Creating DTO and DAO classes

1. Create DTO/Model classes
   1. Go to Backend project.
   2. Create o package named “model” inside the package src>>main>>java>>com>>health>>HealthMedicineQuestBackEnd package.
   3. Create “Product” class inside the model package.

**package** com.health.HealthMedicineQuestBackEnd.model;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

@Entity

**public** **class** Product {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)//for autonumber

**int** productId;

String productName;

String productDescription;

**int** productPrice;

**int** productQuantity;

String imageUrl;

**public** **int** getProductId() {

**return** productId;

}

**public** **void** setProductId(**int** productId) {

**this**.productId = productId;

}

**public** String getProductName() {

**return** productName;

}

**public** **void** setProductName(String productName) {

**this**.productName = productName;

}

**public** String getProductDescription() {

**return** productDescription;

}

**public** **void** setProductDescription(String productDescription) {

**this**.productDescription = productDescription;

}

**public** **int** getProductPrice() {

**return** productPrice;

}

**public** **void** setProductPrice(**int** productPrice) {

**this**.productPrice = productPrice;

}

**public** **int** getProductQuantity() {

**return** productQuantity;

}

**public** **void** setProductQuantity(**int** productQuantity) {

**this**.productQuantity = productQuantity;

}

**public** String getImageUrl() {

**return** imageUrl;

}

**public** **void** setImageUrl(String imageUrl) {

**this**.imageUrl = imageUrl;

}

**public** Product(**int** productId, String productName, String productDescription, **int** productPrice,

**int** productQuantity, String imageUrl) {

**super**();

**this**.productId = productId;

**this**.productName = productName;

**this**.productDescription = productDescription;

**this**.productPrice = productPrice;

**this**.productQuantity = productQuantity;

**this**.imageUrl=imageUrl;

}

**public** Product() {

**super**();

// **TODO** Auto-generated constructor stub

}

}

1. Create DAO classes
   1. Create o package named “dao” inside the package src>>main>>java>>com>>health>>HealthMedicineQuestBackEnd package.
   2. Create an interface named IProductDAO inside the dao package.

package com.health.HealthMedicineQuestBackEnd.dao;

import java.util.List;

import com.health.HealthMedicineQuestBackEnd.model.Product;

public interface IProductDAO {

public List<Product> getAllProducts();

public Product getProduct(int id);

}

* 1. Create o package named “daoimpl” inside the package src>>main>>java>>com>>health>>HealthMedicineQuestBackEnd package.
  2. Create an class named ProductDAOImpl inside the daoimpl package.

**package** com.health.HealthMedicineQuestBackEnd.daoimpl;

**import** java.util.\*;

**import** org.hibernate.SessionFactory;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Repository;

**import** org.springframework.transaction.annotation.Propagation;

**import** org.springframework.transaction.annotation.Transactional;

**import** com.health.HealthMedicineQuestBackEnd.model.Product;

**import** com.health.HealthMedicineQuestBackEnd.dao.IProductDAO;

@Repository("productDAO")

@Transactional(propagation=Propagation.SUPPORTS, readOnly=**false**)

**public** **class** ProductDAOImpl **implements** IProductDAO{

@Autowired

SessionFactory sessionFactory;

**public** List<Product> getAllProducts() {

**return** sessionFactory.openSession().createQuery("from Product", Product.**class**).getResultList();

}

**public** Product getProduct(**int** id){

List<Product> prod =**new** ArrayList<Product>();

prod=getAllProducts();

**return** prod.get(id);

}

}

Linking Backend project with UI

1. Linking Backend project with Frontend project
   1. Open pom.xml file of backend project and copy the following code and paste inside the pom.xml file of front end project inside <dependencies> tag within <dependency></dependency> tag.

<groupId>com.health</groupId>

<artifactId>HealthMedicineQuestBackEnd</artifactId>

<version>0.0.1-SNAPSHOT</version>

* 1. Save the project and check the Maven dependency folder whether the backend project will be linked.

1. Add all Spring, Hibernate, H2, dbcp2, Jackson dependencies inside the pom.xml file of backend project as following
   1. Add following properties variable for versions

<spring.version>4.3.6.RELEASE</spring.version>

<jackson.version>2.8.7</jackson.version>

<persistence.version>1.0.1.Final</persistence.version>

<dbcp2.version>2.1.1</dbcp2.version>

<h2database.version>1.4.191</h2database.version>

<hibernate.version>5.2.5.Final</hibernate.version>

* 1. Add following dependencies code inside <dependencies> tag

<!-- SPRING FREAMEWORK -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- SPRING WEBMVC -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- CONTEXT -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- SPRING ORM -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- PERSISTENT -->

<dependency>

<groupId>org.hibernate.javax.persistence</groupId>

<artifactId>hibernate-jpa-2.0-api</artifactId>

<version>${persistence.version}</version>

</dependency>

<!-- COMMONS DBCP2 -->

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-dbcp2</artifactId>

<version>${dbcp2.version}</version>

</dependency>

<!-- H2 DATABASE -->

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>${h2database.version}</version>

</dependency>

<!-- HIBERNATE -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>${hibernate.version}</version>

</dependency>

<!-- JACKSON -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-core</artifactId>

<version>${jackson.version}</version>

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>${jackson.version}</version>

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-annotations</artifactId>

<version>${jackson.version}</version>

</dependency>

1. Use the dependency injection feature of Spring framework and ‘Autowired’ the class which implements DAO interface.
   1. Open FrontEndPageController class inside frontend project and autowired the IProductDAO object as below:

@Autowired

**private** IProductDAO productDAO;

1. Bringing daoimpl classes from backend to front end project by adding following code inside the dispatcher-servlet.xml file

<context:component-scan base-package="com.health.HealthMedicineQuestBackEnd.daoimpl"/>

1. Create a jsp page named Products.jsp inside the views folder with following code:

<div class=*"container content"*>

<div class=*"row"*>

<div class=*"col-md-12"*>

<!-- DataTable -->

<h2 style="color:">All Product Details</h2>

<div class=*"table-responsive"*>

<table id=*"myDataTable"* class=*"table table-bordered table-striped"*>

<thead>

<th>Product Id</th>

<th>Product Name</th>

<th>Product Benefit</th>

<th>Price/Unit</th>

<th>Product Image</th>

<th>View Item</th>

<th>Add To Cart</th>

</thead>

</table>

<!-- Modal -->

<c:forEach var=*"p"* items="${list}">

<div class=*"modal fade"* id=*"*${p.productId}*"* tabindex=*"-1"*

role=*"dialog"* aria-labelledby=*"myModalLabel"*>

<div class=*"modal-dialog"* role=*"document"*>

<div class=*"modal-content"*>

<div class=*"modal-header"*>

<button type=*"button"* class=*"close"* data-dismiss=*"modal"*

aria-label=*"Close"*>

<span aria-hidden=*"true"*>&times;</span>

</button>

<h2 class=*"modal-title"* id=*"myModalLabel"*>Product Details</h2>

</div>

<div class=*"modal-body"*>

<div class=*"col-sm-12"* id=*"letv"*>

<div class=*"thumbnail"*>

<img src=*"*${images}*/*${p.imageUrl}*"* alt=*""*>

<div class=*"caption"*>

<h2>${p.productName}</h2>

<h2>${p.productDescription}</h2>

<p>${p.productPrice}</p>

<p>

<a href=*"#"* class=*"btn btn-info btn-xs"* role=*"button"*>close</a>

</p>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</c:forEach>

</div>

</div>

</div>

</div>

1. Open page.jsp to add more css and js references as following:

<!-- Style -->

<link rel=*"stylesheet"* href=*"*${css}*/jquery.dataTables.min.css"*/>

<link rel=*"stylesheet"* href=*"*${css}*/bootstrap.min.css"*>

<link rel=*"stylesheet"* href=*"*${css}*/bootstrap-theme.min.css"*/>

<link rel=*"stylesheet"* href=*"*${css}*/dataTables.bootstrap.min.css"*/>

<link rel=*"stylesheet"* href=*"*${css}*/health.css"*>

<!-- jQuery library -->

<script src=*"*${jquery}*/jquery-3.1.1.min.js"*></script>

<!-- Latest compiled JavaScript -->

<script language=*"JavaScript"* type=*"text/javascript"* src=*"*${js}*/jquery.js"*></script>

<script language=*"JavaScript"* type=*"text/javascript"* src=*"*${js}*/jquery.dataTables.min.js"*></script>

<script language=*"JavaScript"* type=*"text/javascript"* src=*"*${js}*/bootstrap.min.js"*></script>

<script language=*"JavaScript"* type=*"text/javascript"* src=*"*${js}*/datatableScript.js"*></script>

<script language=*"JavaScript"* type=*"text/javascript"* src=*"*${js}*/dataTables.bootstrap.min.js"*></script>

1. Add the following code after contact inside the page.jsp file

<!-- Loading only when user clicks contact -->

<c:if test=*"*${userClickProducts==**true**}*"*>

<%@include file=*"products.jsp"* %>

</c:if>