Cognizant Academy

truYum

FSE Spring RESTful Specification Document

Version 1.0

	Prepared By / Last Updated By	Reviewed By	Approved By
Name	Chandrasekaran Janardhanan	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 user interface specification for truYum project.

1.2 Definitions & Acronyms

Definition / Acronym	Description
REST	Representational State Transfer

1.3 Project Overview

Refer truyum-user-stores.xlsx for understanding the functionality and features.

1.4 Scope

Develop RESTful Web Service Application for 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 Desktop PC with 8GB RAM
2. Software Requirement
    a. Eclipse

b. Postman

2.0 Project Setup in Eclipse

The Eclipse project for RESTful Web Service has to be created using the below mentioned steps.

1. Open Eclipse
2. File > New > Project > Maven > Maven Project > Next
3. Check 'Create a simple project'
4. Uncheck 'Use default Workspace location'
5. Click browse and select the truYum Java application provided as a part of skeleton code and click Next
6. Enter Group Id as 'com.cognizant'
7. Enter Artifact Id as 'truyum'
8. Click Finish
9. If error messages are displayed click OK
10. Copy pom.xml from the spring-learn project and overwrite the pom.xml of the newly created project
11. Modify the following properties in pom.xml, so that it reflects truYum project details:

    ```
    <artifactId>truyum</artifactId>
    <name>truyum</name>
    <description>truyum web services</description>
    ```

12. Update the maven configuration by right clicking on truyum project > Maven > Update Project > OK
13. In the root folder of the newly created project include .gitignore file with the following content, which will ensure that unwanted files are not pushed to git:

    ```
    /.settings
    /build
    .classpath
    .project
    /bin/
    /target/
    ```

14. Create following source files (refer code from spring-learn):

    a. com.cognizant.truyum.TruyumApplication (Spring Boot application class, refer SpringLearnApplication.java in spring-learn application, include annotations and main method from this source file)

b. com.cognizant.truyum.WebConfig

c. com.cognizant.truyum.exception.GlobalExceptionHandler

d. src/main/resources/application.properties – including configuration for logs and server port.

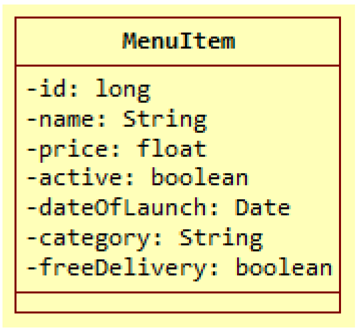
15. Change the logger pattern as specified below, which provides improved readability of logs. Change this configuration in application.properties:

```
logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-
25.25logger{25} %25M %4L %m%n
```

3.0 TYUS003 – View Menu Items

3.1 Rest API

com.cognizant.truyum.model.MenuItem - Create class com.cognizant.truyum.model.MenuItem



src/main/resources/truyum.xml - Include bean definition to load the sample data with list of menu items. Refer implemented screen for sample data

com.cognizant.truyum.dao.MenuItemDao – Copy this interface from Java module

com.cognizant.truyum.dao.MenuItemDaoCollectionImpl

- Copy this class from Java module
- Instead of hard coding the Menu Item list, load it from the spring xml configuration file truyum.xml

com.cognizant.truyum.service.MenuItemService

- Autowire MenuItemDao with MenuItemDaoCollectionImpl
- Include method getMenuItemListCustomer() which invokes getMenuItemListCustomer() from dao.

com.cognizant.truyum.controller.MenuItemController

- Include @RequestMapping("/menu-items") at class level
- Autowire MenuItemService

- Implement method getAllMenuItems() with @GetMapping that invokes getMenuItemListCustomer() and returns the menu item list

3.2 Testing the Service

  - Test the Veiw Menu Item functionality using PostMan

4.0 TYUC003 – Edit Menu Item

4.1 Populate form fields in Edit Menu Item component

4.1.1 Rest API

com.cognizant.truyum.controller.MenuItemController

- Method getMenuItem()
  - @GetMapping("/{id}")
  - Autowire MenuItemService
  - Invoke MenuItemService.getMenuItem(id)
  - Return the MenuItem instance

com.cognizant.truyum.controller.MenuItemService

- Method getMenuItem()
  - Autowire MenuItemDao
  - Invoke MenuItemDao.getMenuItem(id)
  - Return the MenuItem reference

com.cognizant.truyum.controller.MenuItemDaoCollectionImpl

Reuse existing code of getMenuItem()

4.1.2 Testing the Service

  - Test the Edit Menu Item functionality using PostMan

4.2 Save Menu Item

4.2.1 Rest API

com.cognizant.truyum.dao.MenuItemDaoCollectionImpl

Reuse existing code for ModifyMenuItem

com.cognizant.truyum.service.MenuItemService

- Method modifyMenuItem(MenuItem menuItem)
    - Invoke save method in MenuItemDao passing the menuItem

com.cognizant.truyum.controller.MenuItemController

- Method modifyMenuItem(@RequestBody MenuItem menuItem)
    - @PutMapping()
    - Invoke save() method in menu item service passing the menuItem

4.2.2 Testign the Service

- Test the Save Menu Item functionality using PostMan

5.0 TYUC004 – Add Item to Cart

5.1 Rest API

com.cognizant.truyum.dao.CartDao

- Copy and reuse the interface defined in Java Module

com.cognizant.truyum.dao.CartDaoCollectionImpl

- Copy and reuse the class from Java Module.
- Modify the map to type <String, Cart> instead of <long, Cart>. This is to handle storing the user as string directly. Modify code to handle other compilation errors on making this change.

com.cognizant.truyum.service.CartService

- Autowire CartDao with CartDaoCollectionImpl
- Include method addCartItem() with parameters the userId and menuItemId
- Invoke addCartItem() in CartDao passing userId and menuItemId

com.cognizant.truyum.controller.CartController

- @RequestMapping("/carts")
- Autowire CartService
- addCartItem() with annotation @PostMapping("/{userId}/{menuItemId}")
    - Get the userId and menuItemId from the parameter
    - Invoke addCartItem() in CartService passing the userId and menuItemId

5.1.1 Testing the Service

- Test the Add Item to Cart functionality using PostMan

6.0 TYUC005 – View Cart Items

6.1 Rest API

com.cognizant.truyum.dao.CartDaoCollectionImpl

The getAllCartItems() method will be reused here.

com.cognizant.truyum.service.CartService

- getAllCartItems()
  - Invoke getAllCartItems() in cartDao

com.cognizant.truyum.controller.CartController

- Autowire CartService
- getAllCartItems()
  - @GetMapping("/{userId}")
  - Invoke getAllCartItems() in CartService passing the userId

6.1.1 Testing the Service

- Test the View Cart Item functionality using PostMan

7.0 TYUC006 – Remove Cart Item

7.1 Rest API

com.cognizant.truyum.dao.CartDaoCollectionImpl

The deleteCartItem() method will be reused here.

com.cognizant.truyum.service.CartService

- deleteCartItem()
  - Invoke deleteCartItem() in cartDao

com.cognizant.truyum.controller.CartController

- Autowire CartService
- deleteCartItems()
  - @DeleteMapping("/{userId}/{menuItemId}")
  - Invoke deleteCartItem() in CartService passing the userId and menuItemId

7.1.1 Testing the Service

- Test the Remove Cart Item functionality using PostMan

8.0 Coding Standards and Guidelines

- Never use System.out.println(), use logger.debug() instead
- Include start and end logs in each method
- Include debug logs for data retrieval and flow
- Each Rest Controller class should have RequestMapping that maps to the particular domain entity
- The URL definition should be in all lower case with words separated with hyphen.
- Method level URL definitions in rest controller should contain only the parameters and should not contain any other URL definition.
- Apply the right http method based on the operation performed. POST for creation, PUT for updation, GET for reading data and DELETE for removing data.
- Do not include any instances level variables in controller apart from service auto wiring. Remember that in production environment based on volume of usage, there is a good possibility a rest controller method might be invoked in parallel, having instance variables with user specific information will overwrite one over the over and might result in bad user experience.

9.0 Change Log

	Changes Made			
V1.0.0	Initial baseline created on <dd-Mon-yy> by <Name of Author>			
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**