|  |  |
| --- | --- |
|  | **Cognizant Academy**  **truYum**  **JDBC 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** | 23 May 2019 | 23 May 2019 | 17 Jun 2019 | |
|  |

Table of Contents

[1.0 Introduction 3](#_Toc12282365)

[1.1 Purpose of this document 3](#_Toc12282366)

[1.2 Definitions & Acronyms 3](#_Toc12282367)

[1.3 Project Overview 3](#_Toc12282368)

[1.4 Scope 3](#_Toc12282369)

[1.5 Intended Audience 3](#_Toc12282370)

[1.6 Hardware and Software Requirement 3](#_Toc12282371)

[2.0 Class Diagram 5](#_Toc12282372)

[2.1 Data Access Layer 5](#_Toc12282373)

[2.2 ConnectionHandler.java 6](#_Toc12282374)

[3.0 DAO for View Menu Item List Admin (TYUC001) 6](#_Toc12282375)

[3.1 MenuItemDaoSqlImpl.java 6](#_Toc12282376)

[4.0 DAO for View Menu Item List Customer (TYUC002) 6](#_Toc12282377)

[4.1 MenuItemDaoSqlImpl.java 7](#_Toc12282378)

[5.0 DAO for Edit Menu Item (TYUC003) 7](#_Toc12282379)

[5.1 MenuItemDaoSqlImpl.java 7](#_Toc12282380)

[6.0 DAO for Add a Menu Item to Cart (TYUC004) 8](#_Toc12282381)

[6.1 CartDaoSqlImpl.java 8](#_Toc12282382)

[7.0 DAO for View Cart (TYUC005) 8](#_Toc12282383)

[7.1 CartDaoSqlImpl.java 8](#_Toc12282384)

[8.0 DAO for Remove Item from Cart (TYUC006) 8](#_Toc12282385)

[8.1 CartDaoSqlImpl.java 8](#_Toc12282386)

[9.0 Integration of Dao with Servlets 9](#_Toc12282387)

[10.0 Standards and Guidelines 9](#_Toc12282388)

[10.1 DAO 9](#_Toc12282389)

[11.0 Submission 9](#_Toc12282390)

[11.1 Code submission instructions 9](#_Toc12282391)

[12.0 Change Log 10](#_Toc12282392)

# Introduction

## Purpose of this document

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

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| DAO | Data Access Object |
| JDBC | Java Database Connectivity |

## Project Overview

Refer truYum-use-case-specification.docx for understanding the functionality and features.

## Scope

1. Creation of DAO classes and methods for reading and persisting data of 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 PC with 4GB Ram
2. Software Requirement
   1. Git
   2. JDK 1.8
   3. Eclipse IDE for Enterprise Java Developers 2019-03 R
   4. MySQL Community Server 8.0

# Class Diagram

## 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.

Test method specification is not provided in this document. Refer similar implementation in Java Specification document.

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

## ConnectionHandler.java

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

The connection details has to be stored in a properties file. Refer details below:

1. File: truYum/src/connection.properties
2. Content for connection.properties:

driver= com.mysql.jdbc.Driver

connection-url= jdbc:mysql://localhost:3306/lch\_marketplace

user=root

password=password123

**static getConnection(): Connection**

1. Using java.io.FileInputStream and java.util.Properties read the properties from connection.properties file.
2. Gets connection using ConnectionManager based on properties from previous step and return the connection.

# DAO for View Menu Item List Admin (TYUC001)

## MenuItemDaoSqlImpl.java

**getMenuItemListAdmin(): List<MenuItem>**

1. Get connection using ConnectionHandler
2. Initialize an ArrayList of MenuItem
3. Using PreparedStatement 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 ArrayList created in the step 2 and return the ArrayList

# DAO for View Menu Item List Customer (TYUC002)

## MenuItemDaoSqlImpl.java

**getMenuItemListCustomer(): List<MenuItem>**

1. Get connection using ConnectionHandler
2. Initialize an ArrayList of MenuItem
3. Using PreparedStatement execute the select query that retrieves the records from menu\_item table applying the following filters:
   1. The menu item is active and
   2. The menu item is past the launch date
4. Iterate through the ResultSet
5. For each item in the ResultSet create a new MenuItem instance and add it to the ArrayList created in the step 2 and return the ArrayList

# DAO for Edit Menu Item (TYUC003)

## MenuItemDaoSqlImpl.java

**getMenuItem(menuItemId: long): MenuItem**

1. Get connection using ConnectionHandler
2. Execute select query using PreparedStatement 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): void**

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

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

## CartDaoSqlImpl.java

**addCartItem(userId: long, menuItemId: long): void**

1. Get connection using ConnectionHandler
2. Execute insert statement using PreparedStatement for inserting data into cart table with userId and menuItemId.

# DAO for View Cart (TYUC005)

## CartDaoSqlImpl.java

**getAllCartItems(userId: long): List<MenuItem>**

1. Get connection using ConnectionHandler
2. Create a new instance of Cart with new ArrayList<MenuItem> and price as 0 in Cart constructor.
3. Execute select statement using PreparedStatement that joins Cart and MenuItem table to retrieve the list of products filtered based on the input userId.
4. Iterate through the ResultSet
5. For each item in the ResultSet create a new Product instance and add it to the ArrayList created in the step 2 and return the ArrayList
6. Execute select statement using PreparedStatement that gets sum of price after joining Cart and MenuItem filtered based on input userId.
7. Set the totalPrice of the Cart based on the result from above query.

# DAO for Remove Item from Cart (TYUC006)

## CartDaoSqlImpl.java

**removeCartItem(userId: long, menuItemId: long): void**

1. Get connection using ConnectionHandler
2. Execute delete statement using PreparedStatement for delete data into cart table based on userId and menuItemId.

# Integration of Dao with Servlets

Instantiation of MenuItemDaoCollectionImpl and CartDaoCollectionImpl needs to be changed to instantiate MenuItemDaoSqlImpl and CartDaoSqlImpl in each Servlet. This will integrate the Servlet with Database.

# Standards and Guidelines

## DAO

1. All Java coding standards are applicable
2. Read database connection details from properties file
3. Closure of connection should be done within finally block

# Submission

## Code submission instructions

Once your code is evaluated by the trainer and all the issues reported by the trainer are corrected, the code needs to be submitted to the remote repository. Follow the steps below to submit the code to remote repository.

1. In Windows Explorer go to the truYum folder
2. Right click on the empty space in the right hand side of Windows Explorer and select “Git Bash here”
3. Execute the following commands

To display the added or modified files

git status

To stage the added or modified files

git add .

To display the staged files

git status

To save the code to local repository

git commit -m "jdbc"

To transfer the changes from local machine to server

git push origin master

1. Successful execution of the above commands will upload the files to the server repository.
2. Login into <https://code.cognizant.com>
3. Click on the project truYum
4. Check if the files that are uploaded correctly with appropriate folder structure.

# 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** |
|  |  |  |  |