Creating DTO and DAO classes

1.Creation of DTO class.

a.In the backend project , add model package inside thesrc>main>java>net>sports> ZenSportsBackEnd package .

b. Add a class , Product,java inside the model class.Add the following code inside it :

package net.sports.ZenSportsBackEnd.model;

import javax.persistence.\*;

@Entity

public class Product {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int productId;

private String productName;

private String productCategory;

private String productDescription;

private String productKeyword;

private String productImage;

private int productPrice;

private int productQuantity;

public Product() {

super();

}

public Product(int productId, String productName, String productCategory, String productDescription,

String productKeyword, String productImage, int productPrice, int productQuantity) {

super();

this.productId = productId;

this.productName = productName;

this.productCategory = productCategory;

this.productDescription = productDescription;

this.productKeyword = productKeyword;

this.productImage = productImage;

this.productPrice = productPrice;

this.productQuantity = productQuantity;

}

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 getProductCategory() {

return productCategory;

}

public void setProductCategory(String productCategory) {

this.productCategory = productCategory;

}

public String getProductDescription() {

return productDescription;

}

public void setProductDescription(String productDescription) {

this.productDescription = productDescription;

}

public String getProductKeyword() {

return productKeyword;

}

public void setProductKeyword(String productKeyword) {

this.productKeyword = productKeyword;

}

public String getProductImage() {

return productImage;

}

public void setProductImage(String productImage) {

this.productImage = productImage;

}

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;

}

}

2.Creation of dao class.

a.Also add package in the same level , dao.

b.Inside the dao package , add IProductDAO interface,add:

package net.sports.ZenSportsBackEnd.dao;

import net.sports.ZenSportsBackEnd.model.\*;

import java.util.\*;

public interface IProductDAO {

public List<Product> getProducts();

public Product getProduct(int id);

public void addProducts(Product p);}

3.Creation of daoimpl class.

a.Inside the daoimpl package , add ProductDAOImpl class , add :

package net.sports.ZenSportsBackEnd.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.Transactional;

import net.sports.ZenSportsBackEnd.model.\*;

import net.sports.ZenSportsBackEnd.dao.\*;

@Repository("productDAO")

@Transactional

public class ProductDAOImpl implements IProductDAO

{

@Autowired

private SessionFactory sessionFactory;

public List<Product> getProducts()

{

return sessionFactory.getCurrentSession().createQuery("FROM Product",Product.class).getResultList();

}

public void addProducts(Product p)

{

sessionFactory.getCurrentSession().persist(p);

}

public Product getProduct(int id)

{

List<Product> products = new ArrayList<Product>();

products = getProducts();

return products.get(id);

}

}

4.Add these dependencies in pom.xml file of backend project.

<properties>

<junit.version>3.8.1</junit.version>

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

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

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

<hibernatecore.version>5.2.5.Final</hibernatecore.version>

<h2.version>1.4.190</h2.version>

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

</properties>

<!-- JUnit dependency -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

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

<scope>test</scope>

</dependency>

<!-- Spring MVC Dependency -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

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

</dependency>

<!-- Spring orm dependency -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

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

</dependency>

<!-- Spring context dependency -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

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

</dependency>

<!-- Hibernate persistence Dependency -->

<dependency>

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

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

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

</dependency>

<!-- Commons-dbcp2 dependency -->

<dependency>

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

<artifactId>commons-dbcp2</artifactId>

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

</dependency>

<!-- Hibernate-core -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

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

</dependency>

<!-- H2 dependency -->

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>${h2.version}</version>

</dependency>

<!-- Jackson dependency -->

<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>