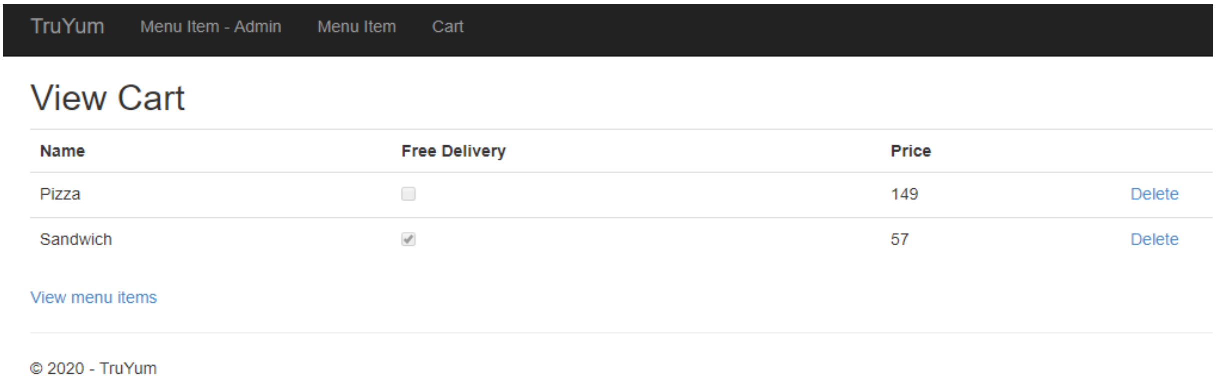
3. Set variable `userId` with type `int` having value 1

4. Get all the cart items of the user and show the menu item detail

5. If there are no cart items, then display a message and a link to View menu items to add to cart. This should direct the Customer to the Menu item list with a link to add the item to cart.

6. On any exception, user should be directed to “**Error**” view with a message `Exception occurred in [Action] method of [controller].` Along with this, please append the exception message.

6.1 View Cart



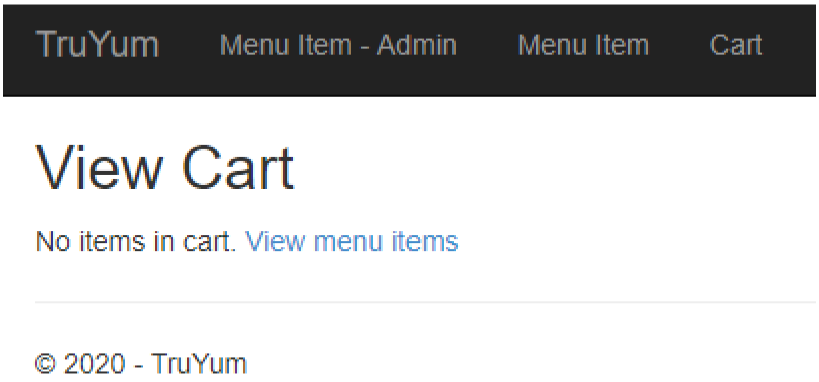
View Cart

Name	Free Delivery	Price	
Pizza	□	149	Delete
Sandwich	☑	57	Delete

View menu items

© 2020 - TruYum

6.1 No item in cart



View Cart

No items in cart. View menu items

© 2020 - TruYum

6.2 Technical guidelines

- Create CartController as an empty MVC controller
- Create Index action method to Get the cart items for the user whose Id is 1, a hardcoded value
- Include “System.Data.Entity” namespace
- Use Entity Framework’s Include feature to get the MenuItem detail, to get its data included with Cart detail
- Create a View for the menu item detail listing. Have link in each row to “Remove” item from cart. Bind the cart id to the link. Use appropriate tag helpers
- Use MVC Razor syntax in the View to check if the bound model contain data or not. If there is no data display textual content stating “There are no items in your cart.” Provide a link “View Menu Items” aiding the customer to navigate to Menu items page. This page has a link to add the item to cart.

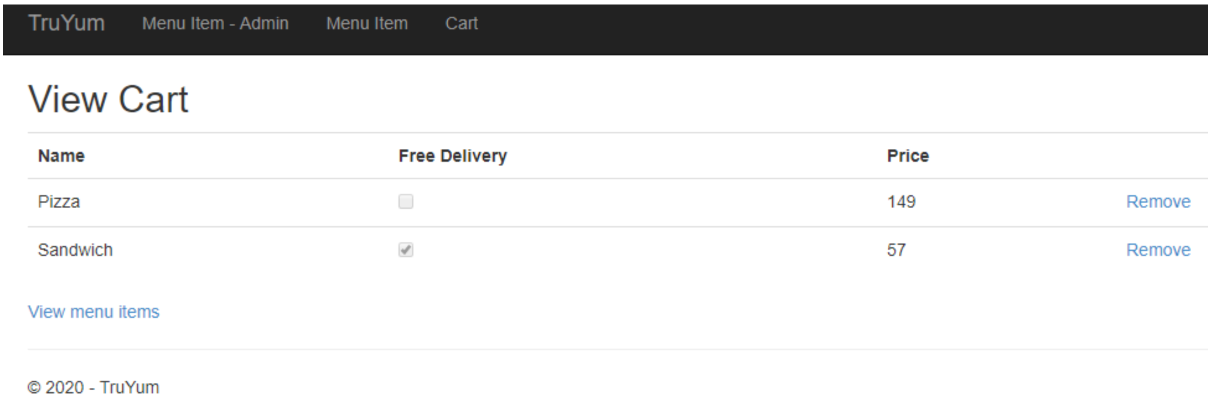
7.0 Design for Remove Item from Cart (TYUC006)

7.1 Requirement

Steps Explanation

1. Customer clicks on “Remove” link on menu item in the `Cart listed thru /Cart or /Cart/Index`.
2. On clicking the “Remove” link, the entity in Cart gets removed
3. Customer gets directed to the View cart page

7.2 Remove Cart item



View Cart

Name	Free Delivery	Price	
Pizza	☐	149	Remove
Sandwich	☑	57	Remove

View menu items

© 2020 - TruYum

7.3 Technical guidelines

- The View cart list has the Id bound to the “Remove” link
- Identify the Cart item from the Carts entity
- Delete the item and Save changes for the action to be updated in the database

8.0 Standards and Guidelines

8.1 Controller & View

1. Action methods should have a meaningful name
2. Remove unused Action methods

3. Use try catch blocks to catch exceptions

4. There should not be any hard coded values in code. It has to be referenced from Web.config file

5. Database connection string should be set in the ConnectionStrings section of Web.config and NOT in the AppSettings

6. Meaningful names should be given to the controls created in View

9.0 Change Log

<table>
<tr><th></th><th colspan="4">Changes Made</th></tr>
<tr><td>V1.0.0</td><td colspan="4">Initial baseline created on 6-Aug-20 by Seshadri M R</td></tr>
<tr><td rowspan="3">Vx.y.z</td><td colspan="4"><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></td></tr>
<tr><th>Section No.</th><th>Changed By</th><th>Effective Date</th><th>Changes Effected</th></tr>
<tr><td></td><td></td><td></td><td></td></tr>
</table>