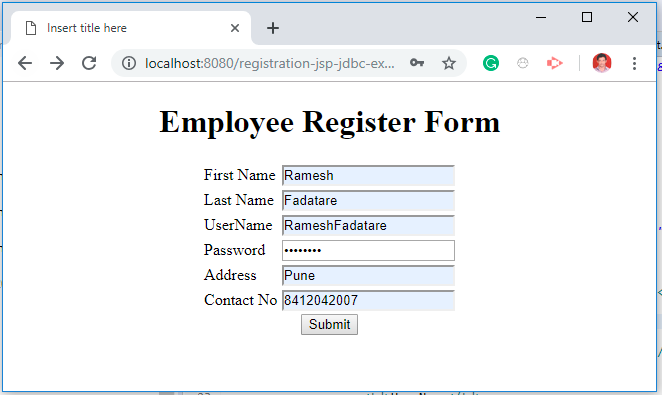
Below diagram shows **the Employee Registration** JSP page:

**[](https://4.bp.blogspot.com/-KbNOD6lfPPk/XHoqNIqwpPI/AAAAAAAAFsg/-T3X_JFJGkwLl4qVhyDgH1752xE27SaLwCLcBGAs/s1600/register-employee-page.PNG)**

Let me list out the tools and technologies that I have used to develop this application.

**Tools and technologies used**

* JSP - 2.2 +
* IDE - STS/Eclipse Neon.3
* JDK - 1.8 or later
* Apache Tomcat - 8.5
* JSTL - 1.2.1
* Servlet API - 2.5
* MySQL - mysql-connector-java-8.0.13.jar

**Development Steps**

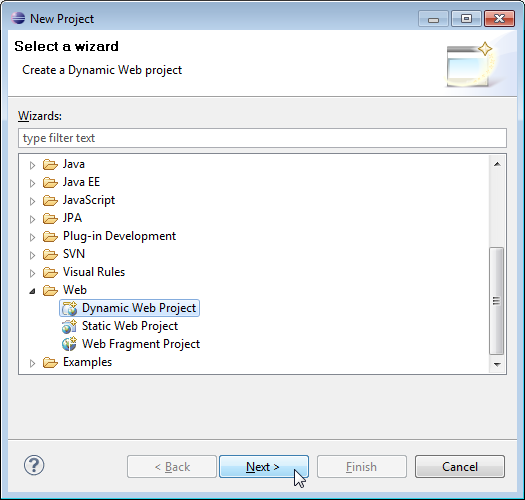
1. Create an Eclipse Dynamic Web Project
2. Add Dependencies
3. Project Structure
4. MySQL Database Setup
5. Create a JavaBean - Employee.java
6. Create a EmployeeDao.java
7. Create a EmployeeServlet.java
8. Create a employeeregister.jsp
9. Create a employeedetail.jsp
10. Demo

**1. Create an Eclipse Dynamic Web Project**

To create a new dynamic Web project in Eclipse:

1. On the main menu select **File > New > Project....**

2. In the upcoming wizard choose **Web > Dynamic Web Project.**

**[](https://3.bp.blogspot.com/-CeA278XIG4g/XDiv1ePWVlI/AAAAAAAAFZI/2mvCujNS2Co_gp1UqevtbZk0qhDsaa9DwCLcBGAs/s1600/create-web-proj-1.png)**

3. Click **Next**.

4. Enter project name as "jsp-servlet-jdbc-mysql-example";  
5. Make sure that the target runtime is set to Apache Tomcat with the currently supported version.

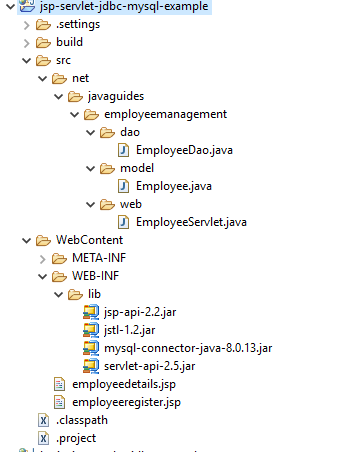
**2. Add Dependencies**

Add the latest release of below jar files to the *lib* folder.

* jsp-api.2.3.1.jar
* servlet-api.2.3.jar
* mysql-connector-java-8.0.13.jar
* jstl-1.2.jar

**3. Project Structure**

Standard project structure for your reference -

**[](https://4.bp.blogspot.com/-_BhczrcuB98/XHoq0l2EXGI/AAAAAAAAFso/XgmaqUc9IhICLEARyseremI-08q9IqC6QCLcBGAs/s1600/project-structure.PNG)**

**4. MySQL Database Setup**

Let's create a database named "**employees**" in MySQL. Now, let's create an **employee**table using below DDL script:

CREATE TABLE `employee` (

`id` int(3) NOT NULL AUTO\_INCREMENT,

`first\_name` varchar(20) DEFAULT NULL,

`last\_name` varchar(20) DEFAULT NULL,

`username` varchar(250) DEFAULT NULL,

`password` varchar(20) DEFAULT NULL,

`address` varchar(45) DEFAULT NULL,

`contact` varchar(45) DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

**5. Create a JavaBean - Employee.java**

Let's create an *Employee* JavaBean class which we will use in JSP action tags.

package net.javaguides.jsp.jdbc.bean;

import java.io.Serializable;

/\*\*

\* JavaBean class used in jsp action tags.

\* @author Ramesh Fadatare

\*/

public class Employee implements Serializable {

/\*\*

\*

\*/

private static final long serialVersionUID = 1 L;

private String firstName;

private String lastName;

private String username;

private String password;

private String address;

private String contact;

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getContact() {

return contact;

}

public void setContact(String contact) {

this.contact = contact;

}

}

**6. Create an EmployeeDao.java**

Let's create EmployeeDao class which contains JDBC code to connect with MySQL database. Add the following code to an EmployeeDao class:

package net.javaguides.jsp.jdbc.database;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import net.javaguides.jsp.jdbc.bean.Employee;

public class EmployeeDao {

public int registerEmployee(Employee employee) throws ClassNotFoundException {

String INSERT\_USERS\_SQL = "INSERT INTO employee" +

" (id, first\_name, last\_name, username, password, address, contact) VALUES " +

" (?, ?, ?, ?, ?,?,?);";

int result = 0;

Class.forName("com.mysql.jdbc.Driver");

try (Connection connection = DriverManager

.getConnection("jdbc:mysql://localhost:3306/employees?useSSL=false", "root", "root");

// Step 2:Create a statement using connection object

PreparedStatement preparedStatement = connection.prepareStatement(INSERT\_USERS\_SQL)) {

preparedStatement.setInt(1, 1);

preparedStatement.setString(2, employee.getFirstName());

preparedStatement.setString(3, employee.getLastName());

preparedStatement.setString(4, employee.getUsername());

preparedStatement.setString(5, employee.getPassword());

preparedStatement.setString(6, employee.getAddress());

preparedStatement.setString(7, employee.getContact());

System.out.println(preparedStatement);

// Step 3: Execute the query or update query

result = preparedStatement.executeUpdate();

} catch (SQLException e) {

// process sql exception

printSQLException(e);

}

return result;

}

private void printSQLException(SQLException ex) {

for (Throwable e: ex) {

if (e instanceof SQLException) {

e.printStackTrace(System.err);

System.err.println("SQLState: " + ((SQLException) e).getSQLState());

System.err.println("Error Code: " + ((SQLException) e).getErrorCode());

System.err.println("Message: " + e.getMessage());

Throwable t = ex.getCause();

while (t != null) {

System.out.println("Cause: " + t);

t = t.getCause();

}

}

}

}

}

You can learn complete JDBC at [**JDBC Tutorial**](http://www.javaguides.net/p/jdbc-tutorial.html)

**7. Create an EmployeeServlet.java**

 Let's create an EmployeeServlet class to process HTTP request parameters and redirect to the appropriate JSP page after request data stored in the database:

package net.javaguides.employeemanagement.web;

import java.io.IOException;

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 net.javaguides.employeemanagement.dao.EmployeeDao;

import net.javaguides.employeemanagement.model.Employee;

/\*\*

\* @email Ramesh Fadatare

\*/

@WebServlet("/register")

public class EmployeeServlet extends HttpServlet {

private static final long serialVersionUID = 1 L;

private EmployeeDao employeeDao;

public void init() {

employeeDao = new EmployeeDao();

}

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String firstName = request.getParameter("firstName");

String lastName = request.getParameter("lastName");

String username = request.getParameter("username");

String password = request.getParameter("password");

String address = request.getParameter("address");

String contact = request.getParameter("contact");

Employee employee = new Employee();

employee.setFirstName(firstName);

employee.setLastName(lastName);

employee.setUsername(username);

employee.setPassword(password);

employee.setContact(contact);

employee.setAddress(address);

try {

employeeDao.registerEmployee(employee);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

response.sendRedirect("employeedetails.jsp");

}

}

**8. Create a employeeregister.jsp**

Let's design employee registration HTML form with the following fields:

* firstName
* lastName
* username
* password
* address
* contact

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

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

<title>Insert title here</title>

</head>

<body>

<div align="center">

<h1>Employee Register Form</h1>

<form action="<%= request.getContextPath() %>/register" method="post">

<table style="with: 80%">

<tr>

<td>First Name</td>

<td><input type="text" name="firstName" /></td>

</tr>

<tr>

<td>Last Name</td>

<td><input type="text" name="lastName" /></td>

</tr>

<tr>

<td>UserName</td>

<td><input type="text" name="username" /></td>

</tr>

<tr>

<td>Password</td>

<td><input type="password" name="password" /></td>

</tr>

<tr>

<td>Address</td>

<td><input type="text" name="address" /></td>

</tr>

<tr>

<td>Contact No</td>

<td><input type="text" name="contact" /></td>

</tr>

</table>

<input type="submit" value="Submit" />

</form>

</div>

</body>

</html>

**9. Create an employeedetails.jsp**

After an employee successfully registered then this page show a successful message on screen:

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@page import="net.javaguides.employeemanagement.dao.\*"%>

<!DOCTYPE html>

<html>

<head>

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

<title>Insert title here</title>

</head>

<body>

<h1>User successfully registered!</h1>

</body>

</html>

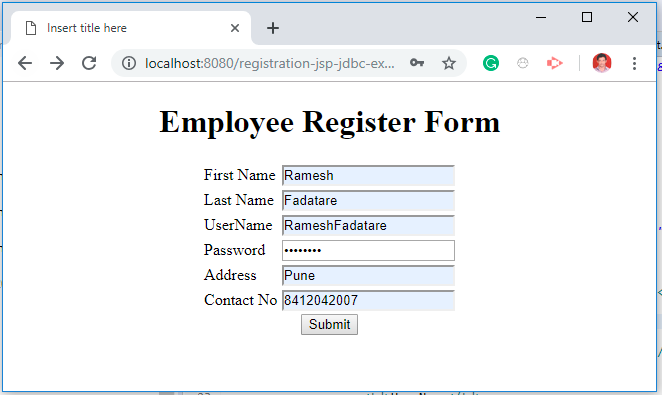
Note that in the above page, we have used JSP action tags. Read more about action tags here.

**10. Demo**

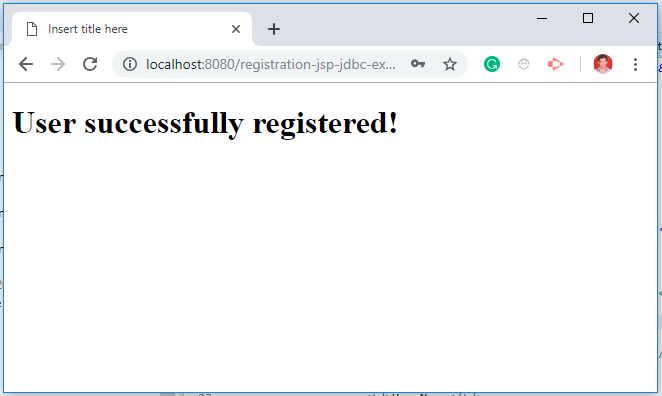
It's time to see a demo of the above development. Deploy this web application in tomcat server.

**Employee Registration**

Once you deploy this application successfully then hit this link into a browser - [**http://localhost:8080/jsp-servlet-jdbc-mysql-example/employeeregister.jsp**](http://localhost:8080/jsp-jdbc-mysql-example/employeeregister.jsp)

**[](https://3.bp.blogspot.com/-KbNOD6lfPPk/XHoqNIqwpPI/AAAAAAAAFsk/CHFe5cJQdxkhkkF1dPqfcSd5yDztmMiTQCEwYBhgL/s1600/register-employee-page.PNG)**

**Registration Success Page**

**[](https://1.bp.blogspot.com/-hjtT2h_uvvI/XHosOJkch0I/AAAAAAAAFs0/OEk4lW-w0VgJ0Ydsk3TRvgbHrS1j5ZrmgCLcBGAs/s1600/register-employee-output.PNG)**