|  |  |
| --- | --- |
|  | **Cognizant Academy**  **truYum**  **FSE Spring Data JPA & Hibernate 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 4](#_Toc23211999)

[1.1 Purpose of this document 4](#_Toc23212000)

[1.2 Definitions & Acronyms 4](#_Toc23212001)

[1.3 Project Overview 4](#_Toc23212002)

[1.4 Scope 4](#_Toc23212003)

[1.5 Intended Audience 4](#_Toc23212004)

[1.6 Hardware and Software Requirement 4](#_Toc23212005)

[2.0 Project Setup in Eclipse 5](#_Toc23212006)

[3.0 Database and Bean Mapping configuration 5](#_Toc23212007)

[4.0 TYUS003 – View Menu Items (anonymous user) 5](#_Toc23212008)

[5.0 TYUS004 – Search Food Item 6](#_Toc23212009)

[6.0 TYUS005 – Add Item to Cart (Anonymous User) 6](#_Toc23212010)

[7.0 TYUS006 – Login (Customer & Admin) 6](#_Toc23212011)

[8.0 TYUC001 & TYUC002 – View Menu Item List (Admin & Customer) 6](#_Toc23212012)

[9.0 TYUC003 – Edit Menu Item 6](#_Toc23212013)

[9.1 Populate form fields in Edit Menu Item component 6](#_Toc23212014)

[9.2 Save Menu Item 6](#_Toc23212015)

[10.0 TYUC004 – Add Item to Cart 7](#_Toc23212016)

[11.0 TYUC005 – View Cart Items 7](#_Toc23212017)

[12.0 TYUC006 – Remove Cart Item 7](#_Toc23212018)

[13.0 TYUS006 – Signup 8](#_Toc23212019)

[13.1 Rest API 8](#_Toc23212020)

[14.0 TYUS008 – Logout 8](#_Toc23212021)

[15.0 Mockito 8](#_Toc23212022)

[16.0 Jenkins 8](#_Toc23212023)

[17.0 Coding Standards and Guidelines 8](#_Toc23212024)

[18.0 Submission 9](#_Toc23212025)

[18.1 Code submission instructions 9](#_Toc23212026)

[19.0 Change Log 9](#_Toc23212027)

# Introduction

## Purpose of this document

The purpose of this document is to define the user interface specification for truYum project.

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| JPA | Java Persistence API |

## Project Overview

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

## Scope

Develop RESTful Web Service that will be consumed by Angular

Application for truYum application.

## Intended Audience

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

## Hardware and Software Requirement

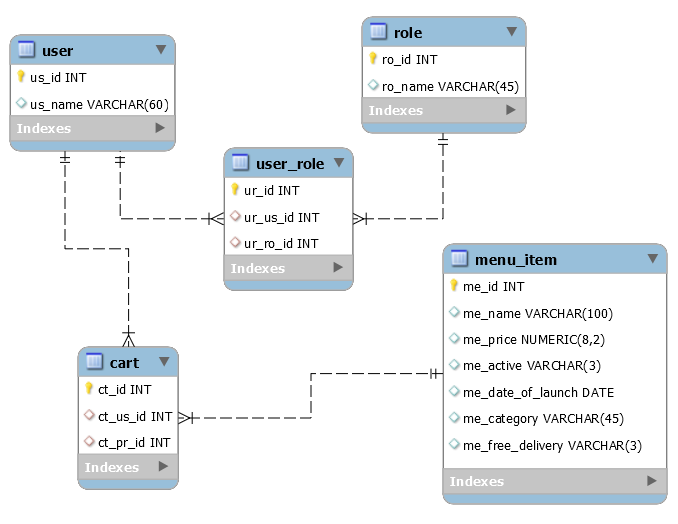
1. Hardware Requirement:
   1. Developer Desktop PC with 8GB RAM
2. Software Requirement
   1. Eclipse
   2. Postman
   3. Visual Studio Code
   4. Angular

# Project Setup in Eclipse

Use the same Eclipse project created in monolithic-service folder during Spring RESTful Web Service module.

# Database and Bean Mapping configuration

* Refer ER diagram below to create the schema for truyum application. If schema is already present and role is missing. Create role table and linking table. Ensure relevant data populated in user, role, user\_role and menu\_item table. In user table ensure that this is one role for customer and one role for admin.



* Create below folder structure to store SQL script files. schema.sql should contain the schema and table creation scripts. data.sql should contain the insert scripts for user, role, user\_role and menu\_item tables.

truyum-v2/monolithic-services/dbscripts/schema.sql

truyum-v2/monolithic-services/dbscripts/data.sql

* Modify pom.xml to support Spring Data JPA and Hibernate dependencies.
* Modify all classes in model package to refer to the respective tables and columns.
  + IMPORTANT NOTE
    - Move Cart class as CartDTO and move this class to dto package
    - Change all List to Set
* Based on the relationship type define necessary many-to-one, one-to-many and many-to-many relationships.
* Create necessary Repository classes for each model with empty implementation.

# TYUS003 – View Menu Items (anonymous user)

* Create a method in MenuItemRepository to filter out the active items and launch date is not in future
* Modify MenuItemService method to use the repository method.
* Test the respective REST API method using postman or curl
* Test the angular application to check if the menu items are loaded from database

# TYUS004 – Search Food Item

Since the search food item is implemented as part of angular component along, it is sufficient to check if search works correctly.

# TYUS005 – Add Item to Cart (Anonymous User)

This scenario is not applicable, as this will redirect to login page and it is already taken care in angular application.

# TYUS006 – Login (Customer & Admin)

* Refer UserDetailService implementation from spring-learn application and incorporate similar changes here, so that the authentication happens from the database instead of in memory.
* From the angular application test login for Customer role and Admin role. In customer role, check if certain menu items are not shown. In admin role, all menu items in database should be shown.

# TYUC001 & TYUC002 – View Menu Item List (Admin & Customer)

* Modify getMenuItemListAdmin() to invoke repository’s findAll() method.
* Modify getMenuItemListCustomer() to invoke the method defined in “TYUS003 – View Menu Items (anonymous user)” section of this document.
* Test the angular application and check if the results show are appropriate for Admin and Customer role.

# TYUC003 – Edit Menu Item

## Populate form fields in Edit Menu Item component

* Modify MenuItemService.getMenuItem() to use getOne() method of MenuItemRepository.
* Test if angular application shows the Edit Menu form populated from database when clicking on Edit link on a menu item.

## Save Menu Item

* Modify MenuItemService.modifyMenuItem() to use save() method of MenuItemRepository.
* Test if angular application saves the form details in the database.

# TYUC004 – Add Item to Cart

**CartService.addCartItem()**

* Get User object based on userId using UserRepository
* Get MenuItem object based on menuItemId using MenuItemRepository
* Add MenuItem object to User object and save the data.

# TYUC005 – View Cart Items

**com.cognizant.truyum.respository.UserRepository**

* Include an HQL query method getMenuItems() to get a User object that contains the list of menu items.
* Include an HQL query method getCartTotal() get the total of the menu items price.

**com.cognizant.truyum.service.CartService**

* getAllCartItems()
  + Invoke UserRepository.getMenuItems() and get the User object
  + Get the menu items from the user and return.
* double getCartTotal()
  + Invoke UserRepository.getCartTotal() to get the cart total and return the same.

**com.cognizant.truyum.controller.CartController**

* getAllCartItems()
  + Create new instance of CartDTO
  + Set the CartDTO.menuItemList based on the CartService.getAllCartItems()
  + Set the CartDTO.total based on CartService.getCartTotal()
  + Return CartDTO

# TYUC006 – Remove Cart Item

**com.cognizant.truyum.service.UserService**

* deleteCartItem()
  + Get the User object using UserRepository.getMenuItems()
  + Iterate through the menuItemList and remove an item based on the matching menuItemId
  + Save user using UserRepository
* Test from angular application to verify if cart item gets removed in the database

# TYUS006 – Signup

**com.cognizant.truyum.repository.UserRespository**

* Create new method to get user by username.

**com.cognizant.truyum.service.UserDetailsService**

* Create new method named signup() with user object reference as parameter
* Using UserRepository get the user based on username
* If username exists, then throw UserAlreadyExistsException
* If username is nonexistent, call save method of UserRepository

**com.cognizant.truyum.controller.UserController**

* Modify signup() method to invoke signup() method of UserDetailsService
* Include UserAlreadyExistsException in the throws clause of signup() which will take care of throwsing appropriate exception.

Test user signup from angular application. Also test with existing user.

# TYUS008 – Logout

No change required for handling logout.

# Mockito

Implement mockito based using testing for UserDetailsService.signup() method and demonstrate code coverage.

# Jenkins

Build truYum application in Jenkins and demonstrate successful build.

# Coding Standards and Guidelines

* Database script files should be placed in the relevant folders
* Define @Transactional annotation in all service methods
* 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
* All standards defined in Java module needs to be followed.

# Submission

## Code submission instructions

Use git add, commit and push commands to upload your code into remote GltLab repository.

During commit, give the commit message as “spring-data-jpa”.

# 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** |
|  |  |  |  |