1.Start with yesterday’s project.

2.Create a package with the name model inside com.sports.yensports package of java folder.

3.Add a java class(POJO) with the name Product.java inside model,add the variables:

private int productId;

private String productName;

private String productCategory;

private String productDescription;

private String productKeyword;

private String productImage;

private int productPrice;

private int productQuantity;

4.Add contructor and getter/setter methods inside the Product.java class.(Rc select source)

5.Add a package with the name ProductDAO inside com.sports.yensports package of java folder.

6.Inside the package , add an interface with the name IProductDAO,and add :

public List<Product> getProducts();

public Product getProduct(int id);

7.Inside the same package , add a java class with the name ProductDAOImpl and add:

@Repository

public class ProductDAOImpl implements IProductDAO

{

private static List<Product> productList=new ArrayList<Product>();

static

{

productList.add(new Product(101,"ONMOVE 50 GPS WATCH","Fitness","Made for athletes to easily monitor their"

+ " speed and distance without connection to a computer Theeasiest GPS watch","GPS,Atheletes,"

+ "Monitor,Speed,Distance","P1.jpg",3999,40));

productList.add(new Product(102,"SOFT 300 KNEE BRACE - BLACK","Wellness","Made for supporting the knee with"

+ " even compression.The SOFT 300 knee brace is made from a comfortable, knit fabric that wicks "

+ "away perspiration.","Support,Perpiration-free,Comfortable","P2.jpg",499,10));

productList.add(new Product(103,"FORCLAZ 20 AIR BACKPACK: 1ST PRICE VENTILATED","Travel","Made for Hiking"

+ " for a half-day or a full day.20L: The 1st price with a ventilated back. Air Cooling Label: "

+ "The product's capacity to enhance air circulation to limit warmth and moisture","Backpack,"

+ "Capacity","P3.jpg",1799,5));

productList.add(new Product(104,"HIKING FOLDABLE FURNISHINGS GREEN","Travel","Made for furnishing hiking"

+ " camps.Simple and compact extra seating.","Kiking,Camps,Furnishing","P4.jpg",399,10));

productList.add(new Product(105,"ARPENAZ 100 MEN'S SHORT-SLEEVE HIKING SHIRT - GREEN","Travel","Made for "

+ "regular lowland hikers.Light, breathable, easy care shirt.","Shirt,Hike,Breathable","P5.jpg",399,50));

productList.add(new Product(106,"ARPENAZ 500 WOMEN'S LEATHER HIKING SHOES - GREY/GREEN","Fitness","Made "

+ "for day-long lowland hiking in dry weather and on technical terrain. For regular use. "

+ "These are women's hiking shoes designed for walks in the wilderness. Comfortable and light, "

+ "the ample adjustment of their lacing which extends down to the front of the foot.","Shoes,Hike,Terrain"

+ ",Walk","P6.jpg",1499,20)); }

public List<Product> getProducts()

{

return productList;

}

public Product getProduct(int id)

{

Product pr=new Product();

for(Product p:productList)

{

if(id==p.getProductId())

{

pr=p;

break;

}

}

return pr;

}

}

8. Add another java class(POJO) with the name Category.java inside model,add the variables:

private int categoryId;

private String categoryName;

9.Add getters and setters and constructors.

10.Add a package , CategoryDAO inside yensports,inside it add an interface ,ICategoryDAO,add code:

public List<Category> getCategories();

public Category getCategory(int id);

11.Inside the same package , add a java class , CategoryDAOImpl,add the code:

@Repository

public class CategoryDAOImpl implements ICategoryDAO

{

private static List<Category> categoryList=new ArrayList<Category>();

static

{

categoryList.add(new Category(1,"Fitness"));

categoryList.add(new Category(2,"Wellness"));

categoryList.add(new Category(3,"Travel"));

}

public List<Category> getCategories()

{

return new ArrayList<Category>();

}

public Category getCategory(int id)

{

Category cat=null;

for(Category c:categoryList)

{

if(c.getCategoryId()==id)

{

cat=c;

break;

}

}

return cat;

}

}

Linking DAO with controller

11.Open controller class ,YenSportsController,before all the request mapping we have to add dependency injection of spring framework with @Autowired for classes which implements DAO interfaces ,add :

@Autowired

private IProductDAO productDAO;

@Autowired

private ICategoryDAO categoryDAO;

NOTE:Do ctr+SHft+O to add packages.

12.In the ProductDAOImpl and CategoryDAOImpl,add repository annotation which is a marker for class specifying that it fulfilles the role of providing access to the data , which will be managed by the spring framework.Add:

@Repository("productDAO")

In product implementation class

@Repository("categoryDAO")

In category implementation class

NOTE:Here the names in parameters to annotation are the same names created in the controller.

13.In the dispatcher-servlet, after the context:component-scan , add all the packages to be scanner prior to executing the code .Add:

<context:component-scan base-package="com.sports.yensports.model" />

<context:component-scan base-package="com.sports.yensports.ProductDAO" />

<context:component-scan base-package="com.sports.yensports.CategoryDAO" />

14.In the pom.xml,add the dependency for context :

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

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

</dependency>

15.Also add jstl dependency in pom.xml:

<dependency>

<groupId>jstl</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

16.In the products.jsp of the views folder , add taglib for core tag:

[%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%](mailto:%25@taglib%20prefix=%22c%22%20uri=%22http://java.sun.com/jsp/jstl/core%22%25)

Using list as datasource for table in products.jsp

-----------------------------------------------------

17.In the YenSportsController.java , modify the Requestmapping for product page to :

@RequestMapping(value={"/product"})

public ModelAndView product(){

ModelAndView model=new ModelAndView("Products");

//Passing the list of products

model.addObject("products",productDAO.getProducts());

return model;

}

Here we are fetching data from the list of the productDAOImpl and store it inside products and then we can fetch it in the products.jsp page.

18.Modify the table tbody code of this page,to make it display the data of the list created in the ProductDAOImpl on table in rows:

<c:forEach items="${products}" var="product">

<tr>

<td>${product.productId}</td>

<td><img src="${images}/${product.productImage}" style="height:80px; width: 80px" /></td> <td>${product.productName}</td>

<td>${product.productCategory}</td>

<td>${product.productKeyword}</td> <td>&#8377;${product.productPrice}</td>

<td><p data-placement="top" title="View"><a href="${pageContext.request.contextPath}/productDetails" class="btn btn-primary btn-xs">

<span class="glyphicon glyphicon-search"></span></a></p></td>

<td><p data-placement="top" title="Cart">

<a href="#" class="btn btn-danger btn-xs"> <span class="glyphicon glyphicon-shopping-cart"></span></a></p>

</td>

</td>

</tr>

</c:forEach>