**Product.java** (model class):

**package** beans;

**import** javax.persistence.\*;

@Entity

@Table(name="Address")

**public** **class** Address {

@Id

@GeneratedValue(strategy = GenerationType.***AUTO***)

**private** **int** addressId;

@Column(length = 20)

**private** String addressLine1, city, state, country;

@Column

**private** **int** pincode;

@OneToOne(targetEntity = EmployeeDetails.**class**)

**private** EmployeeDetails employee;

**public** EmployeeDetails getEmp() {

**return** employee;

}

**public** EmployeeDetails getEmployee() {

**return** employee;

}

**public** **void** setEmployee(EmployeeDetails emp) {

**this**.employee = emp;

}

**public** **int** getAddressId() {

**return** addressId;

}

**public** **void** setAddressId(**int** addressId) {

**this**.addressId = addressId;

}

**public** String getAddressLine1() {

**return** addressLine1;

}

**public** **void** setAddressLine1(String addressLine1) {

**this**.addressLine1 = addressLine1;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** String getState() {

**return** state;

}

**public** **void** setState(String state) {

**this**.state = state;

}

**public** String getCountry() {

**return** country;

}

**public** **void** setCountry(String country) {

**this**.country = country;

}

**public** **int** getPincode() {

**return** pincode;

}

**public** **void** setPincode(**int** pincode) {

**this**.pincode = pincode;

}

}

2.Add deatil in class

package beans;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.OneToOne;

import javax.persistence.PrimaryKeyJoinColumn;

import javax.persistence.Table;

@Entity

@Table(name="EmployeeDetails")

public class EmployeeDetails {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private int employeeId;

@Column(length=20)

private String name, email;

@OneToOne(targetEntity=Address.class,cascade=CascadeType.ALL)

private Address address;

public Address getAddress() {

return address;

}

public void setAddress(Address address) {

this.address = address;

}

public int getEmployeeId() {

return employeeId;

}

public void setEmployeeId(int employeeId) {

this.employeeId = employeeId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

3. **AddProductServlet.java** (servlet for adding a product):

package Servlet;

import java.io.File;

import java.io.IOException;

import java.io.PrintWriter;

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.Session;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

import beans.Address;

import beans.EmployeeDetails;

@WebServlet("/HibernateTestServlet")

public class HibernateTestServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

int register(EmployeeDetails user){

SessionFactory sessionFactory = new Configuration().configure().buildSessionFactory();

Session session = sessionFactory.openSession();

session.beginTransaction();

int i=(Integer)session.save(user);

session.getTransaction().commit();

session.close();

return i;

}

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

EmployeeDetails e1=new EmployeeDetails();

e1.setName("Kartik");

e1.setEmail("Kartik@gmail.com");

Address address1=new Address();

address1.setAddressLine1("F-9, ABC nagar");

address1.setCity("Ghaziabad");

address1.setState("UP");

address1.setCountry("India");

address1.setPincode(201301);

//to set relationship pass address object to employee and employee object to address

e1.setAddress(address1);

address1.setEmployee(e1);

int i=register(e1);

PrintWriter out=response.getWriter();

if(i>0)

out.println("Record Inserted");

else

out.println("Record not Inserted");

}

}

4**index.jsp** (HTML form for adding a product):

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<a href=*"HibernateTestServlet"*>Click</a>

</body>

</html>

5. **web.xml** (web deployment descriptor for the servlet mapping):

<!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>

<servlet>

<servlet-name>HibernateTestServlet</servlet-name>

<display-name>HibernateTestServlet</display-name>

<description></description>

<servlet-class>Servlet.HibernateTestServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>HibernateTestServlet</servlet-name>

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

</servlet-mapping>

</web-app>

6. Configure Hibernate:

Create the **hibernate.cfg.xml** file in the **src/main/resources** directory. Here's an example configuration for MySQL:

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

<!--

~ Hibernate, Relational Persistence for Idiomatic Java

~

~ License: GNU Lesser General Public License (LGPL), version 2.1 or later.

~ See the lgpl.txt file in the root directory or <http://www.gnu.org/licenses/lgpl-2.1.html>.

-->

<!DOCTYPE hibernate-configuration PUBLIC

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

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

<hibernate-configuration>

<session-factory>

<property name=*"format\_sql"*>false</property>

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

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

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

<property name=*"connection.password"*>passw0rd</property>

<!-- JDBC connection pool (use the built-in) -->

<property name=*"connection.pool\_size"*>4</property>

<!-- SQL dialect -->

<property name=*"dialect"*>org.hibernate.dialect.MySQL5Dialect</property>

<!-- Disable the second-level cache -->

<property name=*"cache.provider\_class"*>org.hibernate.cache.internal.NoCacheProvider</property>

<!-- Echo all executed SQL to stdout -->

<property name=*"show\_sql"*>true</property>

<!-- Drop and re-create the database schema on startup -->

<property name=*"hbm2ddl.auto"*>create</property>

<!-- <property name="hbm2ddl.auto">update</property> -->

<mapping class=*"beans.EmployeeDetails"*/>

<mapping class=*"beans.Address"*/>

</session-factory>

</hibernate-configuration>

7.Maven