**Adding a New Product in the Database**

**Step 1:** Creating a Maven project

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

**Step 2:** Adding the jar files for Hibernate and its dependencies

* Add dependencies to pom.xml file and save it.
* Right click on **ServletWithHibernate**, select maven, then select update and give forceful update
* The required jar files for your projects are automatically downloaded and saved in your maven project as maven dependencies.

**Step 3:** Configuring Hibernate with hibernate.cfg.xml

* In the Project Explorer, expand **ServletWithHibernate ->Java Resources**
* Right click on **src** and choose **New->Other**
* Select **General->File** and click **Next**
* In filename, enter **hibernate.cfg.xml** and click on **Finish**

**Step 4:** Creating a table in Mysql

* In the Mysql, create the table namely products with the fields pid,pname,price.

**Step 5:** Creating an HTML page index.html

* In the Project Explorer, expand the project **ServletWithHibernate**
* Expand **WebContent**. Right click on **WebContent**. Choose **New->HTML File**
* Enter the filename as **index.html** and click on **Finish**
* Create the form to get the user inputs as pid,pname,price.

**Step 6:** Creating a ProductDao class to initiate Hibernate in code

* In the Project Explorer, expand **ServletWithHibernate ->Java Resources**
* Right click on **src** and choose **New->Class**
* In Package Name, enter **com.dao** and in **Name** enter **ProductDao** and click on **Finish**
* Inside the method namely storeProducts, create the configuration to load the xml file and to save the records within try and catch block.

**Step 7:** Creating an ProductController servlet

* In the Project Explorer, expand **ServletWithHibernate ->Java Resources**
* Right click on **src** and choose **New->Servlet**
* In **Class Name,** enter **ProductController** and click on **Finish**
* In doPost, create printWriter and get the inputs from index.html
* Then create an object for Products class(entity class) and set the values
* Then create the object for ProductService class to print the result whether the record is inserted or not based on some business logics.

**Step 8:** Creating a Products class to initiate the variables

* In the Project Explorer, expand **ServletWithHibernate ->Java Resources**
* Right click on **src** and choose **New->Class**
* In Package Name, enter **com.bean** and in **Name** enter **Products** and click on **Finish**
* Here , import @Entity to map the table and @Id to specify the primary key.
* Add getter and setter methods to the variables and add String to String method.

**Step 9:** Creating a ProductService class to write the business logic

* In the Project Explorer, expand **ServletWithHibernate ->Java Resources**
* Right click on **src** and choose **New->Class**
* In Package Name, enter **com.service** and in **Name** enter **ProductService** and click on **Finish**
* Add the business logic , that the record must be greater than one then insert the record and print “record is inserted”. Otherwise, print “record is not inserted”.

**Step 10:** Configuring web.xml

* In the Project Explorer, expand **ServletWithHibernate ->WebContent->WEB-INF**
* Double click on **web.xml** to open it in the editor

**Step 11:** Checking for servlet-api.jar

* Before building the project, we need to add **servlet-api.jar** to the project

**Step 12:** Building the project

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

**Step 13:** Publishing and starting the project

* If you do not see the **Servers** tab near the bottom of the IDE, go to **Window** menu and click **Show View->Servers**
* Right click on the **Server** entry and choose **Add and Remove**
* Click the **Add** button to move **ServletWithHibernate** from the **Available** list to the **Configured** list
* Click on **Finish**
* Right click on the **Server** entry and click on **Publish**
* Right click on the **Server** entry and click on **Start.**
* This will start the server

**Step 14:** Running the project

* To run the project, open a web browser and type:

[**http://localhost:8080/**](http://localhost:8080/ServletConcept) **ServletWithHibernate**