**Assisted Practice: 3.6 Hibernate Lazy Collection**

This section will guide you to:

* Set up Eclipse to work with Hibernate
* Set up database tables to do collection mapping using XML
* Create an HTML page to call a servlet
* Create a servlet that will display data from the tables using Lazy Collections

**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
* Hibernate for Java 5.2.1
* MySQL Connector for Java 8.0.16
* JTA v 1.1
* Java XML Bind (no version)
* JAXB OSGI v.2.4.0
* Java Activation (no version)

This lab has twenty-three subsections, namely:

* + 1. Creating a dynamic web project
    2. Adding the jar files for Hibernate and its dependencies
    3. Creating tables in MySQL: colors, eproduct, finance, os, screensizes in the database and fill them with sample data
    4. Creating a class Color
    5. Creating a class Eproduct
    6. Creating a class Finance
    7. Creating a class OS
    8. Creating a class ScreenSizes
    9. Creating a HibernateUtil class to initiate Hibernate in code
    10. Creating a hibernate table configuration file Color.hbm.xml
    11. Creating a hibernate table configuration file Eproduct.hbm.xml
    12. Creating a hibernate table configuration file Finance.hbm.xml
    13. Creating a hibernate table configuration file Os.hbm.xml
    14. Creating a hibernate table configuration file ScreenSizes.hbm.xml
    15. Configuring Hibernate with hibernate.cfg.xml
    16. Creating an HTML page index.html
    17. Creating a ProductDetails servlet
    18. Configuring web.xml
    19. Checking for servlet-api.jar
    20. Building the project
    21. Publishing and starting the project
    22. Running the project
    23. Pushing the code to your GitHub repositories

**Step 3.6.1:** Creating a dynamic web project

* Open Eclipse
* Go the **File** menu. Choose **New->Dynamic Web Project**
* Enter the project name as **HibernateLazyCollections**. Click on **Next**
* Enter nothing in the next screen and click on **Next**
* Check the checkbox **Generate web.xml deployment descriptor** and click on **Finish**
* This will create the project files in the Project Explorer

**Step 3.6.2:** Adding the jar files for Hibernate and its dependencies

* **Hibernate.jar** file is already present in your practice lab. (Refer FSD: Lab Guide - Phase 2)
* Take **hibernate.jar** from folder mentioned in the lab guide for phase 2 and add it to your project’s **WebContent/WEB-INF/lib** folder
* **mysql-connector-java.jar** file is present in your practice lab. (Refer FSD: Lab Guide - Phase 2)
* Take **mysql-connector-java.jar file** from the folder mentioned in the lab guide for phase 2 and add it to your project’s **WebContent/WEB-INF/lib** folder
* Go to <http://www.java2s.com/Code/Jar/j/Downloadjta11jar.htm>
* Click on **jta-1\_1.jar.zip** link to download it
* Extract **jta-1\_1.jar** from it and add it to your project’s **WebContent/WEB-INF/lib** folder
* Go to <http://www.java2s.com/Code/Jar/j/Downloadjavaxxmlbindjar.htm>
* Click on **javax.xml/javax.xml.bind.jar.zip** link to download it
* Extract **javax.xml.bind.jar** from it and add it to your project’s **WebContent/WEB-INF/lib** folder
* Go to <https://jar-download.com/artifacts/com.sun.xml.bind>
* Click on the button **Download jaxb-osgi.jar** to download it
* Extract **jaxb-osgi-2.4.0-b180830.0438.jar** from it and add it to your project’s **WebContent/WEB-INF/lib** folder

**Step 3.6.3:** Creating tables: colors, eproduct, finance, os, screensizes in the database and fill them 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:
* **DROP** **TABLE** **IF** **EXISTS** `colors`;
* /\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;
* /\*!40101 SET character\_set\_client = utf8 \*/;
* **CREATE** **TABLE** `colors` (
* `ID` bigint(20) NOT NULL AUTO\_INCREMENT,
* `color\_name` varchar(40) DEFAULT NULL,
* `idx` int(11) DEFAULT NULL,
* `product\_id` bigint(20) DEFAULT NULL,
* **PRIMARY** **KEY** (`ID`)
* ) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=latin1;
* /\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;
* --
* -- Dumping data for table `colors`
* --
* **LOCK** **TABLES** `colors` **WRITE**;
* /\*!40000 ALTER TABLE `colors` DISABLE KEYS \*/;
* **INSERT** **INTO** `colors` **VALUES** (1,'Red',0,1),(2,'Silver',1,1),(3,'Gray',0,2),(4,'White',1,2),(5,'Maroon',0,3);
* /\*!40000 ALTER TABLE `colors` ENABLE KEYS \*/;
* **UNLOCK** **TABLES**;
* --
* -- 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**,
* **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,'Acer Laptop ABC',23300.00,'2019-06-04 07:19:07'),(3,'Lenovo Laptop ABC',33322.00,'2019-06-04 07:19:19');
* /\*!40000 ALTER TABLE `eproduct` ENABLE KEYS \*/;
* **UNLOCK** **TABLES**;
* --
* -- Table structure for table `finance`
* --
* **DROP** **TABLE** **IF** **EXISTS** `finance`;
* /\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;
* /\*!40101 SET character\_set\_client = utf8 \*/;
* **CREATE** **TABLE** `finance` (
* `ID` bigint(20) NOT NULL AUTO\_INCREMENT,
* `ftype` varchar(10) DEFAULT NULL,
* `name` varchar(30) DEFAULT NULL,
* `product\_id` bigint(20) DEFAULT NULL,
* **PRIMARY** **KEY** (`ID`)
* ) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=latin1;
* /\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;
* --
* -- Dumping data for table `finance`
* --
* **LOCK** **TABLES** `finance` **WRITE**;
* /\*!40000 ALTER TABLE `finance` DISABLE KEYS \*/;
* **INSERT** **INTO** `finance` **VALUES** (1,'CREDITCARD','EMI on Citibank Card',1),(3,'BANK','40% finance from SBI',2),(4,'BANK','60% finance from ICICI',3),(5,'BANK','20% finance from ICICI',1);
* /\*!40000 ALTER TABLE `finance` ENABLE KEYS \*/;
* **UNLOCK** **TABLES**;
* --
* -- Table structure for table `os`
* --
* **DROP** **TABLE** **IF** **EXISTS** `os`;
* /\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;
* /\*!40101 SET character\_set\_client = utf8 \*/;
* **CREATE** **TABLE** `os` (
* `ID` bigint(20) NOT NULL AUTO\_INCREMENT,
* `name` varchar(30) DEFAULT NULL,
* `product\_id` bigint(20) DEFAULT NULL,
* **PRIMARY** **KEY** (`ID`)
* ) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=latin1;
* /\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;
* --
* -- Dumping data for table `os`
* --
* **LOCK** **TABLES** `os` **WRITE**;
* /\*!40000 ALTER TABLE `os` DISABLE KEYS \*/;
* **INSERT** **INTO** `os` **VALUES** (1,'Windows 10',1),(2,'Windows 10',2),(3,'FreeDOS',2),(4,'RedHat Linux',2),(5,'Windows 10',3);
* /\*!40000 ALTER TABLE `os` ENABLE KEYS \*/;
* **UNLOCK** **TABLES**;
* --
* -- Table structure for table `screensizes`
* --
* **DROP** **TABLE** **IF** **EXISTS** `screensizes`;
* /\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;
* /\*!40101 SET character\_set\_client = utf8 \*/;
* **CREATE** **TABLE** `screensizes` (
* `ID` bigint(20) NOT NULL AUTO\_INCREMENT,
* `size` varchar(10) DEFAULT NULL,
* `product\_id` bigint(20) DEFAULT NULL,
* **PRIMARY** **KEY** (`ID`)
* ) ENGINE=InnoDB AUTO\_INCREMENT=5 DEFAULT CHARSET=latin1;
* /\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;
* --
* -- Dumping data for table `screensizes`
* --
* **LOCK** **TABLES** `screensizes` **WRITE**;
* /\*!40000 ALTER TABLE `screensizes` DISABLE KEYS \*/;
* **INSERT** **INTO** `screensizes` **VALUES** (1,'12 in',1),(2,'14.5 in',2),(3,'14.9 in',2),(4,'15.5 in',3);
* /\*!40000 ALTER TABLE `screensizes` ENABLE KEYS \*/;
* **UNLOCK** **TABLES**;

**Step 3.6.4:** Creating a class Color

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click on **src** and choose **New->Class**
* In **Package,** enter com.ecommerce and in **Name** enter Colorand click on **Finish**
* Enter the following code:

**package** com.ecommerce;

**public** **class** Color {

**private** long COLORID;

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

**public** Color() {

}

**public** Color(**String** name) {

**this**.COLORID = 0;

**this**.name = name;

}

**public** long getCOLORID() {**return** **this**.COLORID; }

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

**public** void setCOLORID(long id) { **this**.COLORID = id;}

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

}

**Step 3.6.5:** Creating a class EProduct

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click on **src** and choose **New->Class**
* In **Package,** enter com.ecommerce and in **Name** enter EProductand click on **Finish**
* Enter the following code:

**package** com.ecommerce;

**import** java.math.BigDecimal;

**import** java.util.Collection;

**import** java.util.Date;

**import** java.util.List;

**import** java.util.Set;

**import** java.util.Map;

**public** **class** EProduct {

**private** long ID;

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

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

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

**private** **List**<Color> colors;

**private** **Set**<Finance> finance;

**private** PDescription pdescrip;

**public** EProduct() {

}

**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** **List**<Color> getColors() { **return** **this**.colors;}

**public** **Set**<Finance> getFinance() { **return** **this**.finance;}

**public** PDescription getPdescrip() { **return** **this**.pdescrip;}

**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;}

**public** void setColors(**List**<Color> colors) { **this**.colors = colors;}

**public** void setFinance(**Set**<Finance> finance) { **this**.finance = finance;}

**public** void setPdescrip(PDescription pdescrip) { **this**.pdescrip = pdescrip;}

}

**Step 3.6.6:** Creating a class Finance

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click **src** and choose **New->Class**
* In **Package** enter com.ecommerce and in **Name** enter Finance and click **Finish**
* Enter the following code:

**package** com.ecommerce;

**public** **class** Finance {

**private** long FINANCEID;

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

**private** **String** ftype;

**public** Finance() {

}

**public** Finance(**String** name, **String** ftype) {

**this**.FINANCEID = 0;

**this**.name = name;

**this**.ftype = ftype;

}

**public** long getFINANCEID() {**return** **this**.FINANCEID; }

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

**public** **String** getFtype() { **return** **this**.ftype;}

**public** void setFINANCEID(long id) { **this**.FINANCEID = id;}

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

**public** void setFtype(**String** ftype) { **this**.ftype= ftype;}

}

**Step 3.6.7:** Creating a class OS

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click on **src** and choose **New->Class**
* In **Package,** enter com.ecommerce and in **Name** enter OS and click on **Finish**
* Enter the following code:

**package** com.ecommerce;

**public** **class** OS {

**private** long OSID;

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

**public** OS() {

}

**public** OS(**String** name) {

**this**.OSID = 0;

**this**.name = name;

}

**public** long getOSID() {**return** **this**.OSID; }

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

**public** void setOSID(long id) { **this**.OSID = id;}

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

}

**Step 3.6.8:** Creating a class ScreenSizes

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click on **src** and choose **New->Class**
* In **Package,** enter com.ecommerce and in **Name** enter ScreenSizes and click on **Finish**
* Enter the following code:

**package** com.ecommerce;

**public** **class** ScreenSizes {

**private** long SCREENID;

**private** **String** size;

**public** ScreenSizes() {

}

**public** ScreenSizes(**String** size) {

**this**.SCREENID = 0;

**this**.size = size;

}

**public** long getSCREENID() {**return** **this**.SCREENID; }

**public** **String** getSize() { **return** **this**.size;}

**public** void setSCREENID(long id) { **this**.SCREENID = id;}

**public** void setSize(**String** size) { **this**.size = size;}

}

**Step 3.6.9:** Creating a HibernateUtil class to initiate Hibernate in code

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click on **src** and choose **New->Class**
* In **Package,** enter com.ecommerceand in **Name** enter HibernateUtiland click on **Finish**
* Enter the following code:

**package** com.ecommerce;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.boot.Metadata;

**import** org.hibernate.boot.MetadataSources;

**import** org.hibernate.boot.registry.StandardServiceRegistry;

**import** org.hibernate.boot.registry.StandardServiceRegistryBuilder;

**public** **class** HibernateUtil {

**private** **static** **final** SessionFactory sessionFactory;

**static** {

**try** {

StandardServiceRegistry standardRegistry = **new** StandardServiceRegistryBuilder()

.configure("hibernate.cfg.xml").build();

Metadata metaData = **new** MetadataSources(standardRegistry).getMetadataBuilder().build();

sessionFactory = metaData.getSessionFactoryBuilder().build();

} **catch** (**Throwable** th) {

**throw** **new** **ExceptionInInitializerError**(th);

}

}

**public** **static** SessionFactory getSessionFactory() {

**return** sessionFactory;

}

}

**Step 3.6.10:** Creating a hibernate table configuration file Color.hbm.xml

* In the Project Explorer, expand **HibernateConfig->Java Resources**
* Right click on **src** and choose **New->Other**
* Select **General->File** and click on **Next**
* In filename, enter **Color.hbm.xml** and click on **Finish**
* Enter the following code:

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="com.ecommerce">

<class name="Color" table="colors">

<id name="COLORID" type="long" column="ID">

<generator class="identity"/>

</id>

<property name="name" type="string" column="COLOR\_NAME"/>

</class>

</hibernate-mapping>

**Step 3.6.11:** Creating a hibernate table configuration file EProduct.hbm.xml

* In the Project Explorer, expand **HibernateConfig->Java Resources**
* Right click on **src** and choose **New->Other**
* Select **General->File** and click on **Next**
* In filename, enter **EProduct.hbm.xml** and click on **Finish**
* Enter the following code:

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="com.ecommerce">

<class name="EProduct" table="eproduct">

<id name="ID" type="long" column="ID">

<generator class="identity"/>

</id>

<property name="name" type="string" column="NAME"/>

<property name="price" type="big\_decimal" column="PRICE"/>

<property name="dateAdded" type="timestamp" column="DATE\_ADDED"/>

<list name="colors" cascade="all" lazy="true">

<key column="product\_id" />

<list-index column="idx" />

<one-to-many class="com.ecommerce.Color" />

</list>

<bag name="screensizes" cascade="all" lazy="true">

<key column="product\_id"></key>

<one-to-many class="com.ecommerce.ScreenSizes"/>

</bag>

<set name = "os" cascade="all" lazy=”true”>

<key column = "product\_id"/>

<one-to-many class="OS"/>

</set>

<**map** name = "finance" cascade="all">

<key column = "product\_id"/>

<index column = "ftype" type = "string"/>

<one-to-many class="com.ecommerce.Finance"/>

</**map**>

</class>

</hibernate-mapping>

**Step 3.6.12:** Creating a hibernate table configuration file Finance.hbm.xml

* In the Project Explorer, expand **HibernateConfig->Java Resources**
* Right click on **src** and choose **New->Other**
* Select **General->File** and click on **Next**
* In filename, enter **Finance.hbm.xml** and click on **Finish**
* Enter the following code:

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="com.ecommerce">

<class name="Finance" table="finance">

<id name="FINANCEID" type="long" column="ID">

<generator class="identity"/>

</id>

<property name="name" type="string" column="NAME"/>

<property name="ftype" type="string" column="FTYPE"/>

</class>

</hibernate-mapping>

**Step 3.6.13:** Creating a hibernate table configuration file Os.hbm.xml

* In the Project Explorer, expand **HibernateConfig->Java Resources**
* Right click on **src** and choose **New->Other**
* Select **General->File** and click on **Next**
* In filename, enter **Os.hbm.xml** and click on **Finish**
* Enter the following code:

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="com.ecommerce">

<class name="OS" table="os">

<id name="OSID" type="long" column="ID">

<generator class="identity"/>

</id>

<property name="name" type="string" column="NAME"/>

</class>

</hibernate-mapping>

**Step 3.6.14:** Creating a hibernate table configuration file ScreenSizes.hbm.xml

* In the Project Explorer, expand **HibernateConfig->Java Resources**
* Right click on **src** and choose **New->Other**
* Select **General->File** and click on **Next**
* In filename, enter ScreenSizes.hbm.xmland click on **Finish**
* Enter the following code:

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="com.ecommerce">

<class name="ScreenSizes" table="screensizes">

<id name="SCREENID" type="long" column="ID">

<generator class="identity"/>

</id>

<property name="size" type="string" column="SIZE"/>

</class>

</hibernate-mapping>

**Step 3.6.15:** Configuring Hibernate with hibernate.cfg.xml

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click on **src** 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 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<!-- Database connection settings -->

<property name="connection.driver\_class">com.mysql.jdbc.Driver</property>

<property name="connection.url">jdbc:mysql://localhost:3306/ecommerce</property>

<property name="connection.username">root</property>

<property name="connection.password">master</property>

<mapping resource="com/ecommerce/EProduct.hbm.xml"/>

<mapping resource="com/ecommerce/Color.hbm.xml"/>

<mapping resource="com/ecommerce/ScreenSizes.hbm.xml"/>

<mapping resource="com/ecommerce/Os.hbm.xml"/>

<mapping resource="com/ecommerce/Finance.hbm.xml"/>

</session-factory>

</hibernate-configuration>

**Step 3.6.16:** Creating an HTML page index.html

* In the Project Explorer, expand the project **HibernateLazyCollections**
* Expand **WebContent**. Right click on **WebContent**. Choose **New->HTML File**
* Enter the filename as **index.html** and click on **Finish**
* Enter the following code:

<!DOCTYPE html>

<**html**>

<**head**>

<**meta** charset="UTF-8">

<**title**>Hibernate Collection Mapping With XML</**title**>

</**head**>

<**body**>

<**a** href="details">Product Details</**a**><**br**>

</**body**>

</**html**>

* Click on the **Save** icon

**Step 3.6.17:** Creating a ProductDetails servlet

* In the Project Explorer, expand **HibernateLazyCollections->Java Resources**
* Right click on **src** and choose **New->Servlet**
* In **Class Name,** enter **ProductDetails** and click on **Finish**
* Enter the following code:

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** javax.servlet.ServletConfig;

**import** javax.servlet.ServletException;

**import** javax.servlet.annotation.WebServlet;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** javax.transaction.**\***;

**import** javax.xml.bind.**\***;

**import** java.io.Serializable;

**import** java.math.BigDecimal;

**import** java.util.ArrayList;

**import** java.util.Calendar;

**import** java.util.Collection;

**import** java.util.List;

**import** java.util.Map;

**import** java.util.Set;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.cfg.Configuration;

**import** com.ecommerce.Color;

**import** com.ecommerce.EProduct;

**import** com.ecommerce.Finance;

**import** com.ecommerce.HibernateUtil;

**import** com.ecommerce.PDescription;

/\*\*

**\*** Servlet implementation class ProductDetails

\*/

**@WebServlet("/ProductDetails")**

**public** **class** ProductDetails **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

/\*\*

**\*** **@see** HttpServlet**#**HttpServlet()

\*/

**public** ProductDetails() {

**super**();

// TODO Auto-generated constructor stub

}

/\*\*

**\*** **@see** HttpServlet**#**doGet(HttpServletRequest request**,** HttpServletResponse response)

\*/

**protected** void doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, **IOException** {

// TODO Auto-generated method stub

**try** {

SessionFactory factory = HibernateUtil.getSessionFactory();

Session session = factory.openSession();

**List**<EProduct> list = session.createQuery("from EProduct").list();

**PrintWriter** out = response.getWriter();

out.println("<html><body>");

out.println("<b>One to One Mapping</b><br>");

**for**(EProduct p: list) {

out.println("ID: " + **String**.valueOf(p.getID()) + ", Name: " + p.getName() +

", Price: " + **String**.valueOf(p.getPrice()) + ", Date Added: " + p.getDateAdded().toString());

PDescription descrip = p.getPdescrip();

out.println("<br>Description:" + descrip.getDescrip());

out.println("<hr>");

}

out.println("<b>One to Many and Many to One Mapping</b><br>");

**for**(EProduct p: list) {

out.println("ID: " + **String**.valueOf(p.getID()) + ", Name: " + p.getName() +

", Price: " + **String**.valueOf(p.getPrice()) + ", Date Added: " + p.getDateAdded().toString());

**List**<Color> colors = p.getColors();

out.println("<br>Colors: <ul>");

**for**(Color c: colors) {

out.print("<li>" + c.getName() + "</li>");

}

out.println("</ul>");

out.println("<hr>");

}

out.println("<b>Many to Many Mapping</b><br>");

**for**(EProduct p: list) {

out.println("ID: " + **String**.valueOf(p.getID()) + ", Name: " + p.getName() +

", Price: " + **String**.valueOf(p.getPrice()) + ", Date Added: " + p.getDateAdded().toString());

**Set**<Finance> finances= p.getFinance();

out.println("<br>Finance Options : <ul>");

**for**(Finance f: finances) {

out.print("<li>" + f.getFtype() + "</li>");

}

out.println("</ul>");

out.println("<hr>");

}

session.close();

out.println("</body></html>");

} **catch** (**Exception** ex) {

**throw** ex;

}

}

/\*\*

**\*** **@see** HttpServlet**#**doPost(HttpServletRequest request**,** HttpServletResponse response)

\*/

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, **IOException** {

// TODO Auto-generated method stub

doGet(request, response);

}

}

**Step 3.6.18:** Configuring web.xml

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

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_4\_0.xsd" id="WebApp\_ID" version="4.0">

<display-name>HibernateLazyCollections</display-name>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

<welcome-file>index.htm</welcome-file>

<welcome-file>index.jsp</welcome-file>

<welcome-file>default.html</welcome-file>

<welcome-file>default.htm</welcome-file>

<welcome-file>default.jsp</welcome-file>

</welcome-file-list>

<servlet>

<servlet-name>ProductDetails</servlet-name>

<servlet-class>ProductDetails</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ProductDetails</servlet-name>

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

</servlet-mapping>

</web-app>

**Step 3.6.19:** Checking for servlet-api.jar

* Before building the project, we need to add **servlet-api.jar** to the project
* Servlet-api.jar file is already present in your practice lab. (Refer FSD: Lab Guide - Phase 2)
* To add it to the project, follow the below mentioned steps:
  + In the Project Explorer, right click on **HibernateLazyCollections** and choose **Properties**
  + Select **Java Build Path** from the options on the left
  + Click on **Libraries** tab on the right
  + Under **ClassPath,** expand the node that says **Apache Tomcat**
  + If there is an existing entry for **servlet-api.jar,** then click on **Cancel** and exit the window
  + If it is not there, then click on **Classpath** entry and click on **Add External JARs** button on the right
  + From the file list, select **servlet-api.jar** file and click on **Ok**
  + Click on **Apply and Close**

**Step 3.6.20:** Building the project

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

**Step 3.6.21:** 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 **HibernateLazyCollections** 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 3.6.22:** Running the project

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

**Step 3.6.23:** 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**