**Assisted Practice: 3.8 Integration of Hibernate with Spring**

This section will guide you to:

* Set up Eclipse to work with Spring and Hibernate
* Set up database tables to use Hibernate with Spring
* Create a pom.xml configuration file to add all the required components
* Create an HTML page to call a Controller
* Create a Controller to display product listing in a jsp page
* Create an Entity class, a DAO and a Service class to work with the database tables

**Development Environment**

* Eclipse IDE for Enterprise Java Developers v2019-03 (4.11.0)
* Apache Tomcat Server v9.0
* JRE: OpenJDK Runtime Environment 11.0.2
* All other software components are defined in the Project Object Model pom.xml

This lab has sixteen subsections, namely:

* + 1. Creating a web-enabled Maven project
    2. Creating pom.xml for including the required components
    3. Creating a table eproduct in the database and filling it with sample data
    4. Creating an Entity class EProductEntity
    5. Creating a DAO class EProductDAO
    6. Creating a Controller class EProductController
    7. Creating the dispatcher servlet eproduct-servlet.xml
    8. Creating a jdbc.properties file
    9. Creating a view productList.jsp to display the products
    10. Editing index.jsp
    11. Configuring Hibernate with hibernate.cfg.xml
    12. Configuring web.xml
    13. Building the project
    14. Publishing and starting the project
    15. Running the project
    16. Pushing the code to your GitHub repositories

**Step 3.8.1:** Creating a web-enabled Maven Project

* Open Eclipse
* Go the **File** menu. Choose **New->Maven Project**
* Uncheck **Create a Simple Project** and check **Use Default Workspace Location** and click on **Next**
* From the **archetype** list, choose the row that has **Artifact Id** as **maven-archetype-webapp** and click on **Next**
* Enter **Group Id** and **Artifact Id** as **SpringHibernateWeb** and click on **Finish**
* This will create the project files in the Project Explorer

**Step 3.8.2:** Creating pom.xml for including the required components

* In the Project Explorer, expand **SpringHibernateWeb** and double click on **pom.xml**
* Add the following entries:

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>SpringHibernateWeb</groupId>

<artifactId>SpringHibernateWeb</artifactId>

<packaging>war</packaging>

<version>0.0.1-SNAPSHOT</version>

<name>SpringHibernateWeb Maven Webapp</name>

<url>http://maven.apache.org</url>

<!-- JBoss repository for Hibernate -->

<repositories>

<repository>

<id>JBoss repository</id>

<url>http://repository.jboss.org/nexus/content/groups/public/</url>

</repository>

</repositories>

<properties>

<org.springframework.version>3.0.5.RELEASE</org.springframework.version>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

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

<artifactId>hibernate-jpa-2.1-api</artifactId>

<version>1.0.0.Final</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-expression</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-beans</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context-support</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.16</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>3.6.3.Final</version>

</dependency>

<dependency>

<groupId>javassist</groupId>

<artifactId>javassist</artifactId>

<version>3.12.1.GA</version>

</dependency>

<dependency>

<groupId>taglibs</groupId>

<artifactId>standard</artifactId>

<version>1.1.2</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>commons-dbcp</groupId>

<artifactId>commons-dbcp</artifactId>

<version>1.4</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.9</version>

</dependency>

</dependencies>

<build>

<finalName>SpringHibernateWeb</finalName>

</build>

</project>

**Step 3.8.3:** Creating a table eproduct in the database and filling it with sample data

* MySQL is already installed in your practice lab. (Refer FSD: Lab Guide - Phase 2)
* Login to the MySQL command line console
* Type **CREATE DATABASE ecommerce** and press **Enter**
* Type **USE ecommerce** and press **Enter**
* Enter the following script and execute it:

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8 \*/;

/\*!40103 SET @OLD\_TIME\_ZONE=@@TIME\_ZONE \*/;

/\*!40103 SET TIME\_ZONE='+00:00' \*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

--

-- Table structure for table `eproduct`

--

**DROP** **TABLE** **IF** **EXISTS** `eproduct`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

**CREATE** **TABLE** `eproduct` (

`**ID**` **bigint**(20) **NOT** **NULL** AUTO\_INCREMENT,

`name` **varchar**(100) **DEFAULT** **NULL**,

`price` **decimal**(10,2) **DEFAULT** **NULL**,

`date\_added` **timestamp** **NOT** **NULL** **DEFAULT** **CURRENT\_TIMESTAMP**,

`parts\_hdd` **varchar**(10) **DEFAULT** **NULL**,

`parts\_cpu` **varchar**(10) **DEFAULT** **NULL**,

`parts\_ram` **varchar**(10) **DEFAULT** **NULL**,

**PRIMARY** **KEY** (`**ID**`)

) ENGINE=InnoDB AUTO\_INCREMENT=4 **DEFAULT** CHARSET=latin1;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `eproduct`

--

**LOCK** **TABLES** `eproduct` **WRITE**;

/\*!40000 ALTER TABLE `eproduct` DISABLE KEYS \*/;

**INSERT** **INTO** `eproduct` **VALUES** (1,'HP Laptop ABC',21900.00,'2019-06-04 07:18:57','2 Gb HDD','AMD Phenom','4 Gb'),(2,'Acer Laptop ABC',23300.00,'2019-06-04 07:19:07','500 Gb HDD','Core-i7','4 Gb'),(3,'Lenovo Laptop ABC',33322.00,'2019-06-04 07:19:19','1 Tb HDD','Core-i7','8 Gb');

/\*!40000 ALTER TABLE `eproduct` ENABLE KEYS \*/;

**UNLOCK** **TABLES**;

/\*!40103 SET TIME\_ZONE=@OLD\_TIME\_ZONE \*/;

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

-- Dump completed on 2019-06-07 10:05:19

**Step 3.8.4:** Creating an Entity class EProductEntity

* In the Project Explorer, expand **SpringHibernateWeb->src->main**
* Right click on **main** and choose **New->Other**
* In the **Wizard** list, choose **Java->Class** and click on **Next**
* In **Package,** enter com.ecommerce.entity and in **Name** enter EProductEntityand click on **Finish**
* Enter the following code:

**package** com.ecommerce.entity;

**import** java.math.BigDecimal;

**import** java.util.Date;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

**@Entity**

**@Table(name= "eproduct")**

**public** **class** EProductEntity {

**@Id** **@GeneratedValue**

**@Column(name = "ID")**

**private** long ID;

**@Column(name = "name")**

**private** **String** name;

**@Column(name = "price")**

**private** **BigDecimal** price;

**@Column(name = "date\_added")**

**private** **Date** dateAdded;

**public** long getID() {**return** **this**.ID; }

**public** **String** getName() { **return** **this**.name;}

**public** **BigDecimal** getPrice() { **return** **this**.price;}

**public** **Date** getDateAdded() { **return** **this**.dateAdded;}

**public** void setID(long id) { **this**.ID = id;}

**public** void setName(**String** name) { **this**.name = name;}

**public** void setPrice(**BigDecimal** price) { **this**.price = price;}

**public** void setDateAdded(**Date** date) { **this**.dateAdded = date;}

}

**Step 3.8.5:** Creating a DAO class EProductDAO

* In the Project Explorer, expand **SpringHibernateWeb->src->main**
* Right click on **main** and choose **New->Other**
* In the **Wizard** list, choose **Java->Class** and click on **Next**
* In **Package,** enter com.ecommerce.dao and in **Name** enter EProductDAOand click on **Finish**
* Enter the following code:

**package** com.ecommerce.dao;

**import** java.util.List;

**import** org.hibernate.SessionFactory;

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

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

**import** org.springframework.stereotype.Service;

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

**import** com.ecommerce.entity.EProductEntity;

**@Repository**

**public** **class** EProductDAO {

**@Autowired**

**private** SessionFactory sessionFactory;

**@SuppressWarnings("unchecked")**

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

**return** **this**.sessionFactory.getCurrentSession().createQuery("from EProducts").list();

}

}

**Step 3.8.6:** Creating a Controller class EProductController

* In the Project Explorer, expand **SpringHibernateWeb->src->main**
* Right click on **main** and choose **New->Other**
* In the **Wizard** list, choose **Java->Class** and click on **Next**
* In **Package,** enter com.ecommerce.Controllerand in **Name** enter EProductControllerand click on **Finish**
* Enter the following code:

**package** com.ecommerce.controller;

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

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.ModelMap;

**import** org.springframework.validation.BindingResult;

**import** org.springframework.web.bind.annotation.ModelAttribute;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RequestMethod;

**import** org.springframework.web.bind.annotation.RequestParam;

**import** com.ecommerce.entity.EProductEntity;

**import** com.ecommerce.service.EProductService;

**@Controller**

**public** **class** EProductController {

**@Autowired**

**private** EProductService eproductService;

**@RequestMapping(value = "/productList", method = RequestMethod.GET)**

**public** **String** listProducts(ModelMap map)

{

map.addAttribute("eproduct", **new** EProductEntity());

map.addAttribute("productList", eproductService.getAllProducts());

**return** "productList";

}

}

**Step 3.8.7:** Creating the dispatcher servlet eproduct-servlet.xml

* In the Project Explorer, expand **SpringHibernateWeb->src->main->webapp**
* Right click on **WEB-INF** and choose **New->File**
* In filename,enter **eproduct-servlet.xml** and click on **Finish**
* Enter the following script:

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:aop="http://www.springframework.org/schema/aop"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:jee="http://www.springframework.org/schema/jee"

xmlns:lang="http://www.springframework.org/schema/lang"

xmlns:p="http://www.springframework.org/schema/p"

xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:util="http://www.springframework.org/schema/util"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/aop/ http://www.springframework.org/schema/aop/spring-aop.xsd

http://www.springframework.org/schema/context/ http://www.springframework.org/schema/context/spring-context.xsd

http://www.springframework.org/schema/jee/ http://www.springframework.org/schema/jee/spring-jee.xsd

http://www.springframework.org/schema/lang/ http://www.springframework.org/schema/lang/spring-lang.xsd

http://www.springframework.org/schema/tx/ http://www.springframework.org/schema/tx/spring-tx.xsd

http://www.springframework.org/schema/util/ http://www.springframework.org/schema/util/spring-util.xsd">

<context:annotation-config />

<context:component-scan base-package="com.ecommerce.controller" />

<bean id="jspViewResolver"

class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<property name="viewClass"

value="org.springframework.web.servlet.view.JstlView"></property>

<property name="prefix" value="/WEB-INF/view/"></property>

<property name="suffix" value=".jsp"></property>

</bean>

<bean id="messageSource"

class="org.springframework.context.support.ReloadableResourceBundleMessageSource">

<property name="basename" value="classpath:messages"></property>

<property name="defaultEncoding" value="UTF-8"></property>

</bean>

<bean id="propertyConfigurer"

class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"

p:location="/WEB-INF/jdbc.properties"></bean>

<bean id="dataSource"

class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close"

p:driverClassName="${jdbc.driverClassName}"

p:url="${jdbc.databaseurl}" p:username="${jdbc.username}"

p:password="${jdbc.password}"></bean>

<bean id="sessionFactory"

class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">

<property name="dataSource" ref="dataSource"></property>

<property name="configLocation">

<value>classpath:hibernate.cfg.xml</value>

</property>

<property name="configurationClass">

<value>org.hibernate.cfg.AnnotationConfiguration</value>

</property>

<property name="hibernateProperties">

<props>

<prop key="hibernate.dialect">${jdbc.dialect}</prop>

<prop key="hibernate.show\_sql">true</prop>

</props>

</property>

</bean>

<bean id="eproductDAO" class="com.ecommerce.dao.EProductDAO"></bean>

<bean id="eproductService" class="com.ecommerce.service.EProductService"></bean>

<tx:annotation-driven />

<bean id="transactionManager"

class="org.springframework.orm.hibernate3.HibernateTransactionManager">

<property name="sessionFactory" ref="sessionFactory"></property>

</bean>

</beans>

**Step 3.8.8:** Creating a jdbc.properties file

* In the Project Explorer, expand **SpringHibernateWeb->src->main->webapp**
* Right click on **WEB-INF** and choose **New->File**
* In filename,enter **jdbc.properties** and click on **Finish**
* Enter the following script:

jdbc.driverClassName=com.mysql.jdbc.Driver

jdbc.dialect=org.hibernate.dialect.MySQLDialect

jdbc.databaseurl=jdbc:mysql://127.0.0.1:3306/ecommerce

jdbc.username=userid

jdbc.password=password

**Step 3.8.9:** Creating a view productList.jsp to display the products

* In the Project Explorer, expand **SpringHibernateWeb->src->main->webapp->WEB-INF**
* Right click on **view** and choose **New->File**
* In filename,enter **productList.jsp** and click on **Finish**
* Enter the following code:

<%@taglib uri="http://www.springframework.org/tags" prefix="spring"%>

<%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>

<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<**html**>

<**head**>

<**title**>Spring With Hibernate</**title**>

</**head**>

<**body**>

<**h3**>Product List </**h3**>

<c:if test="${!empty productList}">

<**table** class="data">

<**tr**>

<**th**>Name</**th**>

<**th**>Price</**th**>

<**th**>Date Added</**th**>

</**tr**>

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

<**tr**>

<**td**>${product.name} </**td**>

<**td**>${product.price}</**td**>

<**td**>${product.date\_added}</**td**>

</**tr**>

</c:forEach>

</**table**>

</c:if>

</**body**>

</**html**>

**Step 3.8.10:** Editing index.jsp

* In the Project Explorer, expand **SpringHibernateWeb->src->main->webapp->WEB-INF->view**
* Double click on **index.jsp**
* Enter the following code:

<**html**>

<**body**>

<**h2**>Spring With Hibernate</**h2**>

<**a** href="/productList">Product List</**a**>

</**body**>

</**html**>

**Step 3.8.11:** Configuring Hibernate with hibernate.cfg.xml

* In the Project Explorer, expand **SpringHibernateWeb->src->main**
* Right click on **resources** and choose **New->Other**
* Select **General->File** and click on **Next**
* In filename, enter **hibernate.cfg.xml** and click on **Finish**
* Enter the following code:

<?xml version='1.0' encoding='utf-8'?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD//EN"

"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<mapping class="com.ecommerce.entity.EProductEntity"></mapping>

</session-factory>

</hibernate-configuration>

**Step 3.8.12:** Configuring web.xml

* In the Project Explorer, expand **SpringHibernateWeb->src->main->webapp->WEB-INF**
* Double click on **web.xml** to open it in the editor
* Enter the following script:

<!DOCTYPE web-app PUBLIC

"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"

"http://java.sun.com/dtd/web-app\_2\_3.dtd" >

<web-app>

<display-name>Archetype Created Web Application</display-name>

<welcome-file-list>

<welcome-file>/WEB-INF/view/index.jsp</welcome-file>

</welcome-file-list>

<servlet>

<servlet-name>eproduct</servlet-name>

<servlet-class>

org.springframework.web.servlet.DispatcherServlet

</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>eproduct</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/eproduct-servlet.xml</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

</web-app>

**Step 3.8.13:** Building the project

* From the **Project** menu at the top, click on **Build**
* If any compile errors are shown, fix them as required

**Step 3.8.14:** Publishing and starting the project

* If you do not see the **Servers** tab near the bottom of the IDE, go to **Window** menu and click on **Show View->Servers**
* Right click the **Server** entry and choose **Add and Remove**
* Click the **Add** button to move **SpringHibernateWeb** from the **Available** list to the **Configured** list
* Click on **Finish**
* Right click the **Server** entry and click on **Publish**
* Right click the **Server** entry and click on **Start**
* This will start the server

**Step 3.8.15:** Running the project

* To run the project, open a web browser and type: [**http://localhost:8080/**](http://localhost:8080/ServletConcept)**SpringHibernate**

**Step 3.8.16:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**