Project structure

Изображение выглядит как текст

Автоматически созданное описание

<?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>HibernateListMapping</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>AddProduct</servlet-name>  
 <servlet-class>com.ecommerce.web.AddProduct</servlet-class>  
 </servlet>  
 <servlet-mapping>  
 <servlet-name>AddProduct</servlet-name>  
 <url-pattern>/addproduct</url-pattern>  
 </servlet-mapping>  
  
 <servlet>  
 <servlet-name>ShowAllProducts</servlet-name>  
 <servlet-class>com.ecommerce.web.ShowAllProducts</servlet-class>  
 </servlet>  
 <servlet-mapping>  
 <servlet-name>ShowAllProducts</servlet-name>  
 <url-pattern>/showlistofproducts</url-pattern>  
 </servlet-mapping>  
  
</web-app>

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Error</title>  
</head>  
<body>  
<h1>  
 <left>  
 <span style='color:red'> Invalid properties entered. Try again.</span>  
 </left>  
</h1>  
</body>  
</html>

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="UTF-8">  
 <title>E-commerce application</title>  
</head>  
<body>  
<h1 align=center>Product adding page</h1>  
<center>  
 <form method=post action="addproduct">  
 Specify product properties and click "Submit" <p>  
 <table width="50%" border="0">  
 <tr valign="top">  
 <td width="40%">Name of product</td>  
 <td><input type="text" name="username" id="username" maxlength=50></td>  
 </tr>  
 <tr valign="top">  
 <td width="40%">Price of product</td>  
 <td><input type="text" name="price" id="price" maxlength=30></td>  
 </tr>  
 <tr valign="top">  
 <td colspan=2 width="100%">  
 <button>Submit</button>  
 </td>  
  
 </tr>  
 </table>  
 </form>  
  
 <a href="showlistofproducts">Click here to see list of all products</a>  
</center>  
</body>  
</html>

<%@ **page** import="java.util.\*" %>  
<%@ **page** import="java.util.Iterator" %>  
<%@ **page** import="com.ecommerce.model.EProduct" %>  
<%@ **page** contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>  
<!DOCTYPE html>  
<html>  
<head>  
 <title>Title</title>  
</head>  
<body>  
<h1 align=center>List of all products in database</h1>  
<p>  
<%  
List products = (List)request.getAttribute("products");  
for (Object product : products) {  
 out.print("<br>" + (EProduct)product);  
}  
%>  
</body>  
</html>

<?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?useSSL=false</property>  
 <property name="connection.username">root</property>  
 <property name="connection.password">vestaeagle</property>  
  
 <mapping resource="hiber/EProduct.hbm.xml"/>  
  
 <mapping class="com.ecommerce.model.EProduct"/>  
 </session-factory>  
</hibernate-configuration>

<?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="com.ecommerce.model.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"/>  
 </class>  
</hibernate-mapping>

package com.ecommerce.web;  
  
import com.ecommerce.model.EProduct;  
import org.hibernate.Session;  
import org.hibernate.SessionFactory;  
  
import javax.servlet.\*;  
import javax.servlet.http.\*;  
import javax.servlet.annotation.\*;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.util.List;  
  
@WebServlet(name = "ShowAllProducts", value = "/ShowAllProducts")  
public class ShowAllProducts extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 SessionFactory factory = HibernateUtil.getSessionFactory();  
 Session session = factory.openSession();  
  
 List<EProduct> result = session.createQuery("from EProduct").list();  
 session.close();  
  
 request.setAttribute("products", result);  
 RequestDispatcher view =  
 request.getRequestDispatcher("productlist.jsp");  
 view.forward(request, response);  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 }  
}

package com.ecommerce.web;  
  
import com.ecommerce.model.EProduct;  
import org.hibernate.Session;  
import org.hibernate.SessionFactory;  
import org.hibernate.Transaction;  
  
import javax.servlet.\*;  
import javax.servlet.http.\*;  
import javax.servlet.annotation.\*;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.math.BigDecimal;  
  
@WebServlet(name = "AddProduct", value = "/AddProduct")  
public class AddProduct extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
  
  
  
 try {  
 String name=request.getParameter("username");  
 BigDecimal price=new BigDecimal(request.getParameter("price"));  
 if (name.equals("")||price==BigDecimal.*ZERO*) throw new Exception();  
  
 SessionFactory factory = HibernateUtil.*getSessionFactory*();  
 Session session = factory.openSession();  
  
 Transaction transaction=session.beginTransaction();  
 EProduct eProduct=new EProduct(name,price);  
 session.save(eProduct);  
 transaction.commit();  
  
 session.close();  
  
 RequestDispatcher requestDispatcher=request.getRequestDispatcher("index.html");  
 requestDispatcher.forward(request,response);  
 }  
  
 catch (Exception e) {  
 RequestDispatcher requestDispatcher=request.getRequestDispatcher("error.html");  
 requestDispatcher.forward(request,response);  
 }  
  
 }  
}

package com.ecommerce.web;  
  
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*;  
 }  
}

package com.ecommerce.model;  
  
import lombok.Getter;  
import lombok.NoArgsConstructor;  
import lombok.Setter;  
import lombok.ToString;  
  
  
import java.math.BigDecimal;  
import java.util.\*;  
  
@Getter  
@Setter  
@NoArgsConstructor  
@ToString  
public class EProduct {  
 private long ID;  
 private String name;  
 private BigDecimal price;  
 private Date dateAdded;  
  
 public EProduct(String name, BigDecimal price) {  
 this.name = name;  
 this.price = price;  
 this.dateAdded = new Date();  
 }  
}