Phase-2 Practice Project: Assisted Practice

7.Demonstrate component mapping in Hibernate.

EProduct.java

```
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 ProductParts parts;

        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 ProductParts getParts() { return this.parts;}

        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 setParts(ProductParts parts) { this.parts = parts;}
}
```

ProductParts.java

```
package com.ecommerce;


public class ProductParts {


        private String hdd;
    private String cpu;
    private String ram;
    public String getHdd() { return this.hdd;}

    public String getCpu() { return this.cpu;}

    public String getRam() { return this.ram;}


    public void setHdd(String value) { this.hdd= value;}

    public void setCpu(String value) { this.cpu= value;}

    public void setRam(String value) { this.ram= value;}


}
```

HibernateUtil

```
package com.simpli;


import org.hibernate.SessionFactory;

import org.hibernate.boot.*;

import org.hibernate.boot.registry.*;


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) {
            System.out.println(th);
            throw new ExceptionInInitializerError(th);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

}
```

EProduct.hbm.xml

```
<?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" column="ID">
                <generator class="increment" />
        </id>
        <property name="name" type="string" column="NAME" />
        <property name="price" type="big_decimal" column="PRICE" />
        <property name="dateAdded" type="timestamp"
                column="DATE_ADDED" />

                <component name="parts" class="com.ecommerce.ProductParts">
                <property name="hdd" column="parts_hdd" type="string" />
                <property name="cpu" column="parts_cpu" type="string" />
                <property name="ram" column="parts_ram" type="string" />
        </component>

        <list name="colors" cascade="all">
                <key column="product_id" />
                <list-index column="idx" />
                <one-to-many class="com.ecommerce.Color" />
        </list>

        <set name="os" cascade="all">
                <key column="product_id" />
                <one-to-many class="com.ecommerce.OS" />
        </set>
```

```
                <bag name="screenSizes" cascade="all">
                        <key column="product_id"></key>
                        <one-to-many class="com.ecommerce.ScreenSizes" />
                </bag>
<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>
```

hibernate.cfg.xml

```
<?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.cj.jdbc.Driver</property>
    <property name="connection.url">jdbc:mysql://localhost:3306/ecommerce</property>
    <property name="connection.username">root</property>
    <property name="connection.password">Simplilearn</property>
    <property name="dialect">org.hibernate.dialect.MySQL8Dialect</property>

   <mapping  resource="com/ecommerce/Color.hbm.xml"/>
   <mapping  resource="com/ecommerce/OS.hbm.xml"/>
   <mapping  resource="com/ecommerce/ScreenSizes.hbm.xml"/>
   <mapping  resource="com/ecommerce/Finance.hbm.xml"/>
   <mapping  resource="com/ecommerce/EProduct.hbm.xml"/>

  </session-factory>
</hibernate-configuration>
```

index.html

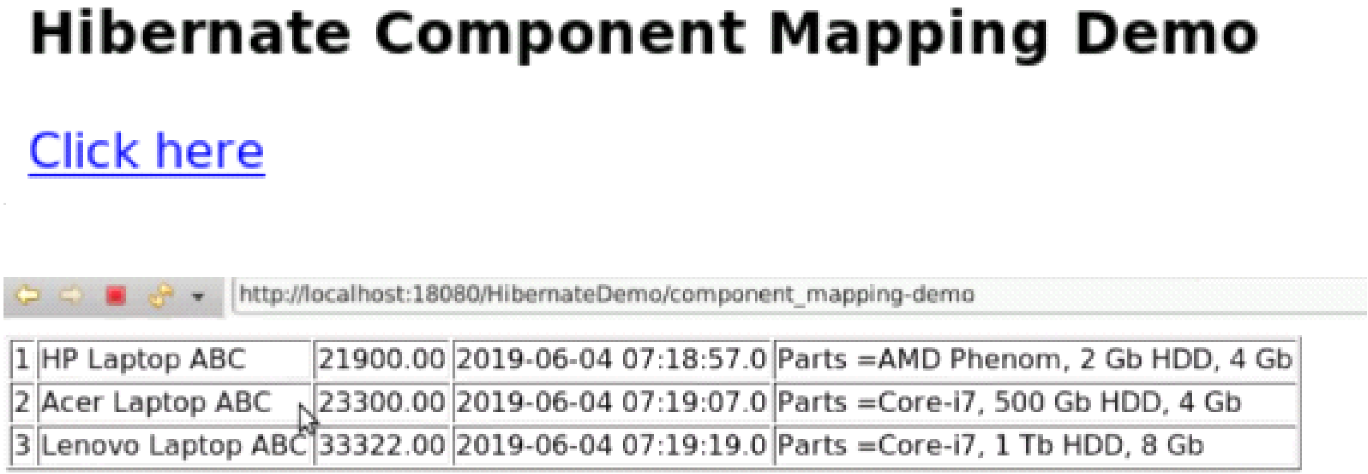
```
<title>Hibernate Demo  </title>
<h3>Hibernate Configuration Example </h3>

<a href="init">Initialize Hibernate</a><br>

<br><br>
<h3>Hibernate Query Demo </h3>

<a href="query-demo">Query  Here and get the data</a><br>

<br><br>
```

```
<h3>Hibernate Collections Mapping  Demo </h3>

<a href="product-details">Click here</a><br>


<br><br>

<h3>Hibernate Component Mapping  Demo </h3>

<a href="component_mapping-demo">Click here</a><br>
```

Productdetails.java

```
package com.simpli;


import java.io.*;

import java.util.*;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.*;

import org.hibernate.*;

import com.ecommerce.*;


@WebServlet("/product-details")

public class ProductDetailsServlet extends HttpServlet {


        protected void doGet(HttpServletRequest request, HttpServletResponse response)

                                throws ServletException, IOException {

                PrintWriter out = response.getWriter();

                out.println("<html><body>");
        SessionFactory factory = HibernateUtil.getSessionFactory();

        Session session = factory.openSession();


        // STEP 3 Query the DB and get the data

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


        out.println(

                "<table
border=1><th>ID<th>NAME<th>PRICE<th>DATE_ADDED<th>COLORS<th>OS<th>SCREENSIZES<
th>FINANCE</th>");


        for (EProduct prod : eproducts) {

            out.println("<tr>" + "<td>" + prod.getID() + "<td>" + prod.getName() +
"<td>" + prod.getPrice() + "<td>"+ prod.getDateAdded());

// Display the colors the product is available in

            List<Color> colors = prod.getColors();

            out.println("<td>");

            for (Color color : colors)

                out.println(color.getName());


            // Display the OS the product is available in

            Set<OS> Oses = prod.getOs();

            out.println("<td>");

            for (OS os : Oses)

                out.println(os.getName());


            // Display the Screen Sizes the product is available in
            Collection<ScreenSizes> screenSizes = prod.getScreenSizes();
            out.println("<td>");
            for (ScreenSizes ss : screenSizes)
                out.println(ss.getSize());

//          Map finances = prod.getFinance();
//          out.println("<td>");
//    if (finances.get("CREDITCARD") != null) {
//        Finance f = (Finance) finances .get("CREDITCARD");
//        out.println(f.getName() + " &nbsp;");
//    }
//    if (finances .get("BANK") != null) {
//        Finance f = (Finance) finances .get("BANK");
//        out.println(f.getName() + " &nbsp;");
//    }

        }
out.println("</table>");

        session.close();
        out.println("</body></html>");
    }
}
```

Output



Hibernate Component Mapping Demo

Click here

http://localhost:18080/HibernateDemo/component_mapping-demo

1	HP Laptop ABC	21900.00	2019-06-04 07:18:57.0	Parts =AMD Phenom, 2 Gb HDD, 4 Gb
2	Acer Laptop ABC	23300.00	2019-06-04 07:19:07.0	Parts =Core-i7, 500 Gb HDD, 4 Gb
3	Lenovo Laptop ABC	33322.00	2019-06-04 07:19:19.0	Parts =Core-i7, 1 Tb HDD, 8 Gb