1.Configure Hibernate in Eclipse IDE.

hibernate.cfg.xml

```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">8143303511@Sri</property>

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

Index.html

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

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

HibernateUtil class

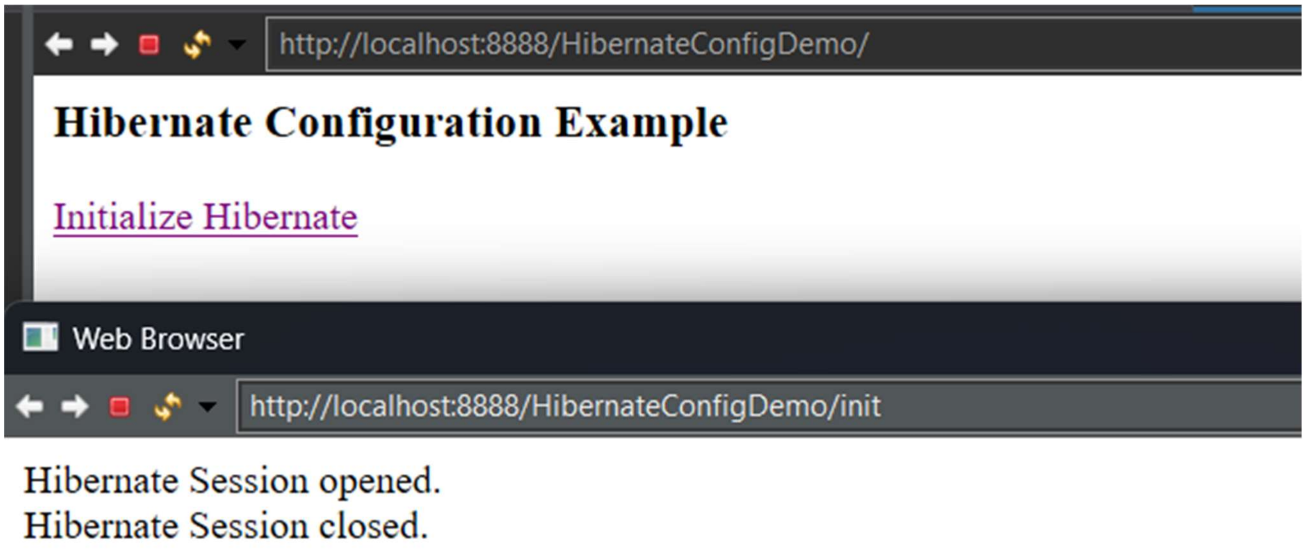
```java
package hibernateConfig;
import org.hibernate.SessionFactory;
import org.hibernate.boot.*;
import org.hibernate.boot.registry.*;

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) {
                        throw new ExceptionInInitializerError(th);
                }
        }

        public static SessionFactory getSessionFactory() {
                return sessionFactory;
        }

}
```

InitDemo Servlet

```
package hibernateConfig;
import java.io.*;
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 org.hibernate.*;
@WebServlet("/init")
public class InitDemo extends HttpServlet {
        private static final long serialVersionUID = 1L;
        protected void doGet(HttpServletRequest request, HttpServletResponse
response)
                        throws ServletException, IOException {

                PrintWriter out = response.getWriter();
                out.println("<html><body>");

                // STEP 1: Get a Session (connection) from the Session Factory
class
                SessionFactory factory = HibernateUtil.getSessionFactory();

                Session session = factory.openSession();

                out.println("Hibernate Session opened.<br>");

                session.close();

                out.println("Hibernate Session closed.<br>");

                // STEP 2 execute the HQL commands
                // for now we will only test if the connection is establised with
MySQL server.

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

OUTPUT


http://localhost:8888/HibernateConfigDemo/

Hibernate Configuration Example

Initialize Hibernate

Web Browser

http://localhost:8888/HibernateConfigDemo/init

```
Hibernate Session opened.
Hibernate Session closed.
```

2.Configure Hibernate using XML in Eclipse IDE.

Index.html

```
<br> <h3> Hibernate Query Demo</h3>
<a href="HibernateQueryDemo"> Hibernate Query Demo</a><br>
```

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"/>
    </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">8143303511@Sri</property>

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


</hibernate-configuration>
```

HibernateUtil class

```
package hibernateConfig;
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) {
			throw new ExceptionInInitializerError(th);
		}
	}

	public static SessionFactory getSessionFactory() {
		return sessionFactory;
	}

}
```

Eproduct class

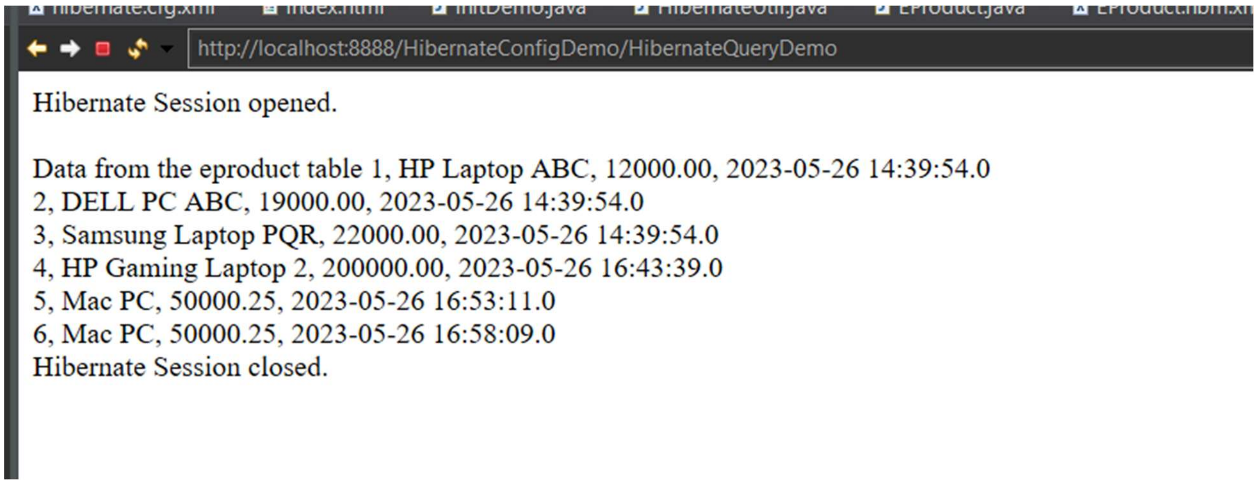
```
package com.ecommerce;
import java.math.BigDecimal;
import java.util.Date;

public class EProduct {

	private long ID;
	private String name;
	private BigDecimal price;
	private Date dateAdded;

	public EProduct() {

	}

	public long getID() {
		return ID;
	}

	public void setID(long iD) {
		ID = iD;
	}

	public String getName() {
		return name;
	}

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

	public BigDecimal getPrice() {
		return price;
	}

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

	public Date getDateAdded() {
		return dateAdded;
	}

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

}
```

HibernateQueryDemo servlet

```
package hibernateConfig;
import java.io.*;
import java.util.List;

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

import org.hibernate.*;

import com.ecommerce.EProduct;

@WebServlet("/HibernateQueryDemo")
public class HibernateQueryDemo extends HttpServlet {
	private static final long serialVersionUID = 1L;

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

```
            PrintWriter out = response.getWriter();
            out.println("<html><body>");

            // STEP 1: Get a Session (connection) from the Session Factory
class
            SessionFactory factory = HibernateUtil.getSessionFactory();

            Session session = factory.openSession();

            out.println("Hibernate Session opened.<br>");

            // STEP 2 execute the HQL commands
            // for now we will only test if the connection is establised with
MySQL server.
            List<EProduct> eproducts = session.createQuery("from
EProduct").list();
out.println("<br> Data from the eproduct table");
            for(EProduct prod: eproducts) {
                  out.println(prod.getID() + ", " + prod.getName() + ",  "
            + prod.getPrice() + ",   "  + prod.getDateAdded() + "<br>" );
            }


            session.close();

            out.println("Hibernate Session closed.<br>");

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

}
```

Output



http://localhost:8888/HibernateConfigDemo/HibernateQueryDemo

```
Hibernate Session opened.

Data from the eproduct table 1, HP Laptop ABC, 12000.00, 2023-05-26 14:39:54.0
2, DELL PC ABC, 19000.00, 2023-05-26 14:39:54.0
3, Samsung Laptop PQR, 22000.00, 2023-05-26 14:39:54.0
4, HP Gaming Laptop 2, 200000.00, 2023-05-26 16:43:39.0
5, Mac PC, 50000.25, 2023-05-26 16:53:11.0
6, Mac PC, 50000.25, 2023-05-26 16:58:09.0
Hibernate Session closed.
```

3.Configure Hibernate using Annotations in Eclipse IDE.

Hibernate.cfg.xml

```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">8143303511@Sri</property>

     <mapping class="com.ecommerce.EProduct" />
  </session-factory>


</hibernate-configuration>
```

Index.html

```html
<br> <h3> Hibernate Annonated Query Demo</h3>
<a href="HibernateQueryDemo"> Hibernate Query Demo</a><br>
```

HibernateUtil class

```java
package hibernateConfig;
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) {
                    throw new ExceptionInInitializerError(th);
             }
      }

      public static SessionFactory getSessionFactory() {
             return sessionFactory;
      }

}
```

Eproduct class

```
package com.ecommerce;

import java.math.BigDecimal;
import java.util.Date;

import javax.persistence.*;

@Entity
@Table(name= "eproduct")
public class EProduct {
      @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 EProduct() {

      }
public long getID() {
             return ID;
      }

      public void setID(long iD) {
             ID = iD;
      }
```

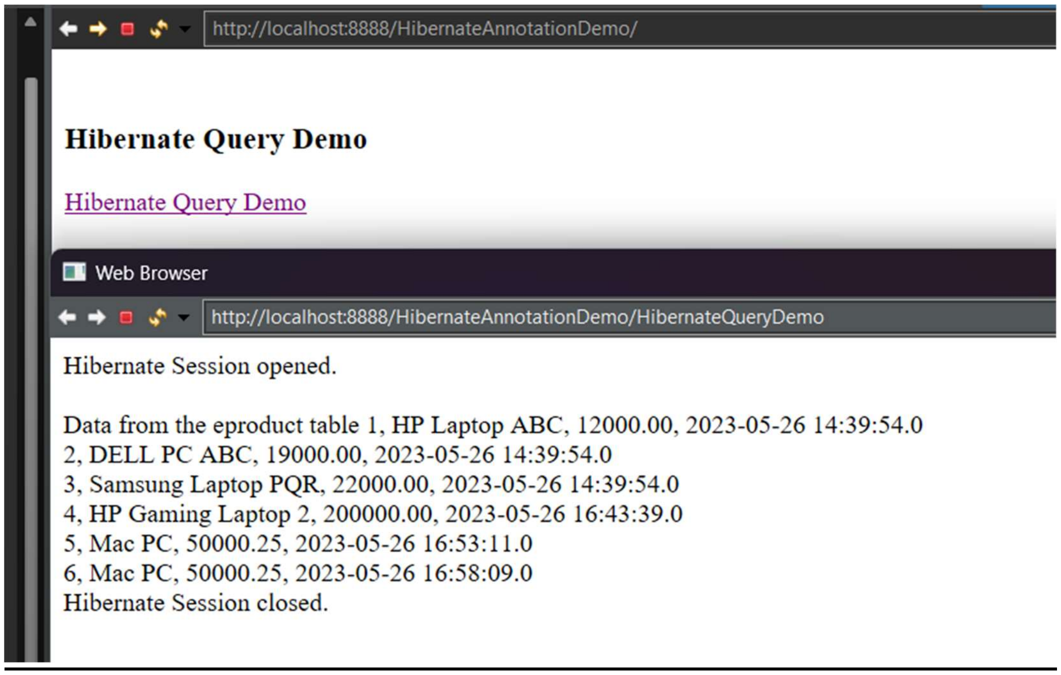
```
	public String getName() {
		return name;
	}

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

	public BigDecimal getPrice() {
		return price;
	}

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

	public Date getDateAdded() {
		return dateAdded;
	}

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

}
```

HibernateQueryDemo servlet

```
package hibernateConfig;
import java.io.*;
import java.util.List;

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

import org.hibernate.*;

import com.ecommerce.EProduct;

@WebServlet("/HibernateQueryDemo")
public class HibernateQueryDemo extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		PrintWriter out = response.getWriter();
		out.println("<html><body>");

		// STEP 1: Get a Session (connection) from the Session Factory
class
		SessionFactory factory = HibernateUtil.getSessionFactory();
```

```
            Session session = factory.openSession();

            out.println("Hibernate Session opened.<br>");

            // STEP 2 execute the HQL commands
            // for now we will only test if the connection is establised with
MySQL server.
            List<EProduct> eproducts = session.createQuery("from
EProduct").list();
out.println("<br> Data from the eproduct table");
            for(EProduct prod: eproducts) {
                  out.println(prod.getID() + ", " + prod.getName() + ",  "
            + prod.getPrice() + ",  "  + prod.getDateAdded() + "<br>" );
            }

            session.close();

            out.println("Hibernate Session closed.<br>");

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

}
```

OUTPUT


http://localhost:8888/HibernateAnnotationDemo/
Web Browser
http://localhost:8888/HibernateAnnotationDemo/HibernateQueryDemo

Hibernate Query Demo

Hibernate Query Demo

```
Hibernate Session opened.

Data from the eproduct table 1, HP Laptop ABC, 12000.00, 2023-05-26 14:39:54.0
2, DELL PC ABC, 19000.00, 2023-05-26 14:39:54.0
3, Samsung Laptop PQR, 22000.00, 2023-05-26 14:39:54.0
4, HP Gaming Laptop 2, 200000.00, 2023-05-26 16:43:39.0
5, Mac PC, 50000.25, 2023-05-26 16:53:11.0
6, Mac PC, 50000.25, 2023-05-26 16:58:09.0
Hibernate Session closed.
```

4.Demonstrate Hibernate logging by Log4j.

Index.html

```
<br> <h3> Hibernate Query Demo</h3>
<a href="HibernateQueryDemo"> Hibernate Query Demo</a><br>
```

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"/>
    </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">8143303511@Sri</property>

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


</hibernate-configuration>
```

HibernateUtil class

```
package hibernateConfig;
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) {
			throw new ExceptionInInitializerError(th);
		}
	}

	public static SessionFactory getSessionFactory() {
		return sessionFactory;
	}

}
```

Eproduct class

```
package com.ecommerce;
import java.math.BigDecimal;
import java.util.Date;

public class EProduct {

	private long ID;
	private String name;
	private BigDecimal price;
	private Date dateAdded;

	public EProduct() {

	}

	public long getID() {
		return ID;
	}

	public void setID(long iD) {
		ID = iD;
	}

	public String getName() {
		return name;
	}

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

	public BigDecimal getPrice() {
		return price;
	}

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

	public Date getDateAdded() {
		return dateAdded;
	}

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

}
```

HibernateQueryDemo servlet

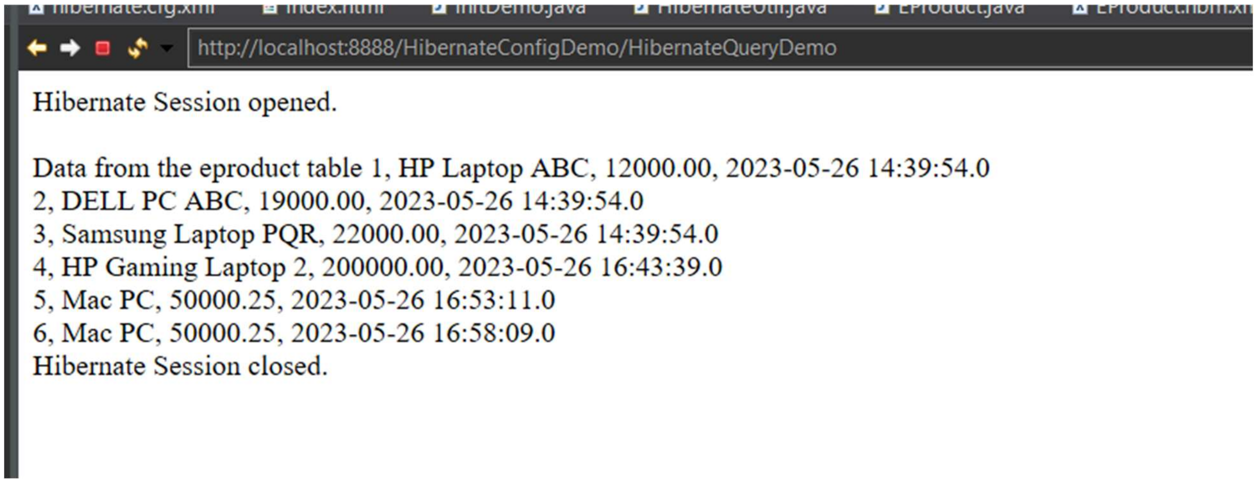
```
package hibernateConfig;
import java.io.*;
import java.util.List;

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

import org.hibernate.*;

import com.ecommerce.EProduct;

@WebServlet("/HibernateQueryDemo")
public class HibernateQueryDemo extends HttpServlet {
	private static final long serialVersionUID = 1L;

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

```
            PrintWriter out = response.getWriter();
            out.println("<html><body>");

            // STEP 1: Get a Session (connection) from the Session Factory
class
            SessionFactory factory = HibernateUtil.getSessionFactory();

            Session session = factory.openSession();

            out.println("Hibernate Session opened.<br>");

            // STEP 2 execute the HQL commands
            // for now we will only test if the connection is establised with
MySQL server.
            List<EProduct> eproducts = session.createQuery("from
EProduct").list();
out.println("<br> Data from the eproduct table");
            for(EProduct prod: eproducts) {
                  out.println(prod.getID() + ", " + prod.getName() + ",  "
            + prod.getPrice() + ",  "  + prod.getDateAdded() + "<br>" );
            }


            session.close();

            out.println("Hibernate Session closed.<br>");

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

}
```

log4j.properties

```
log4j.rootLogger=DEBUG, Appender1,Appender2

log4j.appender.Appender1=org.apache.log4j.ConsoleAppender
log4j.appender.Appender1.layout=org.apache.log4j.PatternLayout
log4j.appender.Appender1.layout.ConversionPattern=%-7p %d [%t] %c %x - %m%n

log4j.appender.Appender2=org.apache.log4j.FileAppender
log4j.appender.Appender2.File=D:\Softwares\applog.txt
log4j.appender.Appender2.layout=org.apache.log4j.PatternLayout
log4j.appender.Appender2.layout.ConversionPattern=%-7p %d [%t] %c %x - %m%n


# Log everything. Good for troubleshooting
log4j.logger.org.hibernate=INFO

# Log all JDBC parameters
log4j.logger.org.hibernate.type=ALL
```

Output

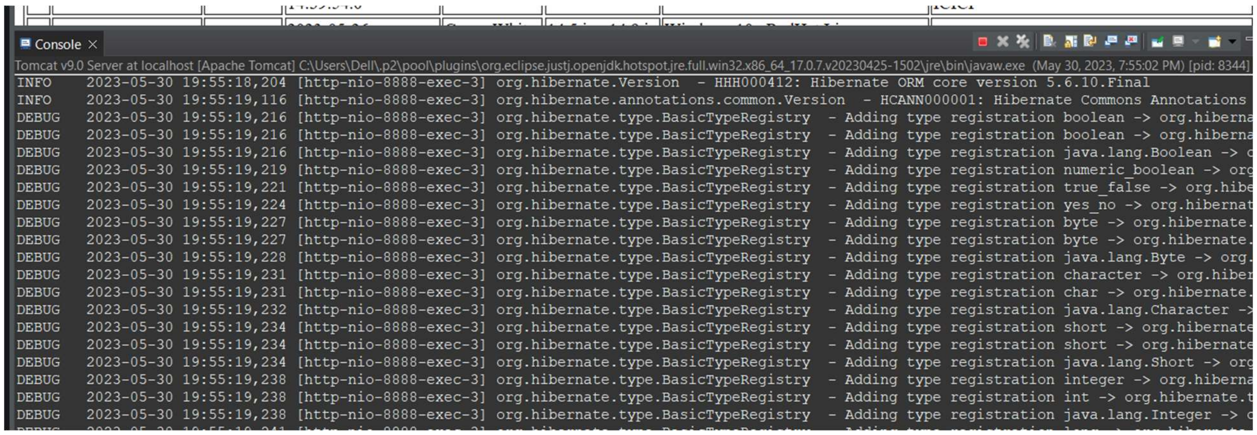


hibernate.cfg.xml   index.html   InitDemo.java   HibernateUtil.java   EProduct.java   EProduct.hbm.xml

http://localhost:8888/HibernateConfigDemo/HibernateQueryDemo

```
Hibernate Session opened.

Data from the eproduct table 1, HP Laptop ABC, 12000.00, 2023-05-26 14:39:54.0
2, DELL PC ABC, 19000.00, 2023-05-26 14:39:54.0
3, Samsung Laptop PQR, 22000.00, 2023-05-26 14:39:54.0
4, HP Gaming Laptop 2, 200000.00, 2023-05-26 16:43:39.0
5, Mac PC, 50000.25, 2023-05-26 16:53:11.0
6, Mac PC, 50000.25, 2023-05-26 16:58:09.0
Hibernate Session closed.
```

Logs output: -



```
Tomcat v9.0 Server at localhost [Apache Tomcat] C:\Users\Dell\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe  (May 30, 2023, 7:55:02 PM) [pid: 8344]
INFO    2023-05-30 19:55:18,204 [http-nio-8888-exec-3] org.hibernate.Version  - HHH000412: Hibernate ORM core version 5.6.10.Final
INFO    2023-05-30 19:55:19,116 [http-nio-8888-exec-3] org.hibernate.annotations.common.Version  - HCANN000001: Hibernate Commons Annotations
DEBUG   2023-05-30 19:55:19,216 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration boolean -> org.hiberna
DEBUG   2023-05-30 19:55:19,216 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration boolean -> org.hiberna
DEBUG   2023-05-30 19:55:19,216 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration java.lang.Boolean -> o
DEBUG   2023-05-30 19:55:19,219 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration numeric_boolean -> org
DEBUG   2023-05-30 19:55:19,221 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration true_false -> org.hibe
DEBUG   2023-05-30 19:55:19,224 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration yes_no -> org.hibernat
DEBUG   2023-05-30 19:55:19,227 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration byte -> org.hibernate.
DEBUG   2023-05-30 19:55:19,227 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration byte -> org.hibernate.
DEBUG   2023-05-30 19:55:19,228 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration java.lang.Byte -> org.
DEBUG   2023-05-30 19:55:19,231 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration character -> org.hiber
DEBUG   2023-05-30 19:55:19,231 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration char -> org.hibernate.
DEBUG   2023-05-30 19:55:19,232 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration java.lang.Character ->
DEBUG   2023-05-30 19:55:19,234 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration short -> org.hibernate
DEBUG   2023-05-30 19:55:19,234 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration short -> org.hibernate
DEBUG   2023-05-30 19:55:19,234 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration java.lang.Short -> org
DEBUG   2023-05-30 19:55:19,238 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration integer -> org.hiberna
DEBUG   2023-05-30 19:55:19,238 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration int -> org.hibernate.t
DEBUG   2023-05-30 19:55:19,238 [http-nio-8888-exec-3] org.hibernate.type.BasicTypeRegistry  - Adding type registration java.lang.Integer -> o
```

5.Demonstrate mapping List, Set, Bag, and Map in collection using XML file.

index.html

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

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

<br> <h3> Hibernate Query Demo</h3>
<a href="HibernateQueryDemo"> Hibernate Query Demo</a><br>


<br> <h3> Hibernate Mapping  Demo</h3>
<a href="product-details"> Hibernate Mapping Demo</a><br>
```

hiberbate.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">8143303511@Sri</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>
```

HibernateQueryDemo servlet

```
package hibernateConfig;
import java.io.*;
import java.util.List;

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

import org.hibernate.*;

import com.ecommerce.EProduct;

@WebServlet("/HibernateQueryDemo")
public class HibernateQueryDemo extends HttpServlet {
      private static final long serialVersionUID = 1L;

      protected void doGet(HttpServletRequest request, HttpServletResponse 
response)
                  throws ServletException, IOException {
            PrintWriter out = response.getWriter();
            out.println("<html><body>");

            // STEP 1: Get a Session (connection) from the Session Factory 
class
            SessionFactory factory = HibernateUtil.getSessionFactory();

            Session session = factory.openSession();

            out.println("Hibernate Session opened.<br>");

            // STEP 2 execute the HQL commands
            // for now we will only test if the connection is establised with 
MySQL server.
            List<EProduct> eproducts = session.createQuery("from 
EProduct").list();
out.println("<br> Data from the eproduct table");
            for(EProduct prod: eproducts) {
                  out.println(prod.getID() + ", " + prod.getName() + ",  " 
            + prod.getPrice() + ",  "  + prod.getDateAdded() + "<br>" );
            }


            session.close();

            out.println("Hibernate Session closed.<br>");

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

}
```

HibernateUtil class

```
package hibernateConfig;
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) {
			throw new ExceptionInInitializerError(th);
		}
	}

	public static SessionFactory getSessionFactory() {
		return sessionFactory;
	}

}
```

InitDemo Servlet

```
package hibernateConfig;
import java.io.*;
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 org.hibernate.*;
@WebServlet("/init")
public class InitDemo extends HttpServlet {
	private static final long serialVersionUID = 1L;
	protected void doGet(HttpServletRequest request, HttpServletResponse 
response)
			throws ServletException, IOException {

		PrintWriter out = response.getWriter();
		out.println("<html><body>");
```

```
            // STEP 1: Get a Session (connection) from the Session Factory
class
            SessionFactory factory = HibernateUtil.getSessionFactory();

            Session session = factory.openSession();

            out.println("Hibernate Session opened.<br>");

            session.close();

            out.println("Hibernate Session closed.<br>");

            // STEP 2 execute the HQL commands
            // for now we will only test if the connection is establised with
MySQL server.

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

Color class

```
package com.ecommerce;

public class Color {

      private long COLORID;
      private String name;

      public Color() {

      }

      public long getCOLORID() {
            return COLORID;
      }

      public void setCOLORID(long cOLORID) {
            COLORID = cOLORID;
      }

      public String getName() {
            return name;
      }

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

}
```

Color.hbm.xml

```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="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>
```

OS class

```java
package com.ecommerce;

public class OS {

	private long OSID;
	private String name;

	public OS() {

	}

	public OS(long oSID, String name) {
		super();
		OSID = oSID;
		this.name = name;
	}

	public long getOSID() {
		return OSID;
	}

	public void setOSID(long oSID) {
		OSID = oSID;
	}

	public String getName() {
		return name;
	}

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

}
```

OS.hbm.xml

```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="OS" table="os">
            <id name="OSID" type="long" column="ID">
                  <generator class="identity" />
            </id>
            <property name="name" type="string" column="NAME" />
      </class>
</hibernate-mapping>
```

ScreenSizes class

```java
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;}


}
```

ScreenSizes.hbm.xml

```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="ScreenSizes" table="screensizes">
```

```
        <id name="SCREENID" type="long" column="ID">
            <generator class="identity"/>
        </id>
        <property name="size" type="string" column="SIZE"/>
    </class>
</hibernate-mapping>
```

Finance class

```
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;}

}
```

Finance.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="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>
```

EProduct class

```
package com.ecommerce;

import java.math.BigDecimal;
import java.util.*;

public class EProduct {

	private long ID;
	private String name;
	private BigDecimal price;
	private Date dateAdded;

	private List<Color> colors;

	private Set<OS> os;

	private Collection<ScreenSizes> screenSizes;

	private Map finance;

	public EProduct() {

	}

	public long getID() {
		return ID;
	}

	public void setID(long iD) {
		ID = iD;
	}

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

	public BigDecimal getPrice() {
		return price;
	}

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

	public Date getDateAdded() {
		return dateAdded;
	}

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

	public List<Color> getColors() {
		return colors;
	}

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

	public Set<OS> getOs() {
		return os;
	}

	public void setOs(Set<OS> os) {
		this.os = os;
	}
public Collection<ScreenSizes> getScreenSizes() {
		return screenSizes;
	}

	public void setScreenSizes(Collection<ScreenSizes> screenSizes) {
		this.screenSizes = screenSizes;
	}

	public Map getFinance() {
		return finance;
	}

	public void setFinance(Map finance) {
		this.finance = finance;
	}

}
```

EProduct.hbm.xml

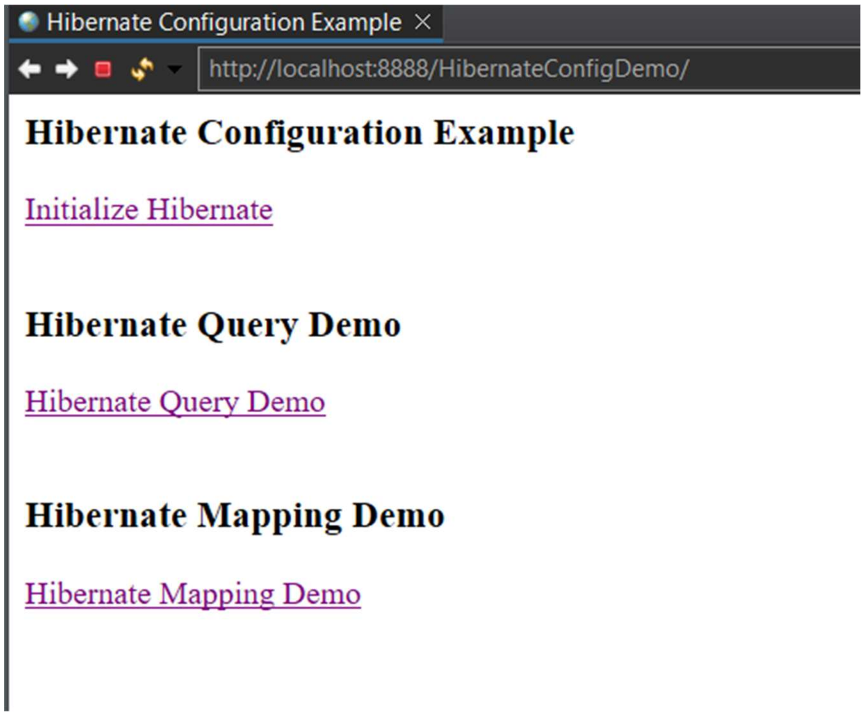
```xml
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.ecommerce">
      <class name="EProduct" table="eproduct">
            <id name="ID" column="ID">
                  <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" />

            <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="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>
```

ProductDetailsServlet

```
package hibernateConfig;
import java.io.IOException;
import java.io.PrintWriter;
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>");

		// STEP 1: Get a Session (connection) from the Session Factory class
		SessionFactory factory = HibernateUtil.getSessionFactory();
		// STEP 2 Session
		Session session = factory.openSession();

		// STEP 32 execute the HQL commands
		// for now we will only test if the connection is establised with MySQL server.
		List<EProduct> eproducts = session.createQuery("from EProduct").list();
out.println("<br> Data from the eproduct table<table border=1>");
		out.println("<th> ID <th> NAME <th> PRICE <th> DATE ADDED <th> COLORS <th> Screen
Sizes <th> OS  <th> FINANCE OPTIONS </th>   ");
		for (EProduct prod : eproducts) {

			// Display Core properties/details
			out.println("<tr><td>" + prod.getID() + "<td>" + prod.getName() + "<td>" +
prod.getPrice() + "<td>"
					+ prod.getDateAdded());

			// Display the available colors
			List<Color> colors = prod.getColors();
			out.println("<td> ");
			for(Color color: colors)
```

```
                                        out.println(color.getName() + " &nbsp;");

                              // Display the available screensizes
                              Collection<ScreenSizes> screenSizes = prod.getScreenSizes();
                              out.println("<td> " );
                              for(ScreenSizes sSize: screenSizes)
                                        out.println(sSize.getSize() + " &nbsp;");

                              // Display the available OSes
                              Set<OS> OSes = prod.getOs();
                              out.println("<td>  ");
                              for(OS os: OSes)
                                        out.println(os.getName() + " &nbsp;");
// Display the available finance options
                               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;");
         }



                    }

                    session.close();

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

          }

}
```

OUTPUT



Hibernate Configuration Example

http://localhost:8888/HibernateConfigDemo/

Hibernate Configuration Example

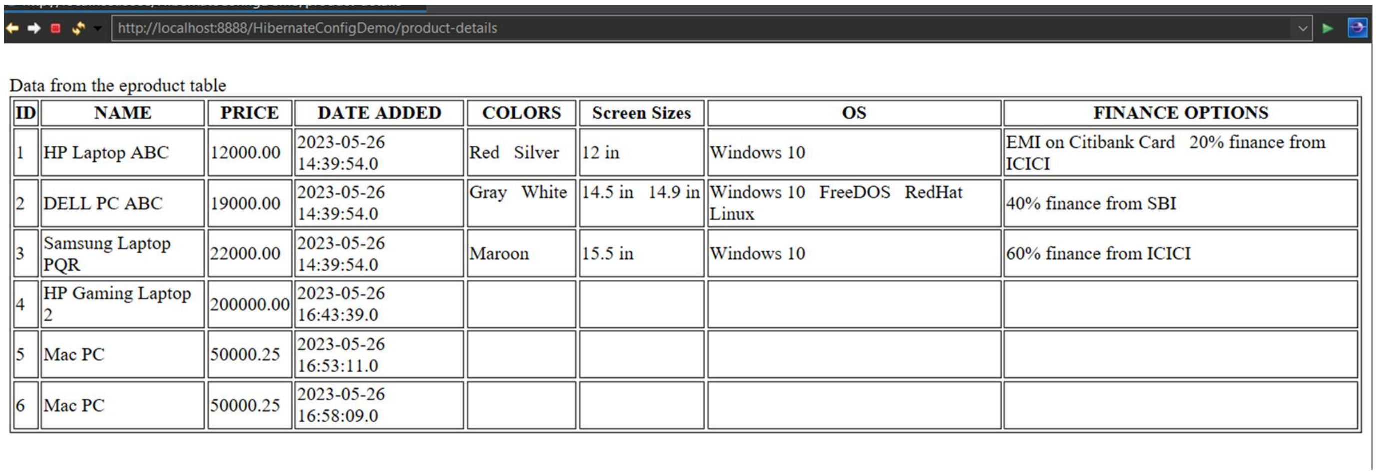
Initialize Hibernate

Hibernate Query Demo

Hibernate Query Demo

Hibernate Mapping Demo

Hibernate Mapping Demo



http://localhost:8888/HibernateConfigDemo/product-details

Data from the eproduct table

ID	NAME	PRICE	DATE ADDED	COLORS	Screen Sizes	OS	FINANCE OPTIONS
1	HP Laptop ABC	12000.00	2023-05-26 14:39:54.0	Red Silver	12 in	Windows 10	EMI on Citibank Card 20% finance from ICICI
2	DELL PC ABC	19000.00	2023-05-26 14:39:54.0	Gray White	14.5 in 14.9 in	Windows 10 FreeDOS RedHat Linux	40% finance from SBI
3	Samsung Laptop PQR	22000.00	2023-05-26 14:39:54.0	Maroon	15.5 in	Windows 10	60% finance from ICICI
4	HP Gaming Laptop 2	200000.00	2023-05-26 16:43:39.0				
5	Mac PC	50000.25	2023-05-26 16:53:11.0				
6	Mac PC	50000.25	2023-05-26 16:58:09.0				

6.Demonstrate lazy collection in Hibernate.

index.html

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

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

<br> <h3> Hibernate Query Demo</h3>
<a href="HibernateQueryDemo"> Hibernate Query Demo</a><br>


<br> <h3> Hibernate Mapping  Demo</h3>
<a href="product-details"> Hibernate Mapping Demo</a><br>
```

hiberbate.cfg.xml

```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">8143303511@Sri</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>
```

HibernateQueryDemo servlet

```java
package hibernateConfig;
import java.io.*;
import java.util.List;

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

import org.hibernate.*;

import com.ecommerce.EProduct;

@WebServlet("/HibernateQueryDemo")
public class HibernateQueryDemo extends HttpServlet {
      private static final long serialVersionUID = 1L;

      protected void doGet(HttpServletRequest request, HttpServletResponse
response)
                  throws ServletException, IOException {
            PrintWriter out = response.getWriter();
            out.println("<html><body>");

            // STEP 1: Get a Session (connection) from the Session Factory
class
            SessionFactory factory = HibernateUtil.getSessionFactory();

            Session session = factory.openSession();

            out.println("Hibernate Session opened.<br>");

            // STEP 2 execute the HQL commands
            // for now we will only test if the connection is establised with
MySQL server.
            List<EProduct> eproducts = session.createQuery("from
EProduct").list();
out.println("<br> Data from the eproduct table");
            for(EProduct prod: eproducts) {
                  out.println(prod.getID() + ", " + prod.getName() + ",  "
            + prod.getPrice() + ",  "  + prod.getDateAdded() + "<br>" );
            }


            session.close();

            out.println("Hibernate Session closed.<br>");

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

}
```

HibernateUtil class

```java
package hibernateConfig;
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) {
			throw new ExceptionInInitializerError(th);
		}
	}

	public static SessionFactory getSessionFactory() {
		return sessionFactory;
	}

}
```

InitDemo Servlet

```java
package hibernateConfig;
import java.io.*;
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 org.hibernate.*;
@WebServlet("/init")
public class InitDemo extends HttpServlet {
	private static final long serialVersionUID = 1L;
	protected void doGet(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {

		PrintWriter out = response.getWriter();
		out.println("<html><body>");

		// STEP 1: Get a Session (connection) from the Session Factory
class
```

```
            SessionFactory factory = HibernateUtil.getSessionFactory();

            Session session = factory.openSession();

            out.println("Hibernate Session opened.<br>");

            session.close();

            out.println("Hibernate Session closed.<br>");

            // STEP 2 execute the HQL commands
            // for now we will only test if the connection is establised with
MySQL server.

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

Color class

```
package com.ecommerce;

public class Color {

      private long COLORID;
      private String name;

      public Color() {

      }

      public long getCOLORID() {
            return COLORID;
      }

      public void setCOLORID(long cOLORID) {
            COLORID = cOLORID;
      }

      public String getName() {
            return name;
      }

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

}
```

Color.hbm.xml

```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="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>
```

OS class

```java
package com.ecommerce;

public class OS {

	private long OSID;
	private String name;

	public OS() {

	}

	public OS(long oSID, String name) {
		super();
		OSID = oSID;
		this.name = name;
	}

	public long getOSID() {
		return OSID;
	}

	public void setOSID(long oSID) {
		OSID = oSID;
	}

	public String getName() {
		return name;
	}

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

}
```

OS.hbm.xml

```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="OS" table="os">
            <id name="OSID" type="long" column="ID">
                  <generator class="identity" />
            </id>
            <property name="name" type="string" column="NAME" />
      </class>
</hibernate-mapping>
```

ScreenSizes class

```
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;}


}
```

ScreenSizes.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="ScreenSizes" table="screensizes">
        <id name="SCREENID" type="long" column="ID">
            <generator class="identity"/>
        </id>
        <property name="size" type="string" column="SIZE"/>
    </class>
```

```
</hibernate-mapping>
```

Finance class

```
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;}

}
```

Finance.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="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>
```

EProduct class

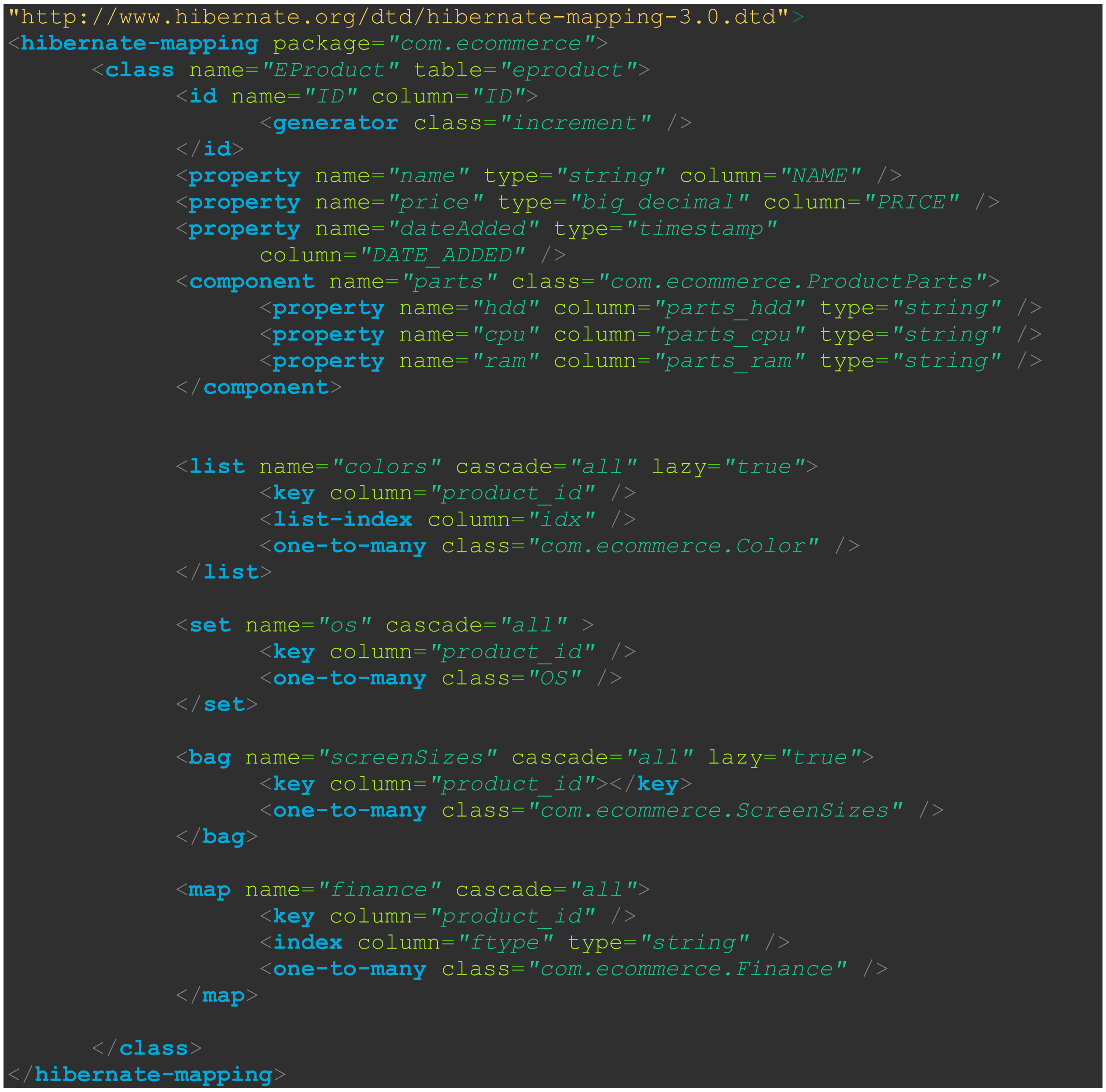
```
package com.ecommerce;
```

```
import java.math.BigDecimal;
import java.util.*;

public class EProduct {

	private long ID;
	private String name;
	private BigDecimal price;
	private Date dateAdded;

	private List<Color> colors;

	private Set<OS> os;

	private Collection<ScreenSizes> screenSizes;

	private Map finance;

	public EProduct() {

	}

	public long getID() {
		return ID;
	}

	public void setID(long iD) {
		ID = iD;
	}

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

	public BigDecimal getPrice() {
		return price;
	}

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

	public Date getDateAdded() {
		return dateAdded;
```

```
        }

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

        public List<Color> getColors() {
                return colors;
        }

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

        public Set<OS> getOs() {
                return os;
        }

        public void setOs(Set<OS> os) {
                this.os = os;
        }
public Collection<ScreenSizes> getScreenSizes() {
                return screenSizes;
        }

        public void setScreenSizes(Collection<ScreenSizes> screenSizes) {
                this.screenSizes = screenSizes;
        }

        public Map getFinance() {
                return finance;
        }

        public void setFinance(Map finance) {
                this.finance = finance;
        }

}
```

EProduct.hbm.xml

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD 3.0//EN"
```

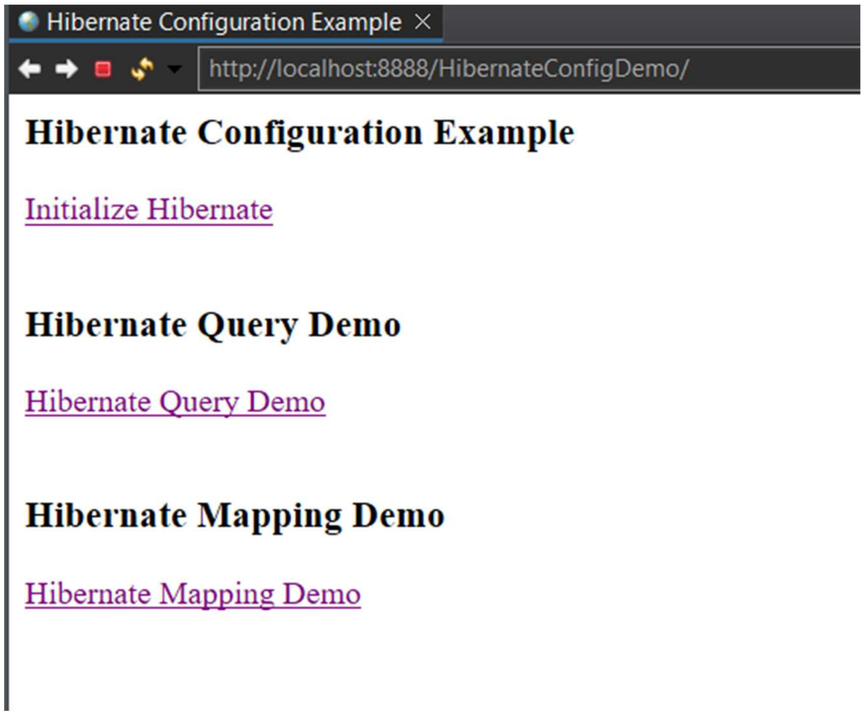


```xml
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.ecommerce">
      <class name="EProduct" table="eproduct">
            <id name="ID" column="ID">
                  <generator class="increment" />
            </id>
            <property name="name" type="string" 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" lazy="true">
                  <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="OS" />
            </set>

            <bag name="screenSizes" cascade="all" lazy="true">
                  <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>
```

ProductDetailsServlet

```
package hibernateConfig;
import java.io.IOException;
import java.io.PrintWriter;
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>");

                // STEP 1: Get a Session (connection) from the Session Factory class
                SessionFactory factory = HibernateUtil.getSessionFactory();
                // STEP 2 Session
                Session session = factory.openSession();

                // STEP 32 execute the HQL commands
                // for now we will only test if the connection is establised with MySQL server.
                List<EProduct> eproducts = session.createQuery("from EProduct").list();
out.println("<br> Data from the eproduct table<table border=1>");
                out.println("<th> ID <th> NAME <th> PRICE <th> DATE ADDED <th> COLORS <th> Screen
Sizes <th> OS  <th> FINANCE OPTIONS </th>   ");
                for (EProduct prod : eproducts) {

                        // Display Core properties/details
                        out.println("<tr><td>" + prod.getID() + "<td>" + prod.getName() + "<td>" +
prod.getPrice() + "<td>"
                                        + prod.getDateAdded());

                        // Display the available colors
                        List<Color> colors = prod.getColors();
                        out.println("<td> ");
                        for(Color color: colors)
                                out.println(color.getName() + " &nbsp;");

                        // Display the available screensizes
                        Collection<ScreenSizes> screenSizes = prod.getScreenSizes();
                        out.println("<td> " );
                        for(ScreenSizes sSize: screenSizes)
                                out.println(sSize.getSize() + " &nbsp;");

                        // Display the available OSes
                        Set<OS> OSes = prod.getOs();
                        out.println("<td>  ");
                        for(OS os: OSes)
                                out.println(os.getName() + " &nbsp;");
// Display the available finance options
                         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;");
        }



                }

                session.close();

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

         }

}
```

OUTPUT



Hibernate Configuration Example

http://localhost:8888/HibernateConfigDemo/

Hibernate Configuration Example

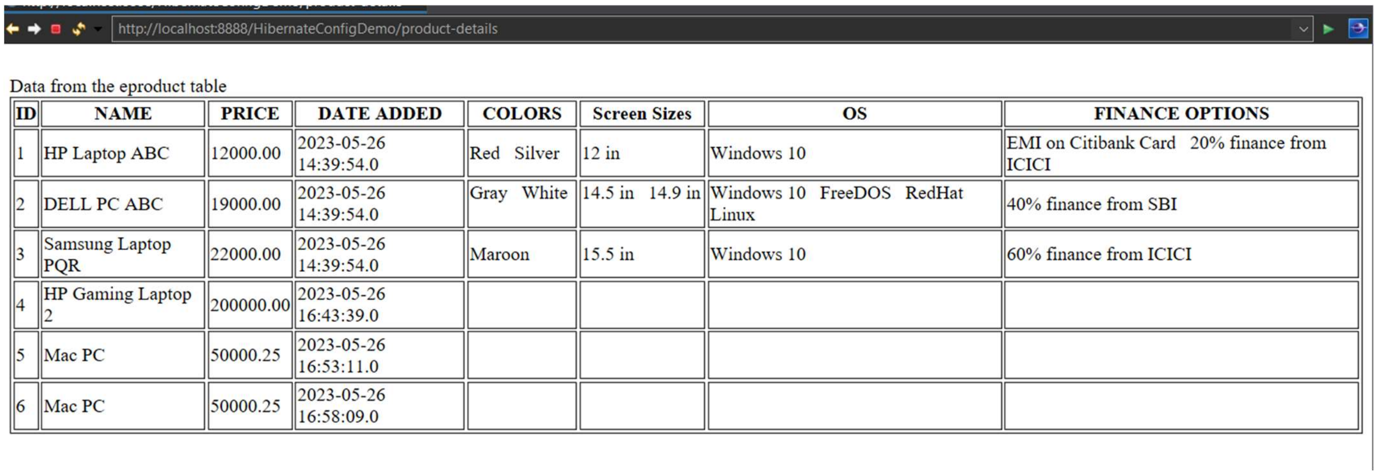
Initialize Hibernate

Hibernate Query Demo

Hibernate Query Demo

Hibernate Mapping Demo

Hibernate Mapping Demo



http://localhost:8888/HibernateConfigDemo/product-details

Data from the eproduct table

ID	NAME	PRICE	DATE ADDED	COLORS	Screen Sizes	OS	FINANCE OPTIONS
1	HP Laptop ABC	12000.00	2023-05-26 14:39:54.0	Red Silver	12 in	Windows 10	EMI on Citibank Card 20% finance from ICICI
2	DELL PC ABC	19000.00	2023-05-26 14:39:54.0	Gray White	14.5 in 14.9 in	Windows 10 FreeDOS RedHat Linux	40% finance from SBI
3	Samsung Laptop PQR	22000.00	2023-05-26 14:39:54.0	Maroon	15.5 in	Windows 10	60% finance from ICICI
4	HP Gaming Laptop 2	200000.00	2023-05-26 16:43:39.0				
5	Mac PC	50000.25	2023-05-26 16:53:11.0				
6	Mac PC	50000.25	2023-05-26 16:58:09.0				

7.Demonstrate component mapping in Hibernate.

Index.html

```
<br> <h3> Hibernate Component Mapping  Demo</h3>
<a href="component-mapping-demo"> Hibernate Component Mapping Demo</a><br>
```

Productparts class

```
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;}


}
```

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.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"/>

  </session-factory>

</hibernate-configuration>
```

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" 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"/>


                <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>

    </class>

</hibernate-mapping>
```

HibernateUtil class

```
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;
        }
}
```

Eproduct class

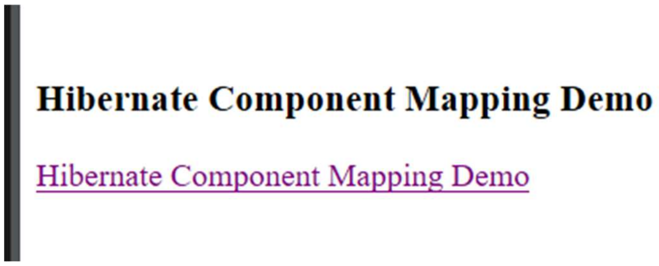
```
package com.ecommerce;

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;}
}
```

ComponentMappingServlet

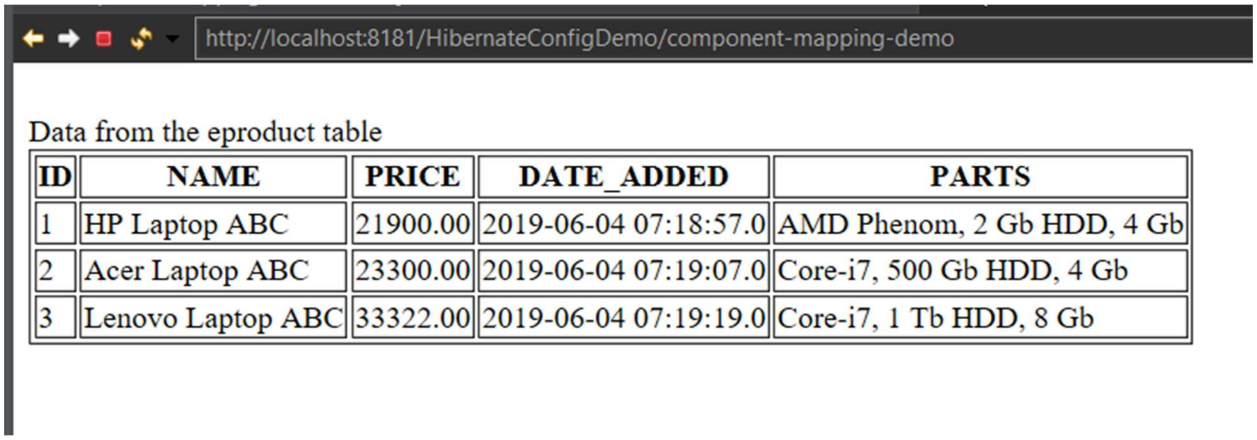
```
package hibernateConfig;
import java.io.*;
import java.util.*;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import org.hibernate.*;
import com.ecommerce.*;

@WebServlet("/component-mapping-demo")
public class ComponentMappingDemoServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;

	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 2 execute the HQL commands
		// for now we will only test if the connection is establised with MySQL server.
		List<EProduct> eproducts = session.createQuery("from EProduct").list();
out.println("<br> Data from the eproduct table<table
border=1><th>ID<th>NAME<th>PRICE<th>DATE_ADDED<th>PARTS");
		for (EProduct prod : eproducts) {
			out.println("<tr><td>" + prod.getID() + "<td>" + prod.getName() + "<td>" +
prod.getPrice() + "<td>"
					+ prod.getDateAdded());

			// Component class info (Product parts)
			ProductParts parts = prod.getParts();

			out.println("<td>" + parts.getCpu() + ", " + parts.getHdd() + ", " +
parts.getRam());

		}

		session.close();

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

}
```

OUTPUT



Hibernate Component Mapping Demo

Hibernate Component Mapping Demo



http://localhost:8181/HibernateConfigDemo/component-mapping-demo

Data from the eproduct table

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

8.Demonstrate integration of Hibernate with spring.

pom.xml

```
<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>
```

```xml
</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>
```

```xml
<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>
```

```xml
<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>
```

```xml
      <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>
```

EProductEntity class

```java
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;}
```

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

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

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

}
```

EProductDAO

```java
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();

    }

}
```

EProductController

```
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";

        }

}
```

eproduct-servlet.xml

```
<?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.ReloadableResourceBundleMessageS
ource">

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

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

  </bean>

  <bean id="propertyConfigurer"
```

```xml
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>
```

jdbc.properties

```
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
```

productList.jsp

```
<%@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>
```

index.jsp

```
<html>

<body>

<h2>Spring With Hibernate</h2>

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

</body>

</html>
```

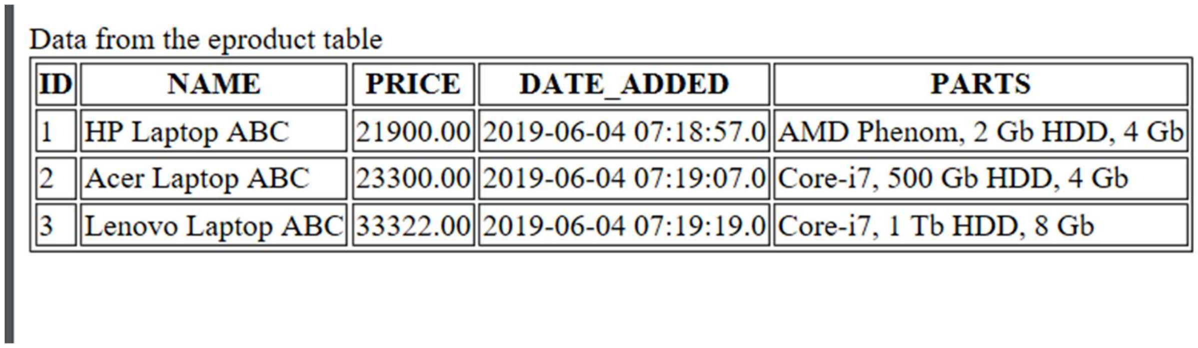
hibernate.cfg.xml

```xml
<?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>
```

web.xml

```xml
<!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>
```

OUTPUT



Data from the eproduct table

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