CS 5513 – Dr. Le Gruenwald

Instructions for setting up an Eclipse Project to develop a dynamic JSP Web-Application which uses Azure SQL database

Table of Contents

Step 1 – Install Java SDK

- Download JDK 8u261 from the below link (we couldn't make our setup work with later versions of Java, please don't try to use them)
  - https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- Follow the installation instructions at the below link
  - https://docs.oracle.com/javase/8/docs/technotes/guides/install/install_overview.html

Step 2 – Install Eclipse IDE

Download and install latest "Eclipse IDE for **Enterprise** Java Developers" available at the link below (look for "Install Guide", if needed):

- https://www.eclipse.org/downloads/packages/

Step 3 – Download Azure JDBC driver

If you haven't done so already, download Microsoft JDBC Driver 8.4 for SQL Server from the below link:

- https://docs.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=azuresqldb-current

Follow the installation instructions on above page, note the installation directory.

Step 4 – Download Apache Tomcat 9 Web-Server

Download a ZIP archive with the latest version of Apache Tomcat 9 Web-Server (9.0.38 at the time of this writing) available at the link below:

- https://tomcat.apache.org/download-90.cgi

Unzip the archive, record the resulting directory location

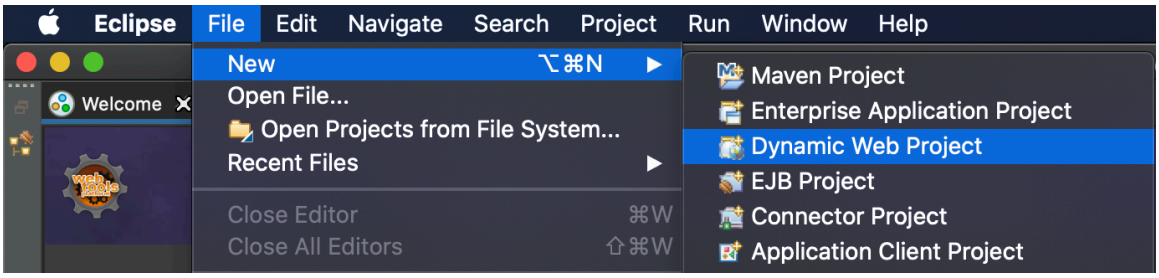
Step 5 – Create the Database Tables

- Using the SQL IDE of your choice (Azure Data Studio for example), create the tables to be used by your web-application.
- Appendix A provides example SQL statements we will use to create the database for our example application.

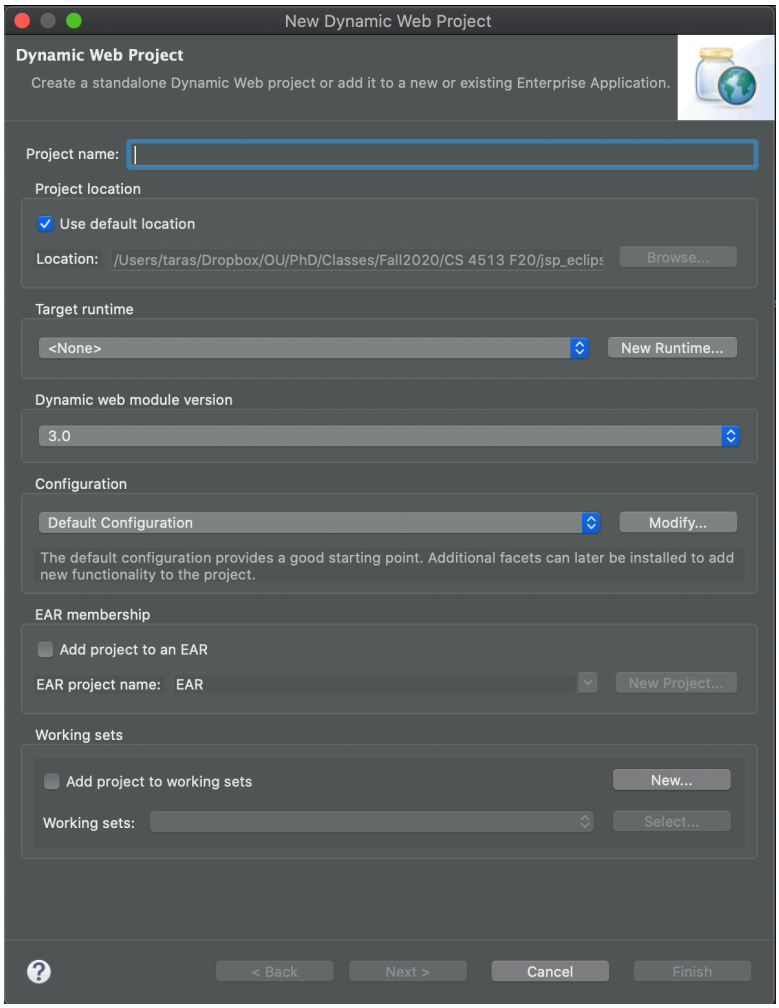
Step 6 – Create a JSP based web-application project in Eclipse

1. Launch Eclipse for Enterprise Java Developers (create and select a new workspace directory when prompted)

2. Navigate the menu and click “File” -> “New” -> “Dynamic Web Project” (see screenshot below)



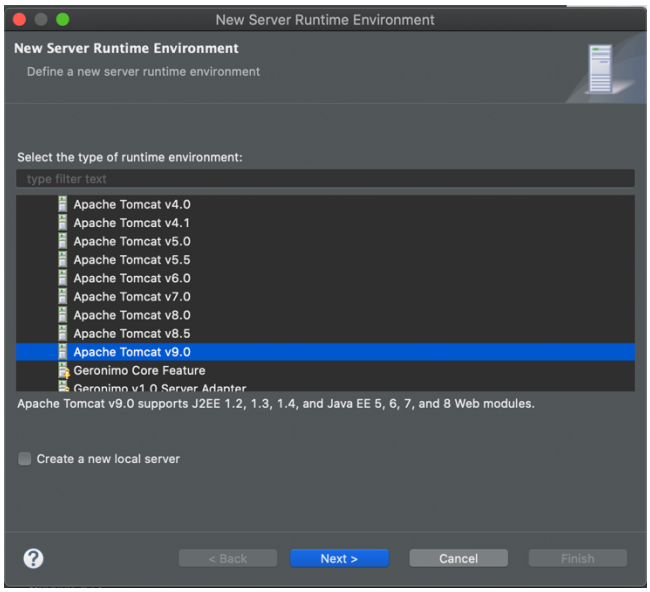
3. You should now see the below project creation window



4. Give project a name, “jsp_azure_test” for example

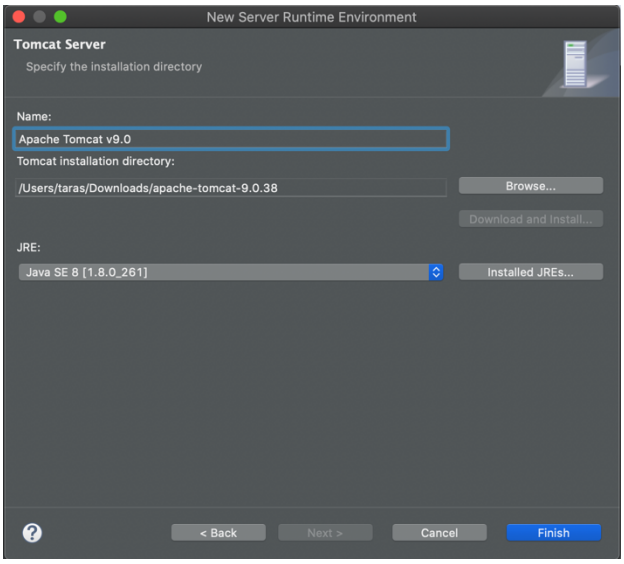
5. Click on “New Runtime” button

6. In a pop-up window, select “Apache Tomcat v9.0” in the “Apache” folder, click on “Next” button (see the screenshot below)

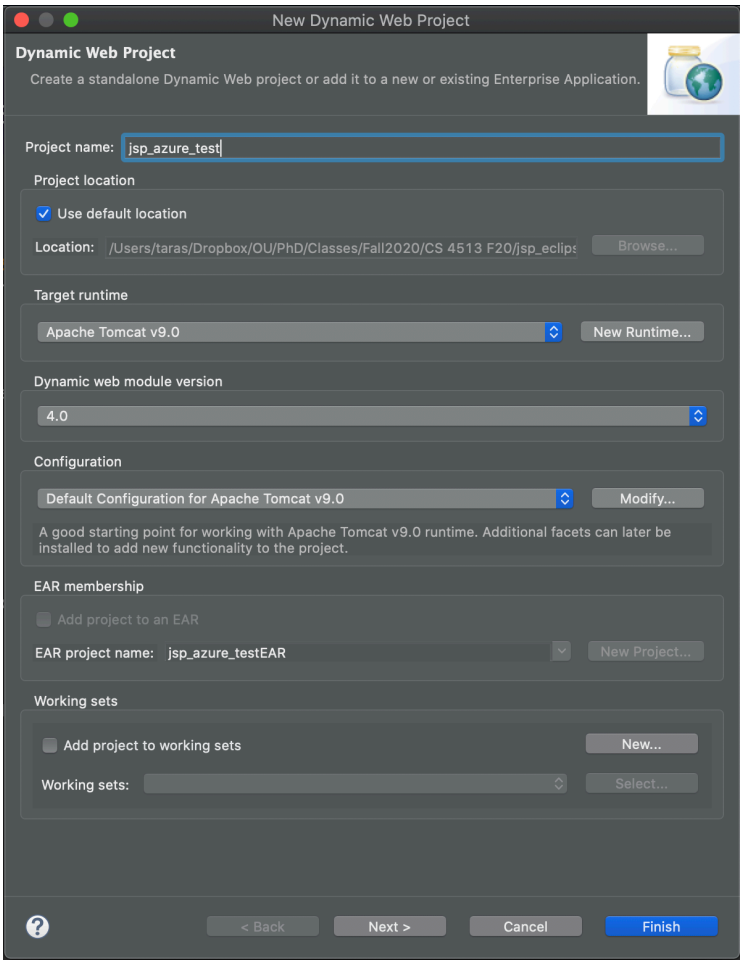


7. Click on the “Browse…” button, navigate to and select the directory where you unzipped your Apache Tomcat 9 archive, click “Open” button.

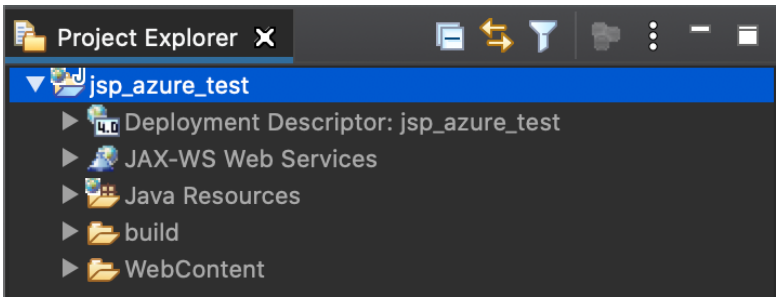
8. Under “JRE”, select Java SE 8 (if that’s not an option, you will have to click on “Installed JREs…” button, and then “Add” -> “Standard (or MacOS) VM” -> “Next” -> “Directory…” -> Select Java SE 8 home directory -> “Finish” -> Select newly added Java SE 8 entry -> “Apply and Close”). Your “New Server Runtime Environment” should now look like the screenshot below. Click the “Finish” button.



9. Now, back at the "New Dynamic Web Project" window (see screenshot below), click on "Finish" button.

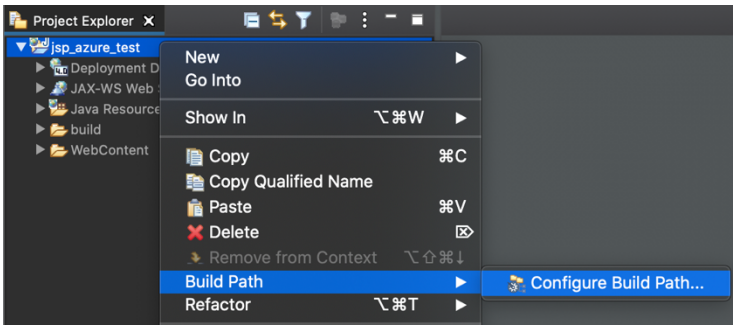


10. Your project explorer window should now look like the screenshot below (you might need to close the “Welcome” window first to see it)

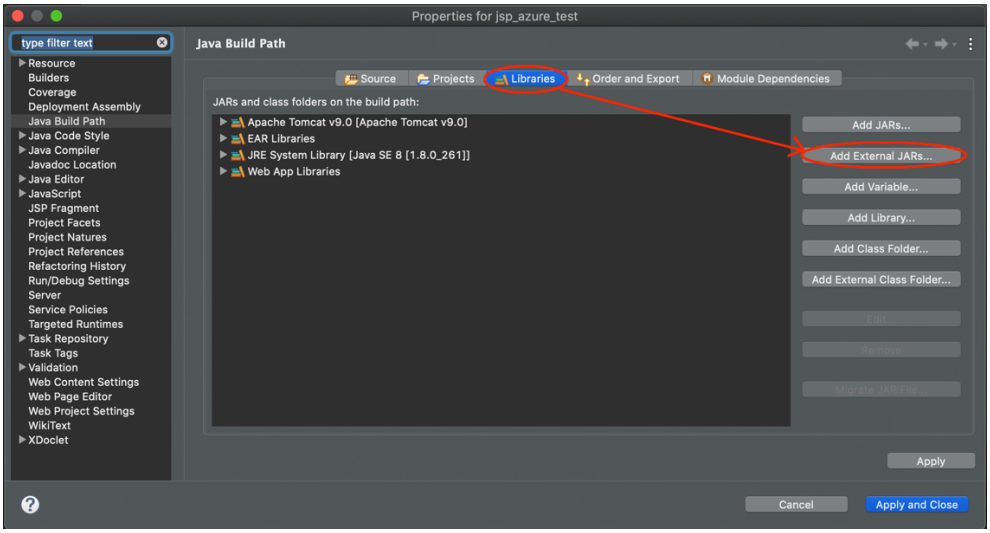


Step 7 – Add Azure JDBC driver to the project

1. Right-click on the project name in the Project Explorer tab, select “Build Path” -> “Configure Build Path…” (see screenshot below)

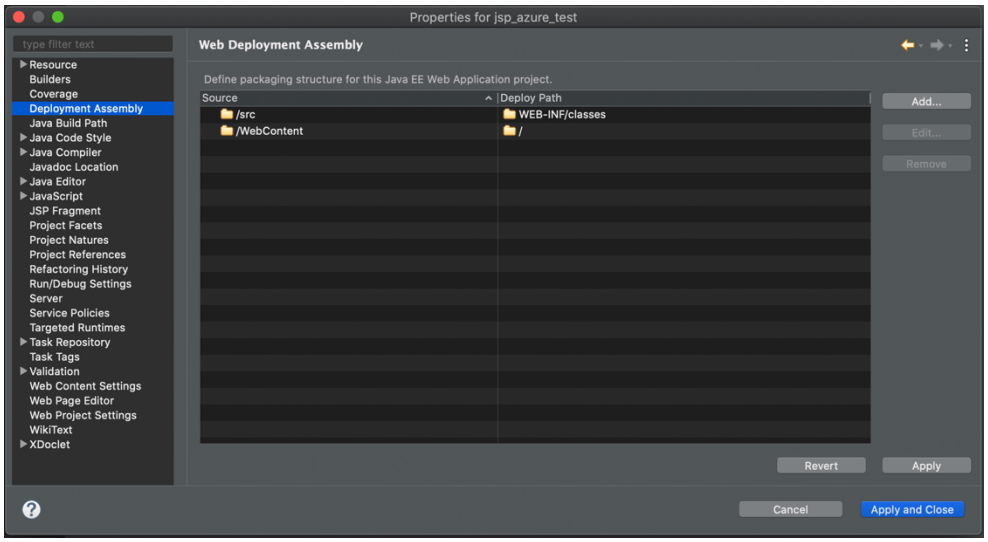


2. Select “Libraries” tab, click on “Add External JARs…” button

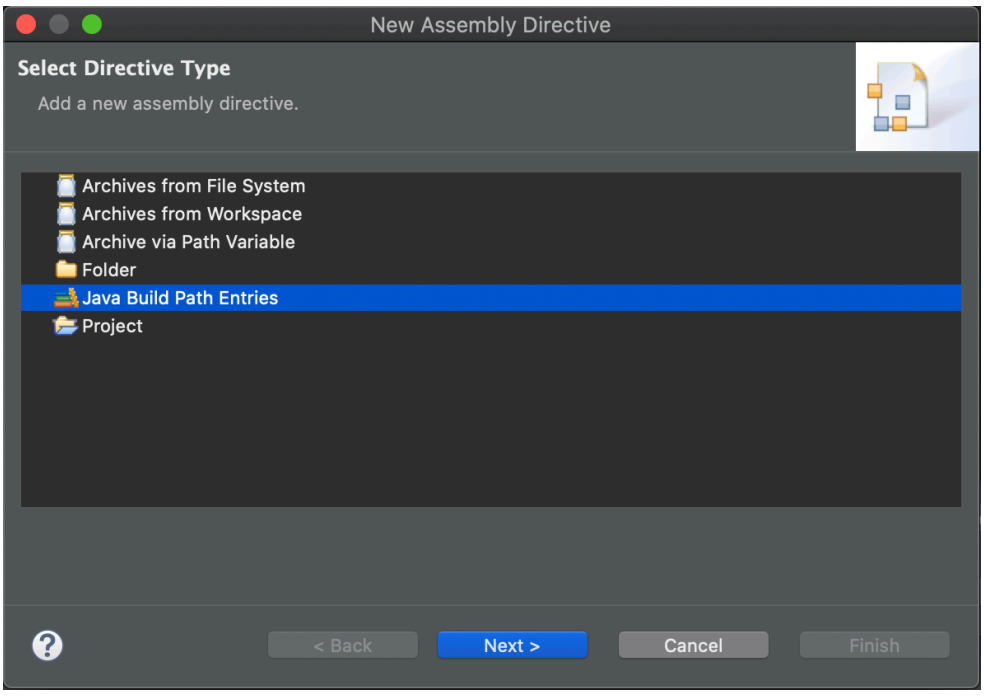


3. Navigate to and select the “mssql-jdbc-8.4.1.jre8.jar” file from the installation directory of your JDBC driver from Step 3, click on “Open” button. Now click on “Apply” button.

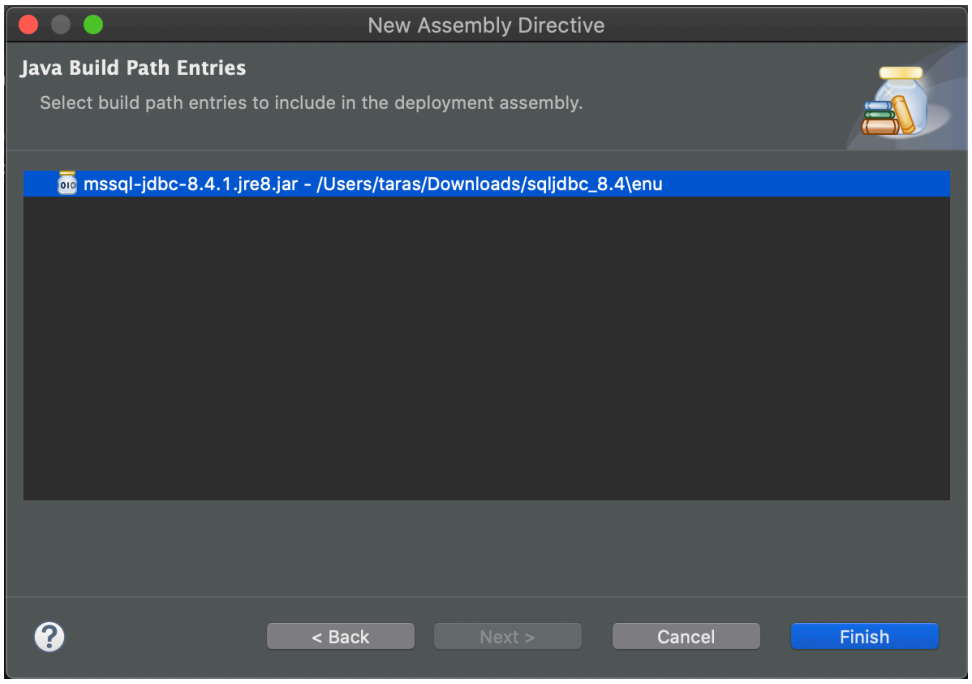
4. Don’t close project properties window yet, select “Deployment Assembly” tab on the left, click on the “Add…” button (see screenshot below).



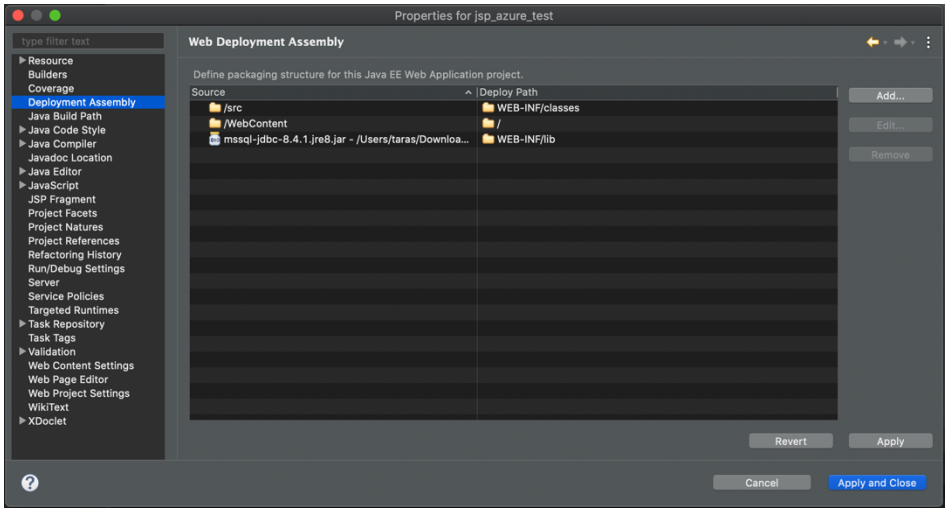
5. In the “New Assembly Directive” window select “Java Build Path Entries”, click “Next >” button (see screenshot below).



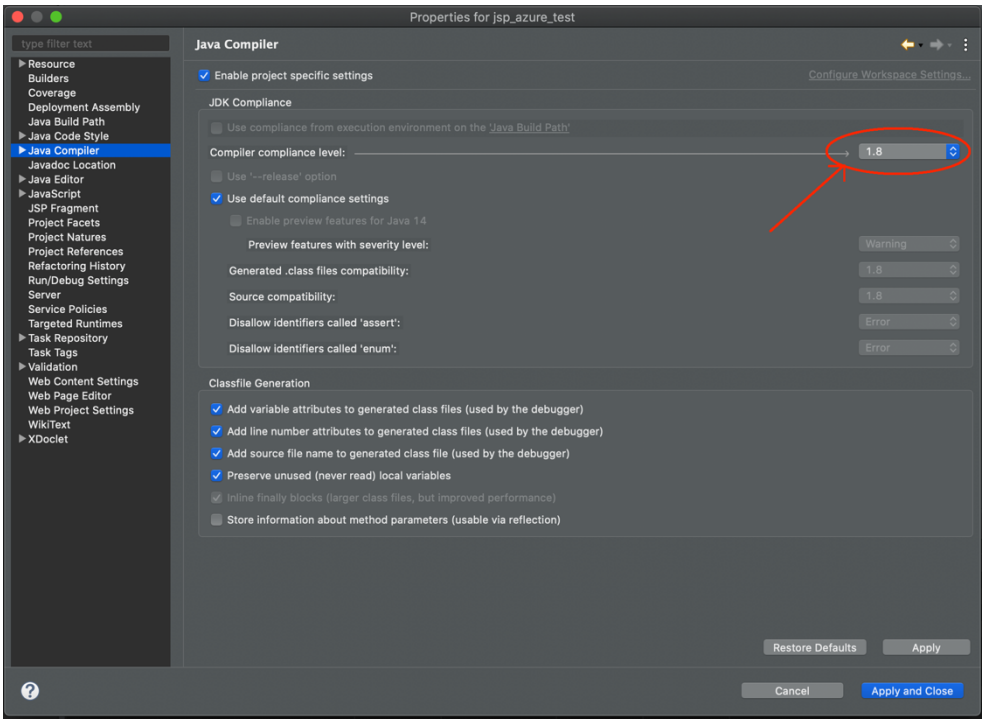
6. Select “mssql-jdbc-8.4.1.jre8.jar” and click on “Finish” button.



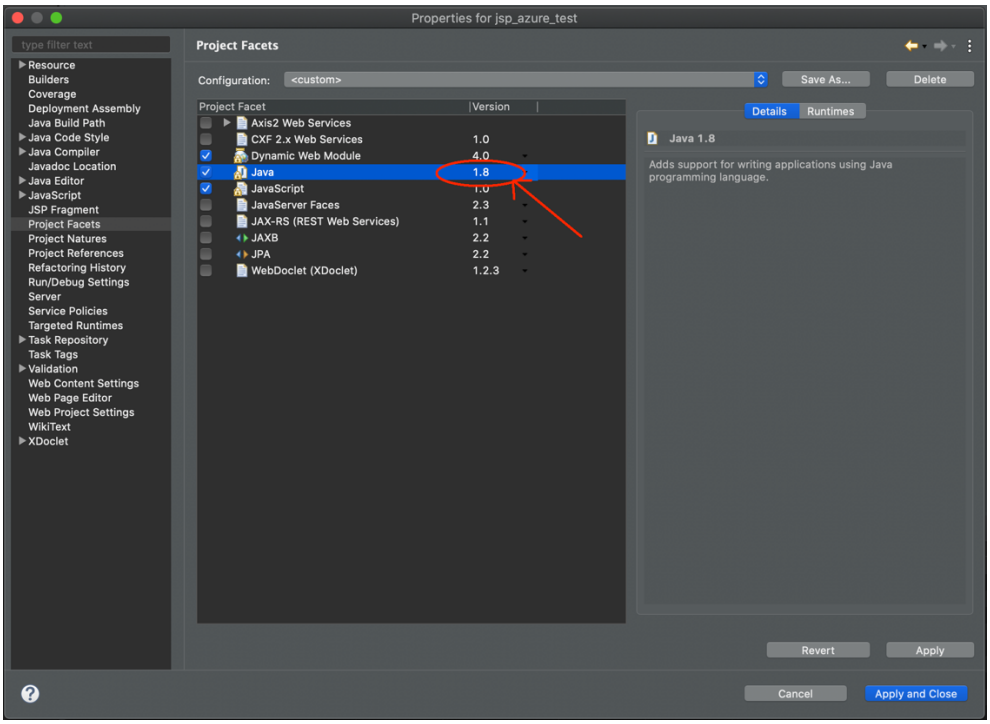
7. Your project properties window should now look like the window below. Click "Apply", but don't close the window yet.



8. Go to the "Java Compiler" tab and make sure that "Complier compliance level" is 1.8 (see the screenshot below).

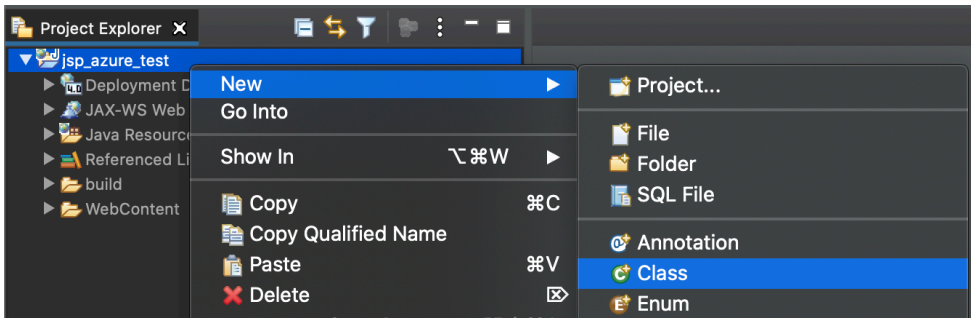


9. Now go to the “Project Facets” tab and make sure that “Java” is also 1.8 (see the screenshot below). Now click on the “Apply and Close” button.

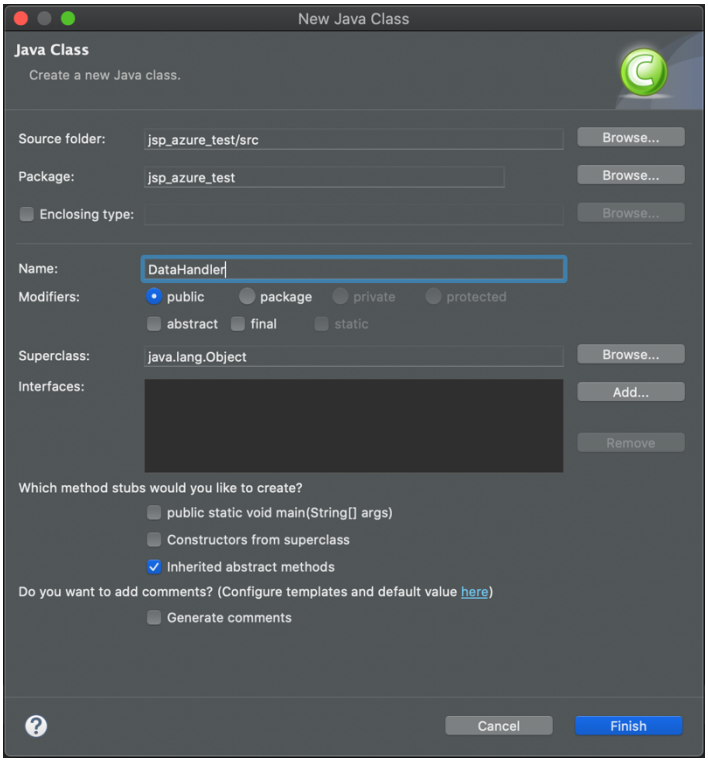


Step 8 – Add DataHandler.java file to the project

1. Right click on the project name, click on “New” -> “Class” (see screenshot below).



2. Name your new Java class DataHandler and click “Finish” button (see screenshot below).



3. Copy-paste the contents of the DataHandler.java file from the Appendix A or a corresponding code-snippet file from Canvas (see screenshot below).

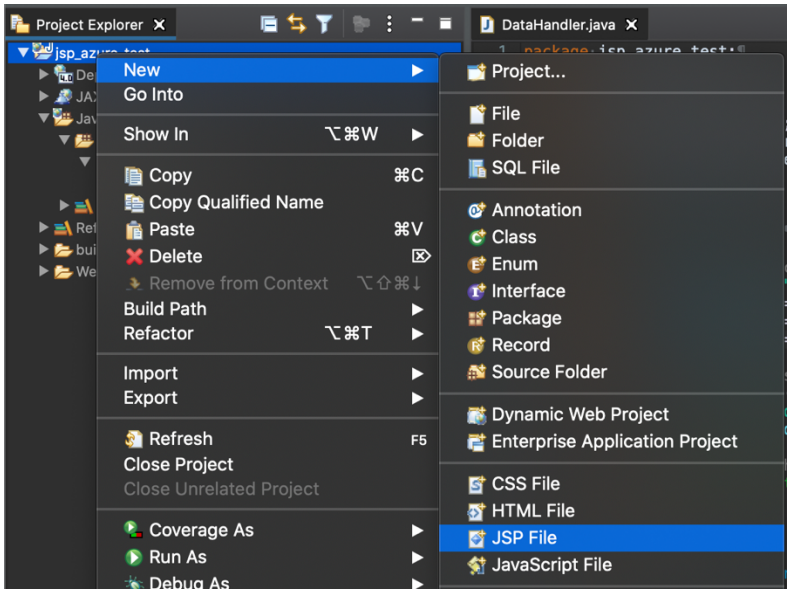
DataHandler.java

```
package jsp_azure_test;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class DataHandler {

    private Connection conn;

    //Azure SQL connection credentials
    private String server = "your4x4-sql-server.database.windows.net";
    private String database = "cs-dsa-4513-sql-db";
    private String username = "your4x4";
    private String password = "                    ";

    //Resulting connection string
    final private String url =
            String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;
                    server, database, username, password);

    //Initialize and save the database connection
    private void getDBConnection() throws SQLException {
        if (conn != null) {
            return;
        }

        this.conn = DriverManager.getConnection(url);
    }

    //Return the result of selecting everything from the movie_night table
    public ResultSet getAllMovies() throws SQLException {
        getDBConnection();

        final String sqlQuery = "SELECT * FROM movie_night;";
        final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
        return stmt.executeQuery();
    }

}
```

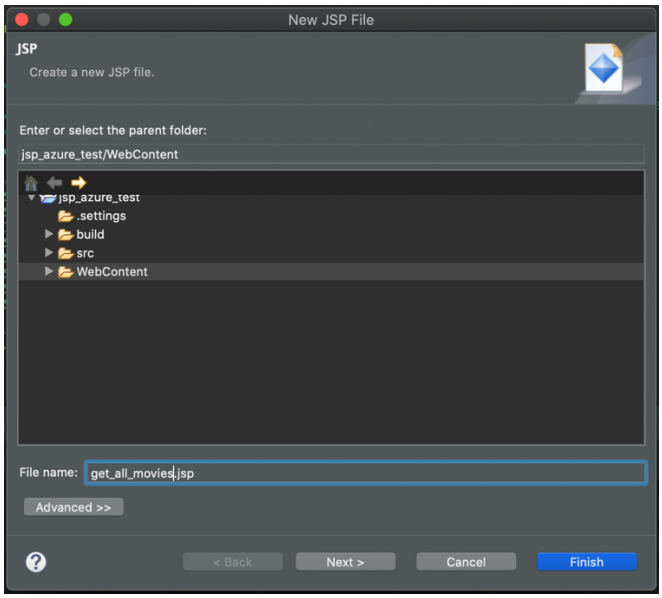
4. Fill in your own values for “server”, “database”, “username” and “password” and save the file.

Step 9 – Add JSP files to the project

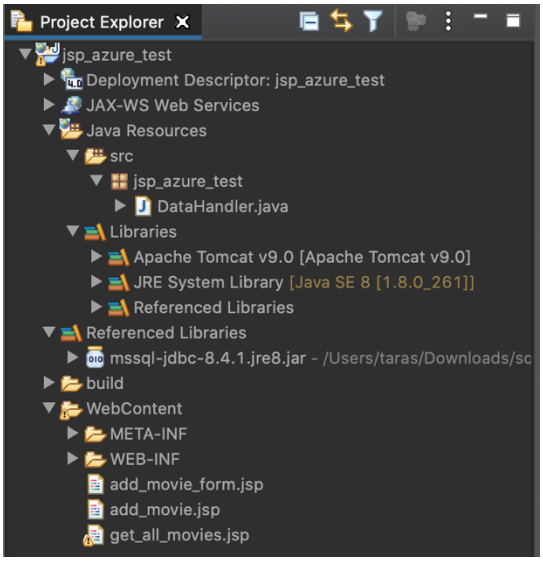
1. Right click on the project name, click on “New” -> “JSP File” (see screenshot below)



2. Name your JSP file “get_all_movies.jsp”, click on “Finish” button.

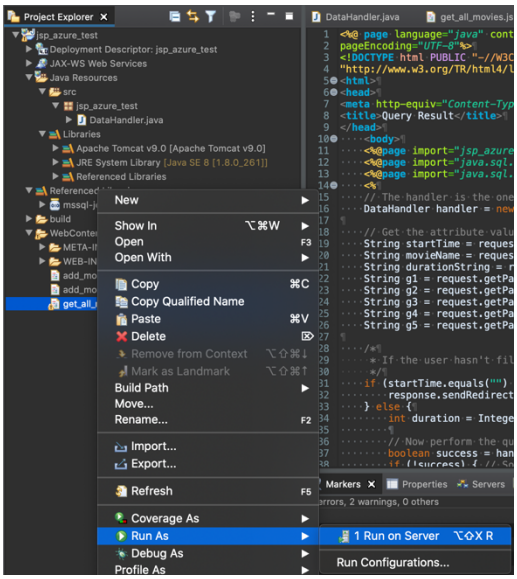


3. Copy-paste contents of the get_all_movies.jsp file from the Appendix A and save the file.
4. Repeat above three instructions for 2 more files found in Appendix A – add_movie_form.jsp and add_movie.jsp
5. Your project explorer tab should now look like the below screenshot.

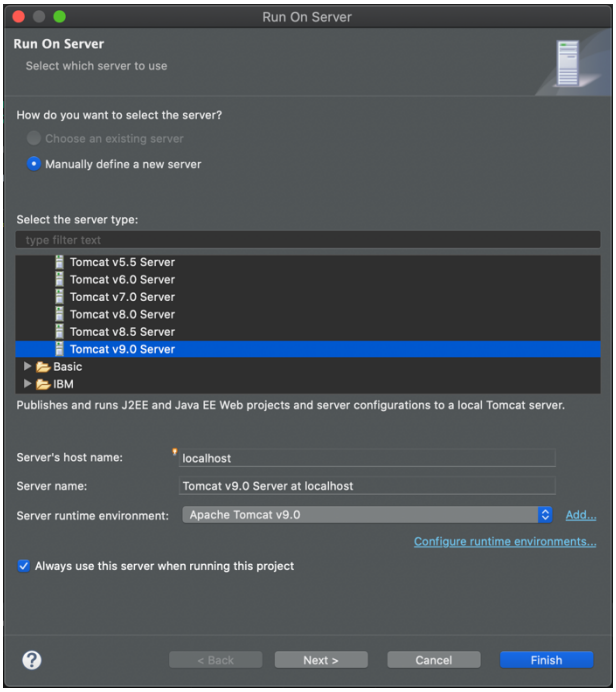


Step 10 – Test the project (please read all 9 steps before executing any)

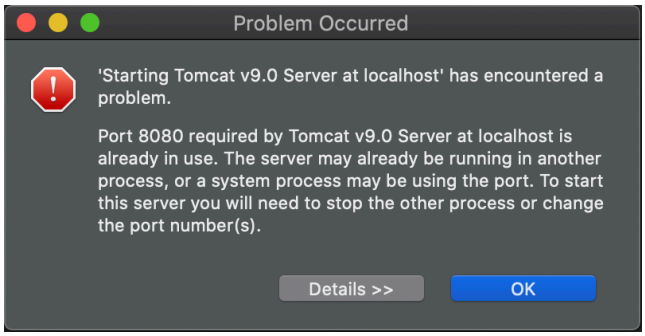
1. Right click on get_all_movies.jsp in the Project Explorer tab, select “Run As” -> “Run on Server” (see screenshot below)



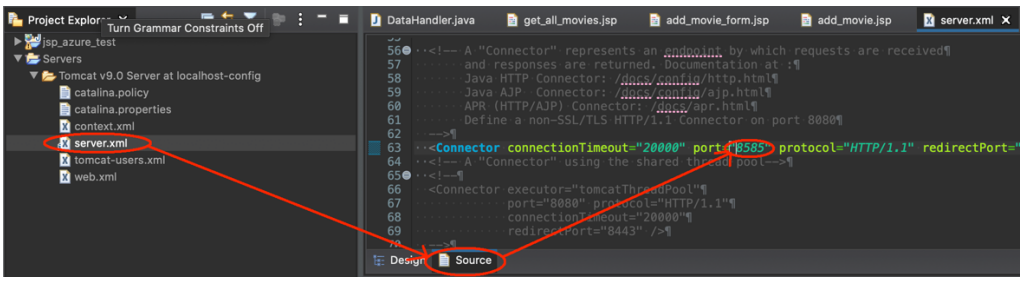
2. First time you do the above, you will be asked to select a web-server (see below screenshot). Select “Always use this server…”, click on “Finish” button.



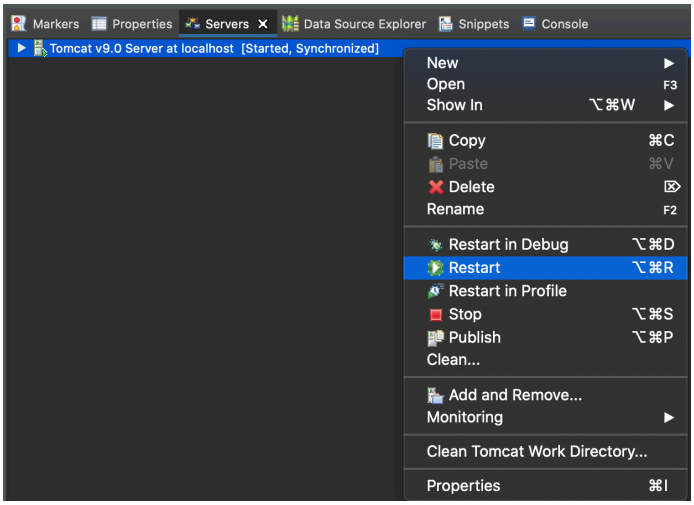
3. If you already have another web-server running locally on port 8080 (and quite likely don't even know about it), you will see the below error. Click "OK" button. If you don't see the problem window, skip the next step about fixing it.



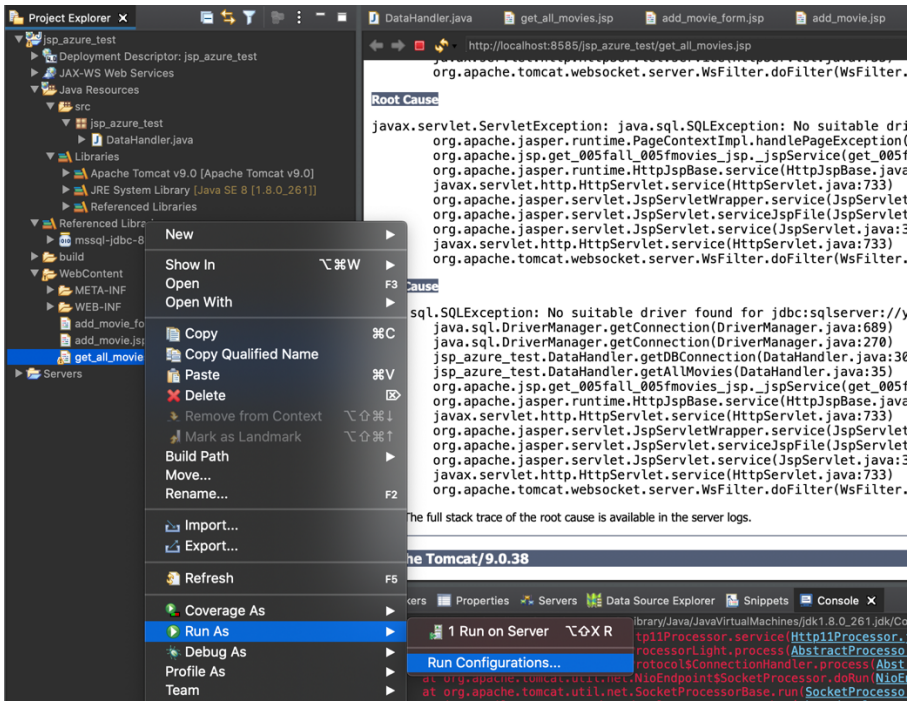
4. To fix the above error, notice "Servers" folder appearing in the "Project Explorer" tab. Expand it and open (via double-click) the server.xml file. Locate the "8080" value (you might need to switch from "Design" to "Source" tab of the file editor), change the value of port attribute to a free port in your system, 8585, for example, is a good guess (see the screenshot below).



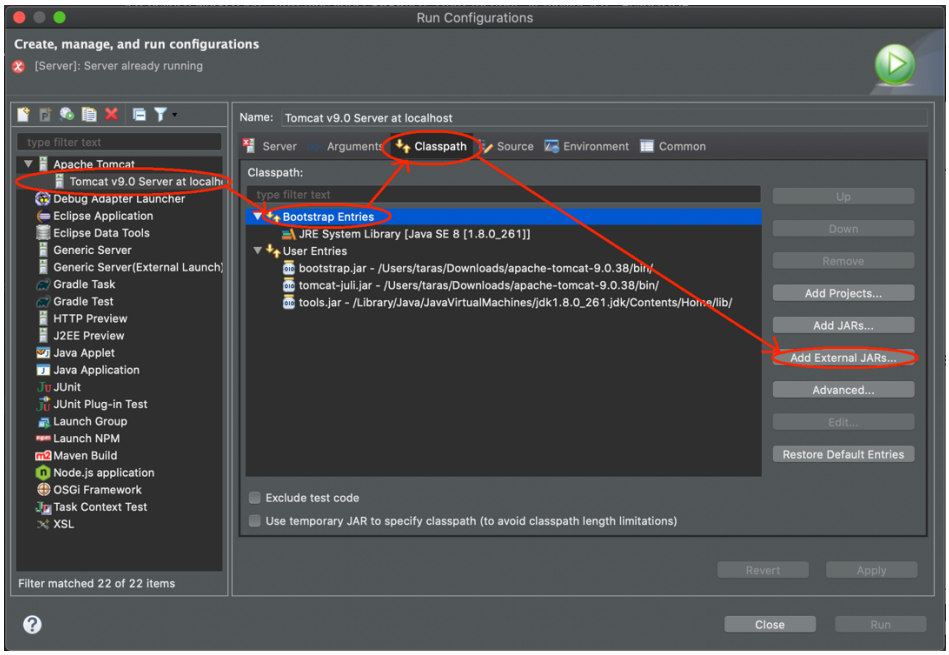
5. Now, for these changes to have effect, restart the server. Then try running the get_all_movies.jsp again.



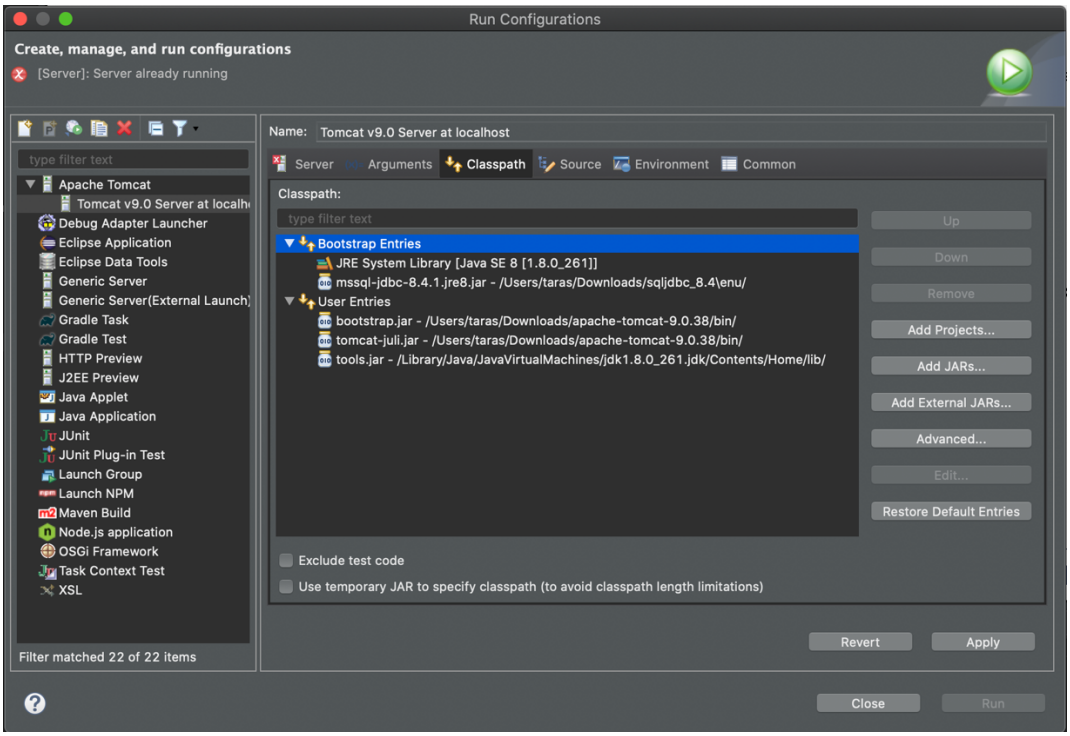
6. Even if you didn't face the port conflict error described above, it's likely that your run resulted in an error related to the wrong installation if the JDBC driver. If that didn't happen, skip the remainder of this and the next two steps. To fix this error, right-click on get_all_movies.jsp -> Run As -> Run Configurations…



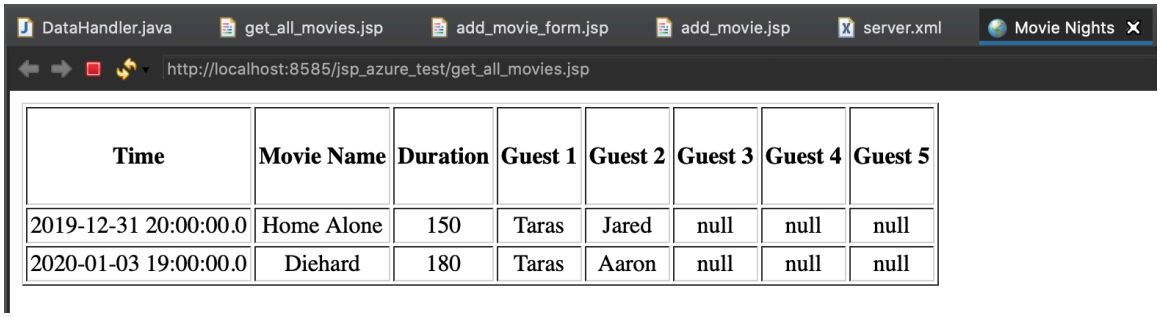
7. Click on, "Tomcat v9.0 Server…", then "Classpath", then "Bootstrap Entries", then "Add External JARs" (see the screenshot bellow)



8. In the pop-up window, navigate to and select the “mssql-jdbc-8.4.1.jre8.jar” file. Click on “Apply” and then on “Close” button (see below screenshot). Now, restart the server again.

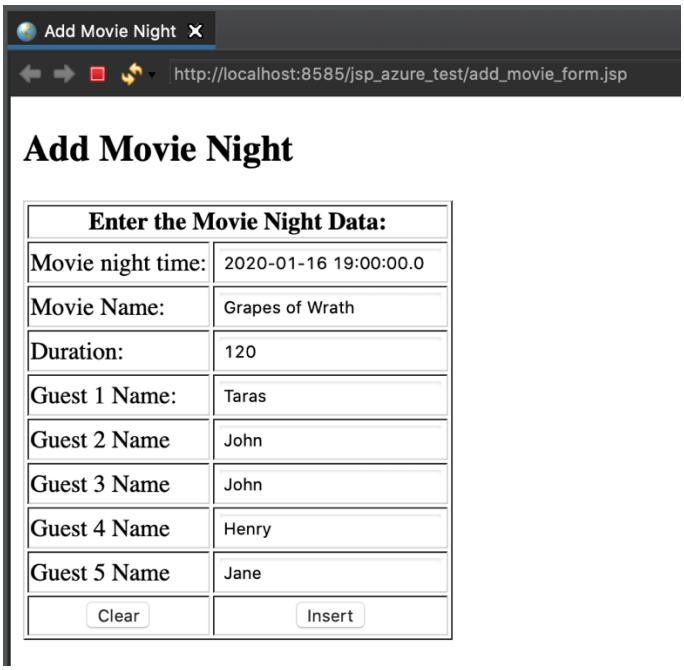


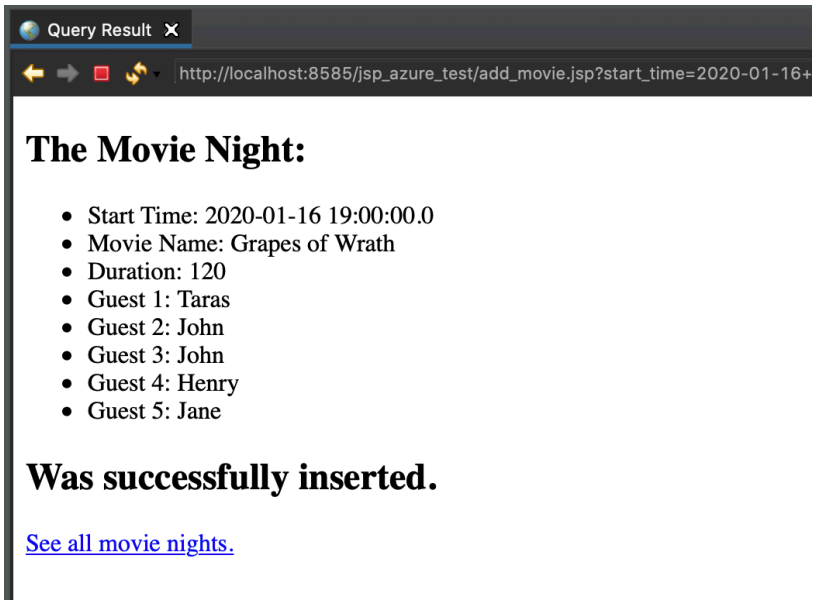
9. Repeat the get_all_movies.jsp -> “Run As” -> “Run on Server” instruction. Now you should be able to see the browser tab open in Eclipse (see screenshot below).



Time	Movie Name	Duration	Guest 1	Guest 2	Guest 3	Guest 4	Guest 5
2019-12-31 20:00:00.0	Home Alone	150	Taras	Jared	null	null	null
2020-01-03 19:00:00.0	Diehard	180	Taras	Aaron	null	null	null

10. Testing the ”add_movie_form.jsp” file is done similarly. After you populate the input form with data and click submit, you should be redirected to “add_movie.jsp” page in the eclipse web-browser (see two below screenshots).


Add Movie Night
http://localhost:8585/jsp_azure_test/add_movie_form.jsp
Add Movie Night
Enter the Movie Night Data:
Movie night time:
2020-01-16 19:00:00.0
Movie Name:
Grapes of Wrath
Duration:
120
Guest 1 Name:
Taras
Guest 2 Name
John
Guest 3 Name
John
Guest 4 Name
Henry
Guest 5 Name
Jane
Clear
Insert


Query Result
http://localhost:8585/jsp_azure_test/add_movie.jsp?start_time=2020-01-16+
The Movie Night:
Start Time: 2020-01-16 19:00:00.0
Movie Name: Grapes of Wrath
Duration: 120
Guest 1: Taras
Guest 2: John
Guest 3: John
Guest 4: Henry
Guest 5: Jane
Was successfully inserted.
See all movie nights.

Appendix A

create_table.sql

Executing the queries in the below .sql file (with Azure Data Studio, for example) creates a very simple database for storing information about upcoming movie nights at someone's house.

```
DROP TABLE IF EXISTS movie_night; --Drop the table if it was previously created

--Create the new table for movie_nights schedule
CREATE TABLE movie_night (
      start_time DATETIME PRIMARY KEY,
      movie_name VARCHAR(64),
      duration_min INT,
      guest_1 VARCHAR(64),
      guest_2 VARCHAR(64),
      guest_3 VARCHAR(64),
      guest_4 VARCHAR(64),
      guest_5 VARCHAR(64),
);

--Insert two records to begin with
INSERT INTO movie_night
 (start_time, movie_name, duration_min, guest_1, guest_2)
VALUES
 ('2019-12-31 20:00:00', 'Home Alone', 150, 'Taras', 'Jared'),
 ('2020-01-03 19:00:00', 'Diehard', 180, 'Taras', 'Aaron');
```

DataHandler.java

Below java file contains code used to connect to the Azure SQL Database and execute the example queries.

```
package jsp_azure_test;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class DataHandler {

    private Connection conn;

    // Azure SQL connection credentials
    private String server = "<Replace Me!>-sql-server.database.windows.net";
    private String database = "cs-dsa-4513-sql-db";
    private String username = "<Replace Me!>";
    private String password = "<Replace Me!>";

    // Resulting connection string
```

```
    final private String url =

String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;trus
tServerCertificate=false;hostNameInCertificate=*.database.windows.net;loginTimeout=30;",
                    server, database, username, password);

    // Initialize and save the database connection
    private void getDBConnection() throws SQLException {
        if (conn != null) {
            return;
        }

        this.conn = DriverManager.getConnection(url);
    }

    // Return the result of selecting everything from the movie_night table
    public ResultSet getAllMovies() throws SQLException {
        getDBConnection();

        final String sqlQuery = "SELECT * FROM movie_night;";
        final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
        return stmt.executeQuery();
    }

    // Inserts a record into the movie_night table with the given attribute values
    public boolean addMovie(
            String startTime, String movieName, int duration, String g1, String g2,
String g3, String g4, String g5) throws SQLException {

        getDBConnection(); // Prepare the database connection

        // Prepare the SQL statement
        final String sqlQuery =
                "INSERT INTO movie_night " +
                    "(start_time, movie_name, duration_min, guest_1, guest_2, guest_3,
guest_4, guest_5) " +
                "VALUES " +
                "(?, ?, ?, ?, ?, ?, ?, ?)";
        final PreparedStatement stmt = conn.prepareStatement(sqlQuery);

        // Replace the '?' in the above statement with the given attribute values
        stmt.setString(1, startTime);
        stmt.setString(2, movieName);
        stmt.setInt(3, duration);
        stmt.setString(4, g1);
        stmt.setString(5, g2);
        stmt.setString(6, g3);
        stmt.setString(7, g4);
        stmt.setString(8, g5);

        // Execute the query, if only one record is updated, then we indicate success by
returning true
        return stmt.executeUpdate() == 1;
    }
}
```

get_all_movies.jsp

Executing below JSP file generates an HTML file with a table containing all the records from the movie_night SQL database table.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
    <meta charset="UTF-8">
```

```
        <title>Movie Nights</title>
    </head>
    <body>
        <%@page import="jsp_azure_test.DataHandler"%>
        <%@page import="java.sql.ResultSet"%>
        <%
            // We instantiate the data handler here, and get all the movies from the
database
            final DataHandler handler = new DataHandler();
            final ResultSet movies = handler.getAllMovies();
        %>
        <!-- The table for displaying all the movie records -->
        <table cellspacing="2" cellpadding="2" border="1">
            <tr> <!-- The table headers row -->
              <td align="center">
                <h4>Time</h4>
              </td>
              <td align="center">
                <h4>Movie Name</h4>
              </td>
              <td align="center">
                <h4>Duration</h4>
              </td>
              <td align="center">
                <h4>Guest 1</h4>
              </td>
              <td align="center">
                <h4>Guest 2</h4>
              </td>
              <td align="center">
                <h4>Guest 3</h4>
              </td>
              <td align="center">
                <h4>Guest 4</h4>
              </td>
              <td align="center">
                <h4>Guest 5</h4>
              </td>
            </tr>
            <%
               while(movies.next()) { // For each movie_night record returned...
                   // Extract the attribute values for every row returned
                   final String time = movies.getString("start_time");
                   final String name = movies.getString("movie_name");
                   final String duration = movies.getString("duration_min");
                   final String guest1 = movies.getString("guest_1");
                   final String guest2 = movies.getString("guest_2");
                   final String guest3 = movies.getString("guest_3");
                   final String guest4 = movies.getString("guest_4");
                   final String guest5 = movies.getString("guest_5");

                   out.println("<tr>"); // Start printing out the new table row
                   out.println( // Print each attribute value
                        "<td align=\"center\">" + time +
                        "</td><td align=\"center\"> " + name +
                        "</td><td align=\"center\"> " + duration +
                        "</td><td align=\"center\"> " + guest1 +
                        "</td><td align=\"center\"> " + guest2 +
                        "</td><td align=\"center\"> " + guest3 +
                        "</td><td align=\"center\"> " + guest4 +
                        "</td><td align=\"center\"> " + guest5 + "</td>");
                   out.println("</tr>");
               }
               %>
          </table>
    </body>
</html>
```

add_movie_form.jsp

Below JSP file (strictly speaking it's just a static HTML file) generates an HTML for collection of user input to insert a new record into a movie_night table. Upon form submission, add_movie.jsp file will be invoked to process the user input.

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Add Movie Night</title>
    </head>
    <body>
        <h2>Add Movie Night</h2>
        <!--
            Form for collecting user input for the new movie_night record.
            Upon form submission, add_movie.jsp file will be invoked.
        -->
        <form action="add_movie.jsp">
            <!-- The form organized in an HTML table for better clarity. -->
            <table border=1>
                <tr>
                    <th colspan="2">Enter the Movie Night Data:</th>
                </tr>
                <tr>
                    <td>Movie night time:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=start_time>
                    </div></td>
                </tr>
                <tr>
                    <td>Movie Name:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=movie_name>
                    </div></td>
                </tr>
                <tr>
                    <td>Duration:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=duration_min>
                    </div></td>
                </tr>
                <tr>
                    <td>Guest 1 Name:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=guest_1>
                    </div></td>
                </tr>
                <tr>
                    <td>Guest 2 Name</td>
                    <td><div style="text-align: center;">
                    <input type=text name=guest_2>
                    </div></td>
                </tr>
                <tr>
                    <td>Guest 3 Name</td>
                    <td><div style="text-align: center;">
                    <input type=text name=guest_3>
                    </div></td>
                </tr>
                <tr>
                    <td>Guest 4 Name</td>
                    <td><div style="text-align: center;">
                    <input type=text name=guest_4>
                    </div></td>
                </tr>
```

```
                <tr>
                    <td>Guest 5 Name</td>
                    <td><div style="text-align: center;">
                    <input type=text name=guest_5>
                    </div></td>
                </tr>
                <tr>
                    <td><div style="text-align: center;">
                    <input type=reset value=Clear>
                    </div></td>
                    <td><div style="text-align: center;">
                    <input type=submit value=Insert>
                    </div></td>
                </tr>
            </table>
        </form>
    </body>
</html>
```

add_movie.jsp

Below JSP file processes the user request to insert a new record into movie_night table initiated by the add_movie_form.jsp file and generates the HTML file response confirming the insertion or notifying of the problem.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Query Result</title>
</head>
    <body>
    <%@page import="jsp_azure_test.DataHandler"%>
    <%@page import="java.sql.ResultSet"%>
    <%@page import="java.sql.Array"%>
    <%
    // The handler is the one in charge of establishing the connection.
    DataHandler handler = new DataHandler();

    // Get the attribute values passed from the input form.
    String startTime = request.getParameter("start_time");
    String movieName = request.getParameter("movie_name");
    String durationString = request.getParameter("duration_min");
    String g1 = request.getParameter("guest_1");
    String g2 = request.getParameter("guest_2");
    String g3 = request.getParameter("guest_3");
    String g4 = request.getParameter("guest_4");
    String g5 = request.getParameter("guest_5");

    /*
     * If the user hasn't filled out all the time, movie name and duration. This is very
simple checking.
     */
    if (startTime.equals("") || movieName.equals("") || durationString.equals("")) {
        response.sendRedirect("add_movie_form.jsp");
    } else {
        int duration = Integer.parseInt(durationString);

        // Now perform the query with the data from the form.
        boolean success = handler.addMovie(startTime, movieName, duration, g1, g2, g3,
g4, g5);
        if (!success) { // Something went wrong
            %>
```

```
                <h2>There was a problem inserting the course</h2>
            <%
        } else { // Confirm success to the user
            %>
            <h2>The Movie Night:</h2>

            <ul>
                <li>Start Time: <%=startTime%></li>
                <li>Movie Name: <%=movieName%></li>
                <li>Duration: <%=durationString%></li>
                <li>Guest 1: <%=g1%></li>
                <li>Guest 2: <%=g2%></li>
                <li>Guest 3: <%=g3%></li>
                <li>Guest 4: <%=g4%></li>
                <li>Guest 5: <%=g5%></li>
            </ul>

            <h2>Was successfully inserted.</h2>

            <a href="get_all_movies.jsp">See all movie nights.</a>
            <%
        }
    }
    %>
    </body>
</html>
```