

# Start Your Project

Environment setup and template project

Notes on Web DB Programming

# Outline

- Notes on Web DB Programming
- Set up Environment and Introduction of the template project

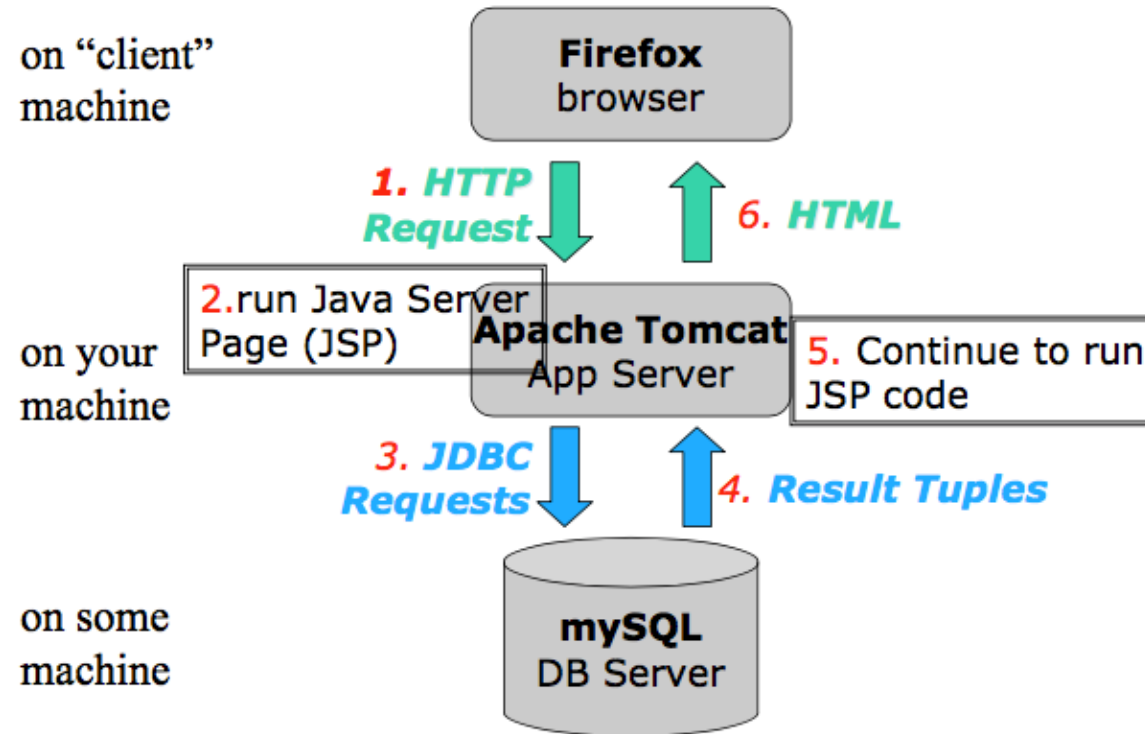
# Notes on Web DB Programming

Based on M.Muscari and UCSD (anon)

# Three-Tier Application

- **Presentation Tier** : user interface to make requests, provide input and see results
- **Middle Tier**: application logic
- **Data Management Tier**: database management

# Three-Tier architecture



# HTTP protocol

- Protocol that allows web servers and clients to **exchange data** over the web.
- It is a **request - response protocol**.
- Clients (web browsers) send requests to web servers
  - GET : ask for a resource
  - POST : send some data (e.g. HTML form)
- Server sends response
  - Status code (200 OK, 404 Not Found!)
- HTTP is a "*stateless*" protocol; each time a client retrieves a Webpage, the client opens a separate connection to the Web server and the server automatically does not keep any record of previous client request.

# Difference between GET/POST requests

## Anatomy of GET request



## Anatomy of POST request



# HTML (Hyper Text Markup Language)

- Standard markup language for creating web pages
  - Language for creating **structured documents**
    - It consists of elements which can be nested
  - The HTML standard specifies a number of universally supported elements (“tags”)
- Web browsers receive HTML documents from a webserver and render them into multimedia web pages.
- HTML is commonly delivered as part of an HTTP response

Tutorial: <https://www.w3schools.com/html/>



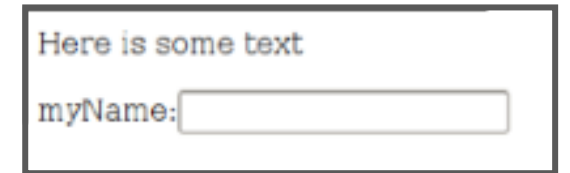
# HTML Tags

## ➤ Common HTML tags include:

- **<div></div>** - a logical division (section)
- **<p> </p>** - a paragraph
- **<table> </table>** -a table of values
  - **<tr> </tr>** - table row
  - **<td> </td>** -table column
- **<form></form>** - a form enclosing input fields
  - **<input></input>** - an input field

# HTML Example

```
<html>
  <body>
    <div>
      <p>Here is some text</p>
      <form action="submit.jsp" method="post">
        myName: <input name="myInput" type="text"/>
      </form>
    </div>
  </body>
</html>
```



Here is some text

myName:

# HTML Table Example

```
<html>
  <body>
    <table border='1'>
      <tr>
        <td>First Name</td>
        <td>Last Name</td>
      </tr>
      <tr>
        <td>Matt</td>
        <td>Muscari</td>
      </tr>
    </table>
  </body>
</html>
```

First Name	Last Name
Matt	Muscari

# JSP (Java Server Pages)

- A technology for building web applications that serve **dynamic content**
- A **JSP page** is a text document that contains two types of text:
  - **static data**, which can be expressed in any text-based format (e.g. HTML)
  - JSP elements, which construct **dynamic content**.
- The dynamic content in a **JSP page** is in specially marked Java code fragments (enclosed between **<%** and **%>**).
- To deploy and run JSPs, a compatible web server with a servlet container, such as Apache Tomcat is required.
- When executed, the Java code fragments usually generate additional HTML into the page (in our case either accessing the database or processing parameters passed to HTTP requests)
- At the end, the resulting HTML page is sent to the browser to be displayed.

# JSP Syntax

## ➤ Comment

`<%-- Comment--%>`

## ➤ Expression

`<%= Java expression %>`

**Expression** tag evaluates the **expression** placed in it, converts the result into String and send the result back to the client through response object.

e.g. `<p>Today is <%= new Date().toString(); %> </p>`

## ➤ Scriptlet

`<% java code fragment %>`

e.g. `<% person.getFirstName(); %>`

## ➤ Include

`<jsp:include page="relativeURL"/>`

# JSP Implicit Objects

Object	Class
request	<b>HttpServletRequest</b>
response	<b>HttpServletResponse</b>
session	<b>HttpSession</b>
out	<b>Writer</b>

# request

- `<%@ page language="java" contentType="text/html"%>`  
  `<html>`  
    `<head>`  
      `<title>RequestExamplePage</title>`  
    `</head>`  
    `<body>`  
      `<%`  
      `// Get the User's Name from the request`  
      `out.println("<b>Hello: " + request.getParameter("myInput") + "</b>");`  
      `%>`  
    `</body>`  
  `</html>`

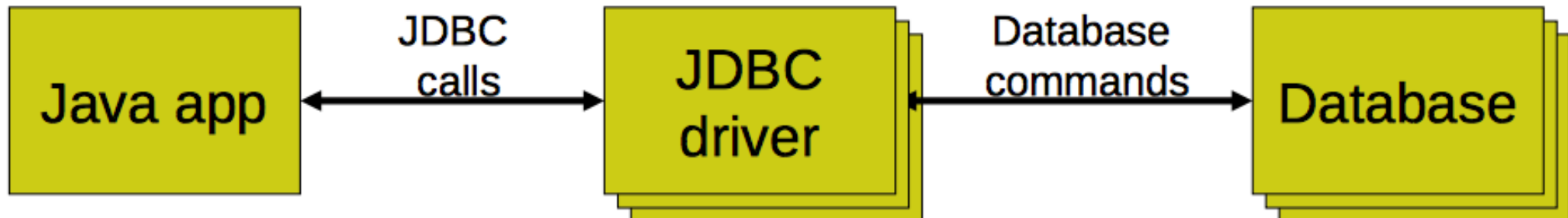
# session

- ```
<%@ page language="java" contentType="text/html"%>
<html>
    <head>
        <title>SessionExamplePage</title>
    </head>
    <body>
        <%
            HttpSession session = request.getSession(); //create a session object
            // Try and get the current count from the session
            Integer count = (Integer)session.getAttribute("COUNT");
            // If COUNT is not found, create it and add it to the session
            if ( count == null ) {
                count = new Integer(1);
                session.setAttribute("COUNT", count);
            } else {
                count = new Integer(count.intValue() + 1);
                session.setAttribute("COUNT", count);
            }
            // Print the number of times the user has visited the site
            out.println("<b>Hello you have visited this site: " + count + " times. </b>");
        %>
    </body> </html>
```



# Java Database Connectivity (JDBC)

- An **interface** to communicate with a relational database
  - Allows database agnostic Java code
  - Treat database tables/rows/columns as Java objects
- JDBC driver
  - An implementation of the JDBC interface
  - Communicates with a particular database



# JDBC steps

1. Connect to database
2. Query database (or insert/update/delete)
3. Process results
4. Close connection to database

# 1. Connect to database

## ➤ Load JDBC driver

- `Class.forName("com.mysql.jdbc.Driver").newInstance();`
- Make connection
  - `Connection conn = DriverManager.getConnection(url);` ?

## ➤ URL

- Format: “`jdbc:mysql://<hostname>:<port>/<databaseName>`”
- `jdbc:mysql://localhost:3036/BarBeerDrinkerSample`

## 2. Query database

### ➤ Create statement

- `Statement stmt = conn.createStatement();`
  - `stmt` object sends SQL commands to database
- Methods
  - `executeQuery()` for `SELECT` statements
  - `executeUpdate()` for `INSERT`, `UPDATE`, `DELETE`, statements

### ➤ Send SQL statements

- `stmt.executeQuery("SELECT ...");`
- `stmt.executeUpdate("INSERT ...");`

## 2. Query database

### ➤ Prepared Statements

- If you want to execute dynamic or parameterized SQL queries, use a “PreparedStatement” object instead of a statement.

```
PreparedStatement updateStud=conn.prepareStatement("UPDATE Student SET  
fname=? WHERE lastname LIKE?");
```

```
updateStud.setString(1,"John");  
updateStud.setString(2,"Doe");  
updateStud.executeUpdate();
```

# 3. Process results

➤ Result of a SELECT statement (rows/columns) returned as a **ResultSet** object

- `ResultSet rs = stmt.executeQuery("SELECT drinker,beer from LIKES");`

➤ Step through each row in the result

- `rs.next()`

➤ Get column values in a row

- `String userid = rs.getString("drinker");`
- `int type = rs.getInt("type");`

### 3. Process results

➤ Add a row to the users table

- `String str = "INSERT INTO LIKES VALUES('Bob', 'Corona')";`

➤ Returns number of rows in table

- `int rows = stmt.executeUpdate(str);`

## 4. Close connection to database

### ➤ Close the ResultSet object

- `rs.close();`

### ➤ Close the Statement object

- `stmt.close();`

### ➤ Close the connection

- `conn.close();`

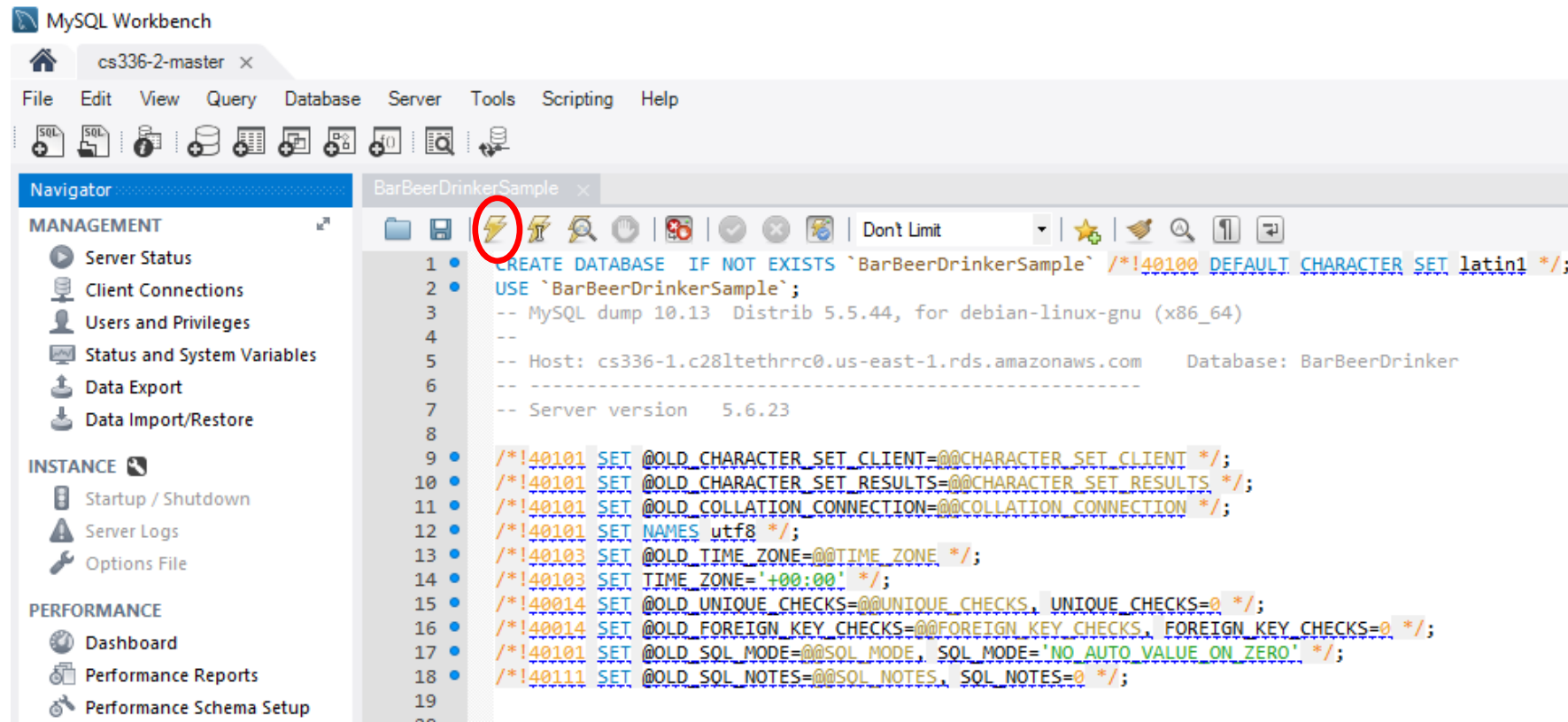


# Needed tools and installation

- **JRE, IDE** (JAVA, Eclipse for EE developers)
- **MySQL**
- **Apache Tomcat** (or any web server)
  - You will install it locally in your computer for development purposes.
- **JDBC**

# The Template Project

0. import schema **BarBeerDrinkerSample** in your DB instance using the provided script "BarBeerDrinkerSample.sql". Open the script and run it in your MySQLWorkbench. (File->Open SQL script)



# The Template Project

## 1. Download Eclipse IDE for **Java EE** Developers

<https://eclipse.org/downloads/eclipse-packages/>

The Eclipse Installer 2021-06 R now includes a JRE for macOS, Windows and Linux.

Try the Eclipse **Installer** 2021-06 R

The easiest way to install and update your Eclipse Development Environment.


[Find out more](#)

📦 764,959 Installer Downloads

**Download**

macOS [x86\\_64](#)  
Windows [x86\\_64](#)  
Linux [x86\\_64](#) | [AArch64](#)

### Eclipse IDE 2021-06 R Packages




**Eclipse IDE for Java Developers**

320 MB 461,764 DOWNLOADS

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration

Windows [x86\\_64](#)  
macOS [x86\\_64](#)  
Linux [x86\\_64](#) | [AArch64](#)



**Eclipse IDE for Enterprise Java and Web Developers**

517 MB 401,186 DOWNLOADS

Tools for developers working with Java and Web applications, including a Java IDE, tools for JavaScript, TypeScript, JavaServer Pages and Faces, Yaml, Markdown, Web Services, JPA and Data Tools, Maven and Gradle, Git, and more.

Click [here](#) to file a bug against Eclipse Web Tools Platform.  
Click [here](#) to file a bug against Eclipse Platform.  
Click [here](#) to file a bug against Maven integration for web projects.  
Click [here](#) to report an issue against Eclipse Wild Web Developer (incubating).

Windows [x86\\_64](#)  
macOS [x86\\_64](#)  
Linux [x86\\_64](#) | [AArch64](#)

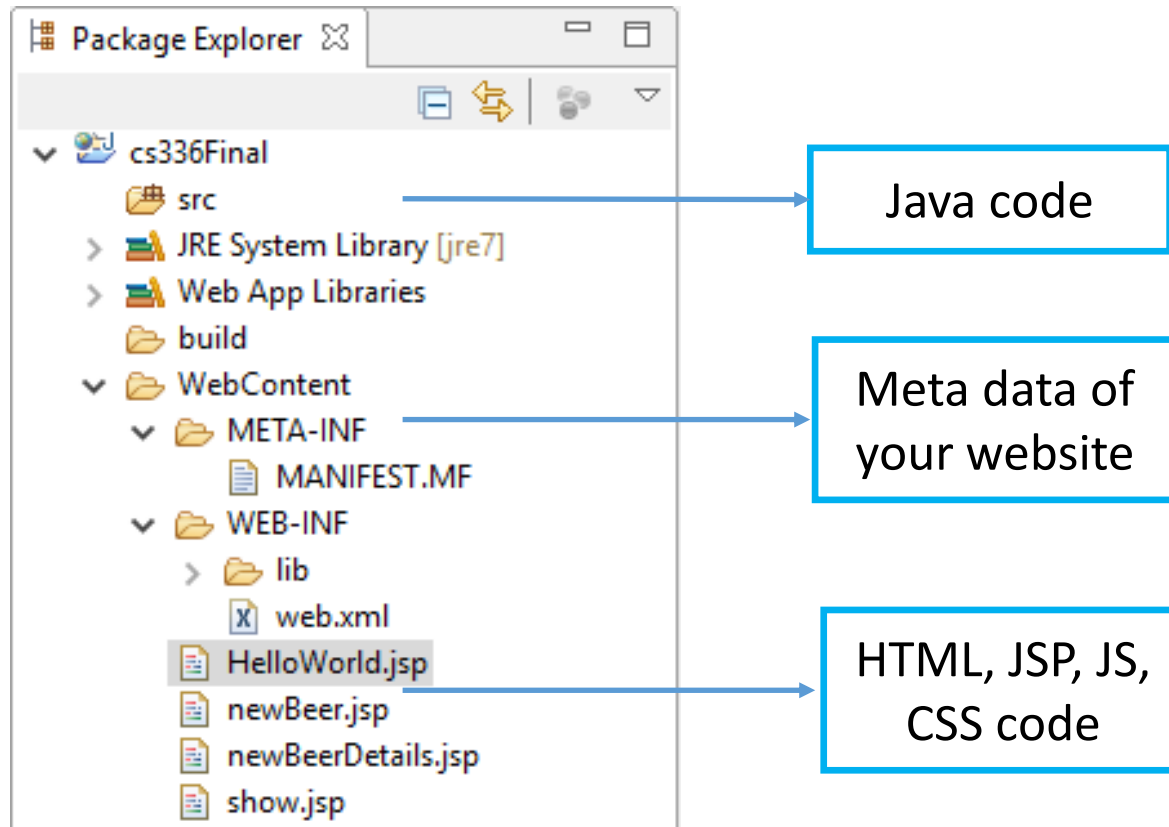
# The Template Project

**2.** Open eclipse and import the template project (cs336Sample.war)

File – Import – Web – WAR file

# The Template Project

## 3. Structure of the template



# The Template Project

## 4. Set your Tomcat server in eclipse

- If you don't have tomcat yet go to:

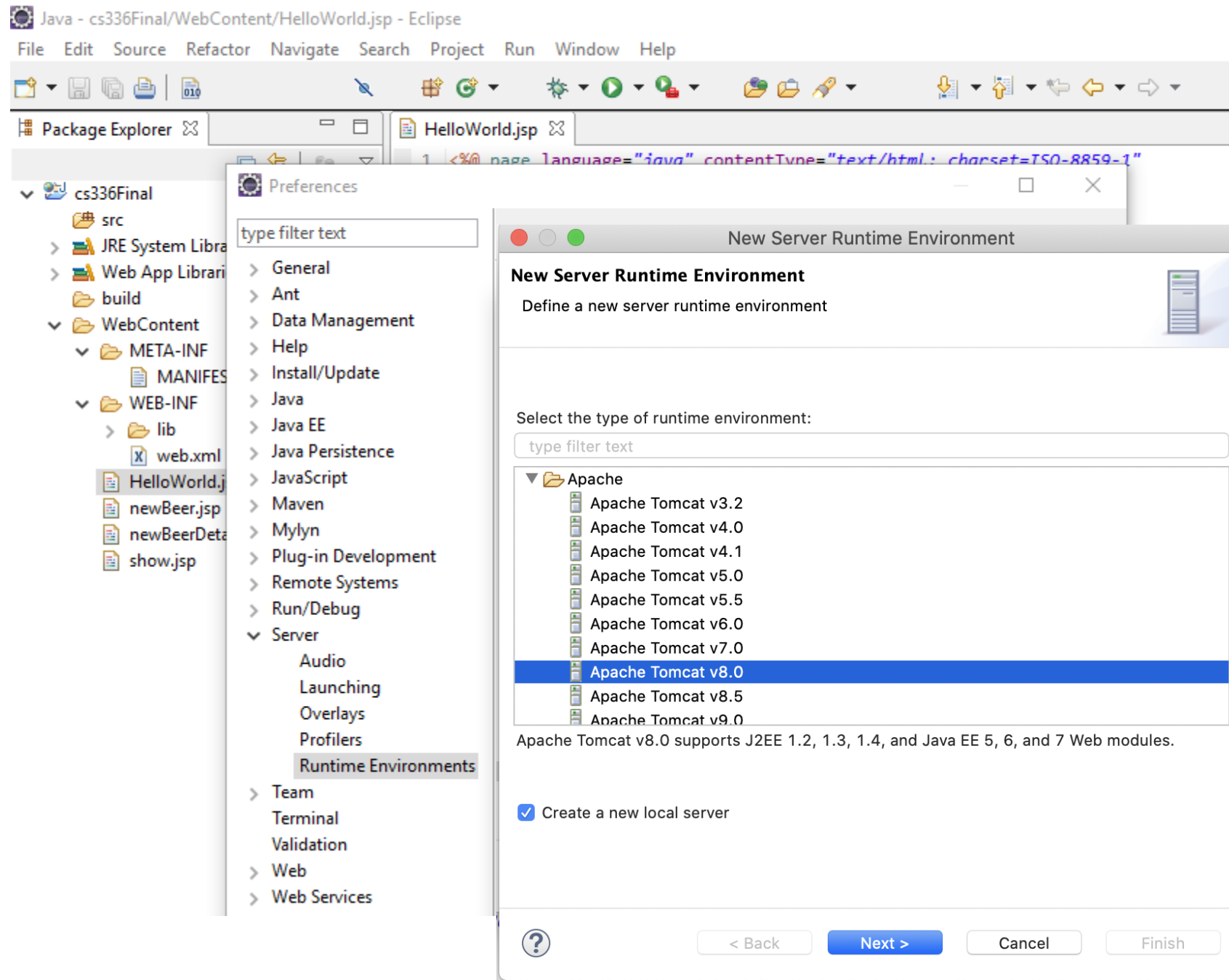
<https://tomcat.apache.org/download-80.cgi>

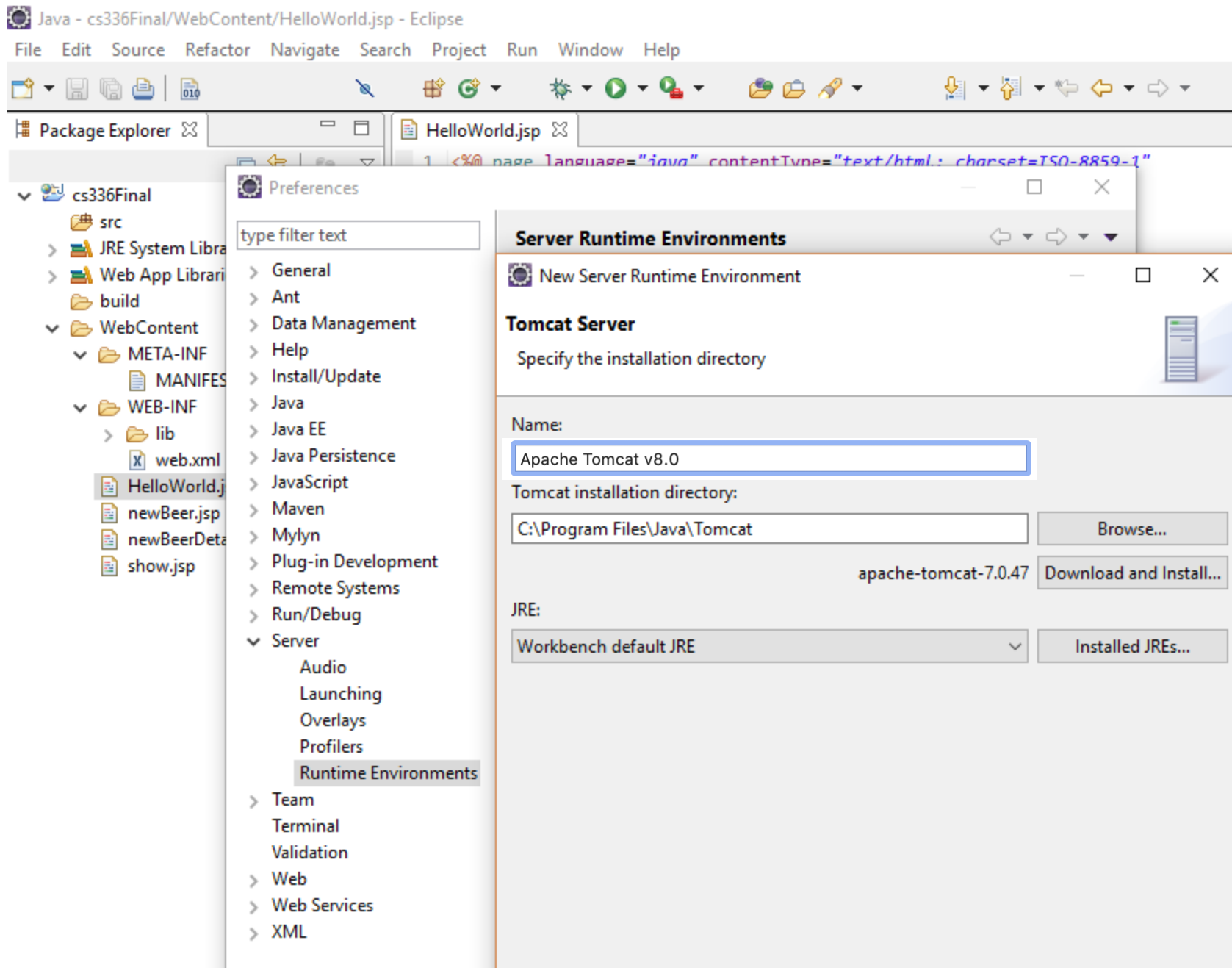
and download the binary distribution for your OS.

- After go back to eclipse:

Windows - Preference - Server - Runtime Environment - Add - Apache Tomcat v8.0 **or**

Eclipse- Preferences - Server - Runtime Environments - Add - Apache Tomcat v8.0







# The Template Project

## 5. Run the project based on Tomcat 8

Right click on the project - Run as - Run on Server - Apache – Tomcat8

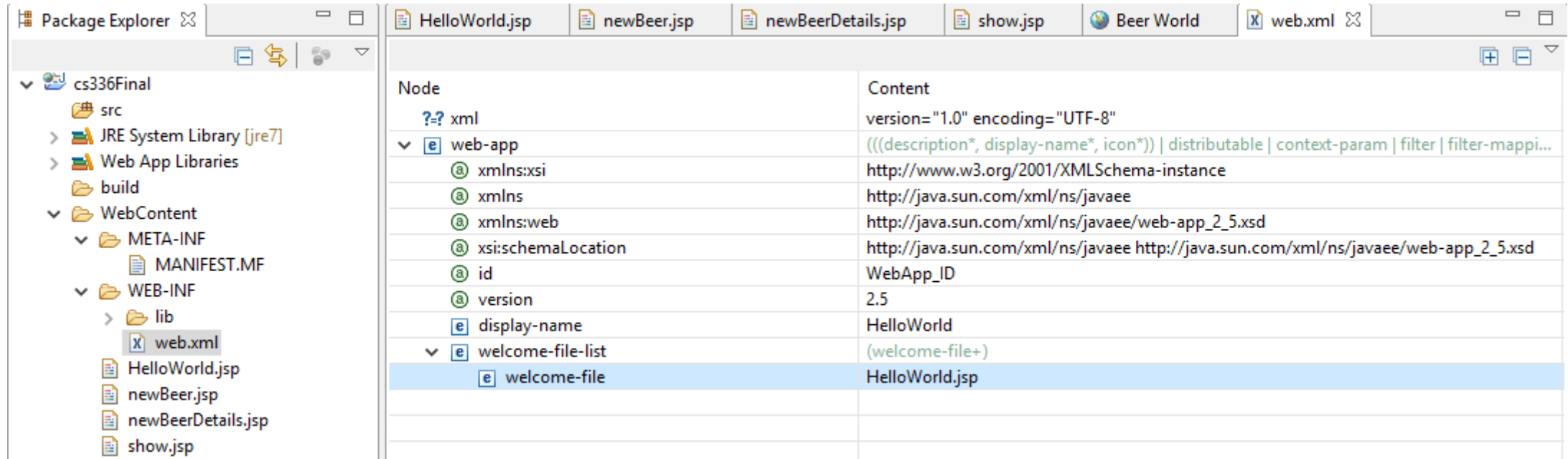
# The Template Project

- Now you can see your project home page, HelloWorld.jsp page.



# The Template Project

6. The home page is set in web.xml, you can set your own page if you want.



The screenshot shows an IDE interface with the Package Explorer on the left and the web.xml file content in the main editor. The Package Explorer shows a project named 'cs336Final' with a 'WebContent' folder containing 'META-INF' and 'WEB-INF'. The 'WEB-INF' folder contains a 'lib' folder and a 'web.xml' file. The 'web.xml' file is selected, and its content is displayed in the main editor.

Node	Content
xml	version="1.0" encoding="UTF-8"
web-app	(((description*, display-name*, icon*))   distributable   context-param   filter   filter-mappi...
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
xmlns	http://java.sun.com/xml/ns/javaee
xmlns:web	http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd
xsi:schemaLocation	http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd
id	WebApp_ID
version	2.5
display-name	HelloWorld
welcome-file-list	(welcome-file+)
welcome-file	HelloWorld.jsp

# The Template Project

## 7. Connect to your own db instance in Project

- In order to interact with db instance (add, delete, update, select), you need to set your own database address in the project.
- At the same time, the database username and password are both essential.
- Replace the database information with your own database information as follows.

# The Template Project

Project Explorer:

- JAX-WS Web Services
- Java Resources
  - src
    - com.cs336.pkg
      - ApplicationDB.java
      - ApplicationDB
- Libraries
- JavaScript Resources
- build
- WebContent
  - META-INF
  - WEB-INF
    - index.jsp
    - newBarBeerPrice.jsp
    - query.jsp
    - sellsNewBeer.jsp
    - show.jsp
    - show2.jsp
    - showGraph.jsp

ApplicationDB.java:

```
10
11 }
12
13 public Connection getConnection(){
14
15     //Create a connection string
16     String connectionUrl = "jdbc:mysql://localhost:3306/BarBeerDrinkerSample";
17     Connection connection = null;
18
19     try {
20         //Load JDBC driver - the interface standardizing the connection procedure. Look at WEB-INF\lib for
21         Class.forName("com.mysql.jdbc.Driver").newInstance();
22     } catch (InstantiationException e) {
23         // TODO Auto-generated catch block
24         e.printStackTrace();
25     } catch (IllegalAccessException e) {
26         // TODO Auto-generated catch block
27         e.printStackTrace();
28     } catch (ClassNotFoundException e) {
29         // TODO Auto-generated catch block
30         e.printStackTrace();
31     }
32     try {
33         //Create a connection to your DB
34         connection = DriverManager.getConnection(connectionUrl,"root", "root");
35     } catch (SQLException e) {
36         // TODO Auto-generated catch block
37         e.printStackTrace();
38     }
39 }
```

Annotations:

- Schema name (points to `BarBeerDrinkerSample`)
- MySQL Username and password (points to `"root", "root"`)

# The Template Project

## 8. Let's have a beer

Select the radio button and then click submit below it

The screenshot shows an IDE with a Package Explorer on the left and a web browser window on the right. The Package Explorer shows a project named 'cs336Final' with a 'WebContent' folder containing 'META-INF', 'WEB-INF', and 'lib' subfolders. The 'WEB-INF' folder contains 'lib' and 'web.xml'. The 'lib' folder contains 'HelloWorld.jsp', 'newBeer.jsp', 'newBeerDetails.jsp', and 'show.jsp'. The web browser window shows the URL 'http://localhost:8080/cs336Final/show.jsp?command=beers' and displays a table of beer information.

name	manf
Blue Moon	Coors Brewing Company
Budweiser	Anheuser-Busch
Creamy Dark	Jacob Leinenkugel Brewing Company
Extra Gold	Coors Brewing Company
Hefeweizen	Jacob Leinenkugel Brewing Company
Hefeweizen Doppelbock	Jacob Leinenkugel Brewing Company
Heiniken	Heiniken International
ICEHOUSE	Plank Road Brewery
Killian's	Coors Brewing Company
Michelob Amber Bock	Anheuser-Busch
Michelob Golden Draft	Anheuser-Busch
Michelob Golden Draft Light	Anheuser-Busch
Michelob ULTRA	Anheuser-Busch
Original Premium	Jacob Leinenkugel Brewing Company
Original Premium Lager	Jacob Leinenkugel Brewing Company
Original Premium Lager Dog	Plank Road Brewery
Sauza Diablo	Miller Brewing Company

# The Template Project

## 9. Let's go to a bar

Select the radio button and then click submit below it

The screenshot shows an IDE with a Package Explorer on the left and a web browser on the right. The Package Explorer shows a project named 'cs336Final' with a 'WebContent' folder containing 'META-INF' (with 'MANIFEST.MF') and 'WEB-INF' (with 'lib' and 'web.xml'). The web browser shows the URL 'http://localhost:8080/cs336Final/show.jsp?command=bars' and displays a table of bars.

name	addr
A.P. Stump's	null
Blue Angel	null
Blue Tattoo	1500 Market St.
Britannia Arms	null
Cabana	1200 California Ave.
Caravan	null
Club 175	null
Coconut Willie's Cocktail Lounge	null
Eulipia	1060 West Addison
Gecko Grill	1060 West Addison
Giza Hookah Lounge	null
Hedley Club	161st St and River Ave
Il Fornaio	161st St and River Ave
Seven Bamboo	161st St and River Ave
The B-Hive	null
The Backbeat	null
The Blank Club	null

# The Template Project

## 10. Insert a tuple into sells table

Input bar name, beer name and cost, then click submit.

You can find a new record inserted into your database after submitting this form.

**-NOTE:** since you insert a tuple in sells table which has FKs in the bar and beer table, make sure the beer and bar you insert already exist in these two tables.

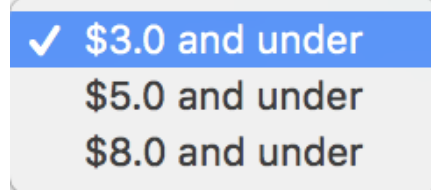


# The Template Project

## 11. Query the beers with cost

Choose one option from the dropdown menu, then click submit.

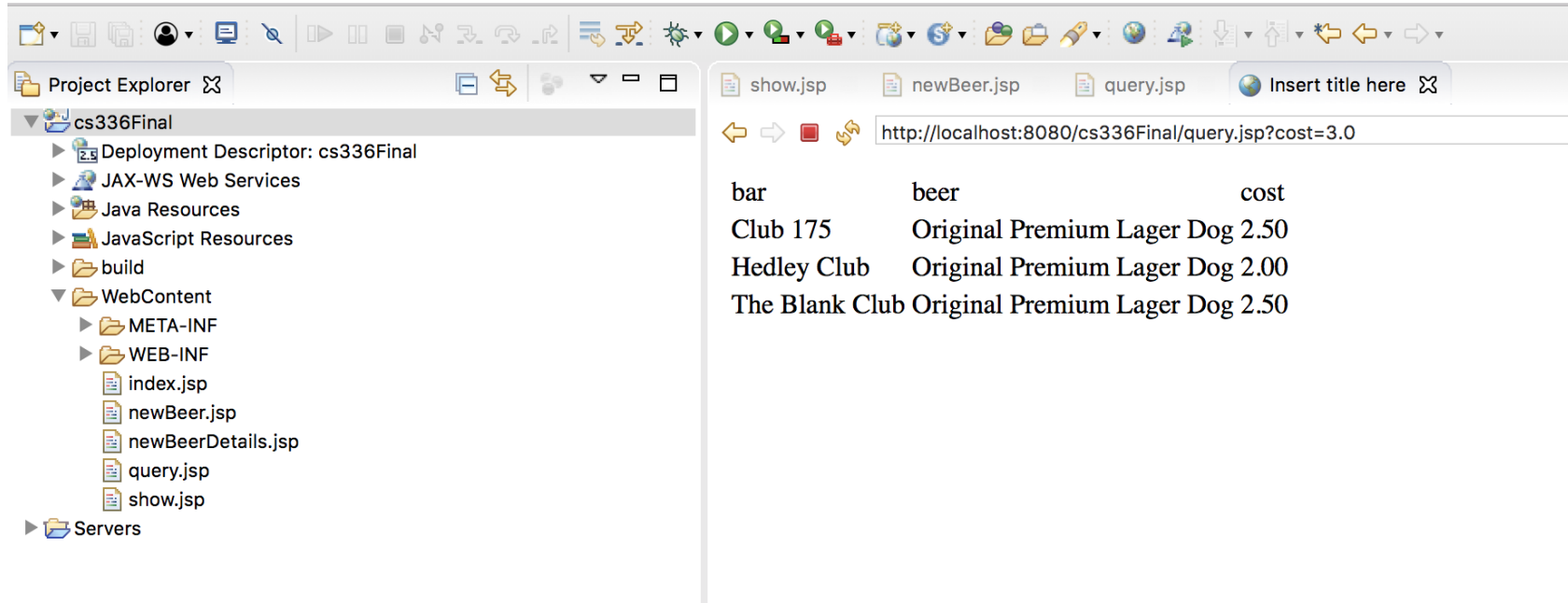
Or we can query the beers with cost:



✓ \$3.0 and under
\$5.0 and under
\$8.0 and under

# The Template Project

- Query the beers with cost  $\leq 3$



# The Template Project

- Query the beers with  $\text{cost} \leq 5$

The screenshot shows an IDE with a project named 'cs336Final'. The Project Explorer on the left lists the following structure:

- cs336Final
  - Deployment Descriptor: cs336Final
  - JAX-WS Web Services
  - Java Resources
  - JavaScript Resources
  - build
  - WebContent
    - META-INF
    - WEB-INF
    - index.jsp
    - newBeer.jsp
    - newBeerDetails.jsp
    - query.jsp
    - show.jsp
  - Servers

The main editor area shows the 'query.jsp' file. The browser window displays the URL `http://localhost:8080/cs336Final/query.jsp?cost=5.0` and the following output:

bar	beer	cost
Club 175	Budweiser	4.50
Club 175	Original Premium Lager Dog	2.50
Hedley Club	Molsons	4.00
Hedley Club	Original Premium Lager Dog	2.00
The Blank Club	Molsons	4.50
The Blank Club	Original Premium Lager Dog	2.50