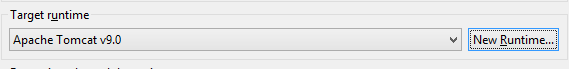
# Data Centric RAD

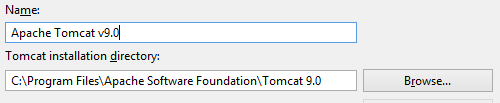
## Lab 11 Part 1 JSF with Datatables

### Setting up a JSF Project

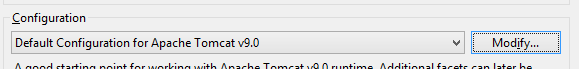
* Open Eclipse Java EE IDE for Web Developers.
* Select *File/New/Other/Web/Dynamic Web Project*.
* Enter a project name and select the Target runtime as shown:



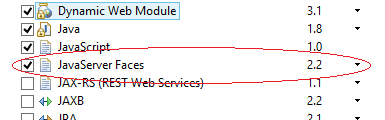
* + If there are no options, press *New Runtime*
  + Click on Apache Tomcat v9.0 and press *Next*
  + Browse to the Apache Tomcat installation directory and press *Finish*



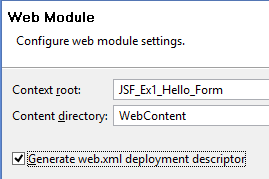
* Press *Modify* in the Configuration section:



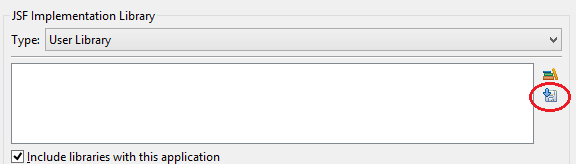
* On the *Project Facets* window select JavaServer Faces 2.2 and press *OK*.



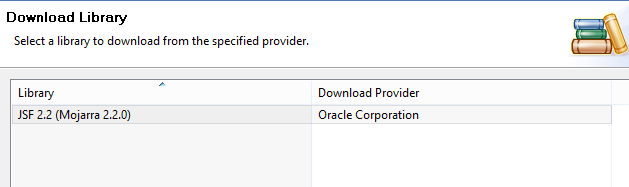
* Press *Next* until the *Web Module* screen appears, select the *Generate web.xml deployment descriptor* and press *Next*.



* On the *JSF Capabilities* screen, press the *Download library* button.



* Select the *JSF 2.2* screen, then press the *Next* followed by *Finish*.



* Press *Finish* on the *JSF Capabilities* window.
* Download MySQL JDBC Driver from <https://dev.mysql.com/downloads/connector/j/> unzip it and put the Jar file in the WEB-INF/lib folder of your application.
* Create a new file called *context.xml* in the *WebContent/META-INF/* folder and add the following to it:

<Context>

<Resource name="jdbc/employeesdb14"

auth="Container" type="javax.sql.DataSource"

maxActive="20" maxIdle="5" maxWait="10000"

username="root" password=""

driverClassName="com.mysql.jdbc.Driver"

url="jdbc:mysql://localhost:3306/employeesdb14"/>

</Context>

* Add the following to the *web.xml* file in *WebContent/WEB-INF/*:

<resource-ref>

<description>EmployeesDB Datasource</description>

<res-ref-name>jdbc/employeesdb14</res-ref-name>

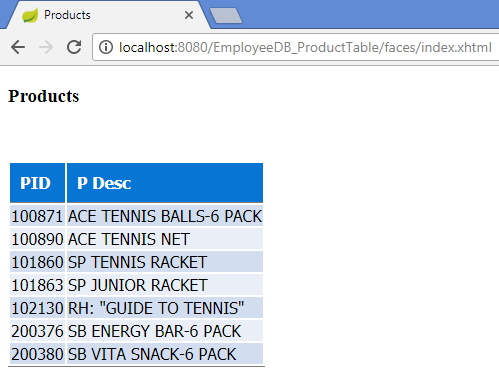
<res-type>javax.sql.DataSource</res-type>

<res-auth>Container</res-auth>

</resource-ref>

### Part 1 Product Details with Datatable

* Download and import *employeesDB14.sql* from Moodle.
* Write a JSF/JDBC application that queries the *product* table in the *employees* database and displays the first all products on a web page as follows.



* The following CSS can be used to style the table. CSS files live in the WebContent\resources\css folder.

.table {

border-bottom:1px solid gray;

font-family: Tahoma,Verdana,Segoe,sans-serif;

}

.table-header {

border-bottom:1px solid gray;

background:#0775d3;

margin-bottom: 22px;

padding:10px;

color: #FFFFFF;

text-align:left;

}

.table-odd-row {

border-top:1px solid #FFFFFF;

background:#D2DEEF;

text-align:left;

}

.table-even-row {

border-top:1px solid #FFFFFF;

background:none repeat scroll 0 0 #EAEFF7;

text-align:left;

}

* The style should be imported in the view as follows:

<h:outputStylesheet library=*"css"* name=*"tables.css"* />

And used in the datatable as follows:

<h:dataTable value=*"#{productController.products}"* var=*"p"*

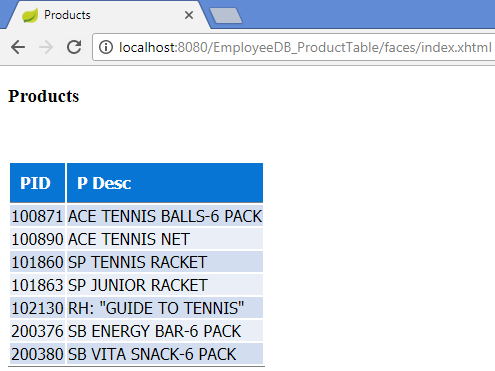
styleClass=*"table"*

headerClass=*"table-header"*

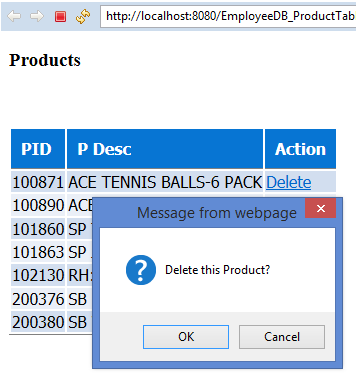
rowClasses=*"table-odd-row,table-even-row"*>

### Part 2 Product Details with Datatable and Delete

* Update Part 1 so that each product has a *Delete* option beside it;



* when the user presses *Delete* beside any product, a he/she is asked to confirm delete:



* If the user presses *Cancel*, nothing happens.
* If the user presses *OK*, the appropriate product is deleted from the database.
* Use this code to help you:

<h:column>

<f:facet name=*"header"*>Action</f:facet>

<h:commandLink value=*"Delete "*

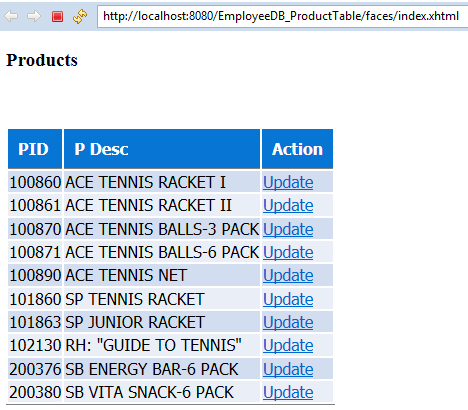
onclick="if (!confirm('Delete this Product?')) return false"

action=*"#{productController.deleteProduct(p.productID)}"*/>

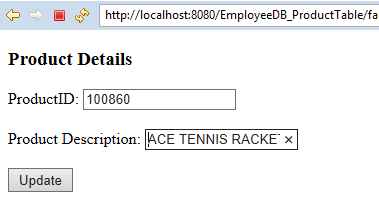
</h:column>

### Part 3 Product Details with Datatable and Update

* Update Part 1 so that each product has an *Update* option beside it;

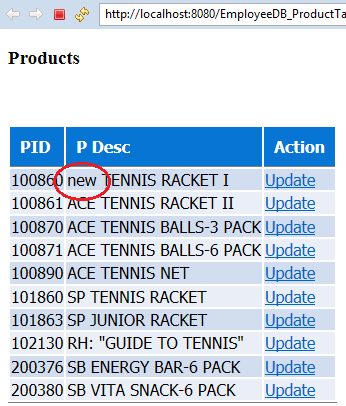


* When the user clicks the *Update* option for a particular product, he/she is brought to the following page, where the existing details of the product are shown and can be updated:



* On that page the product description only may be changed and when *Update* is pressed, the user is returned to the main page (*index.html*):





NOTE: To make an input field read only in JSF, add the following attribute to it in the view:

*readonly="#{facesContext.currentPhaseId.ordinal eq 6}"*