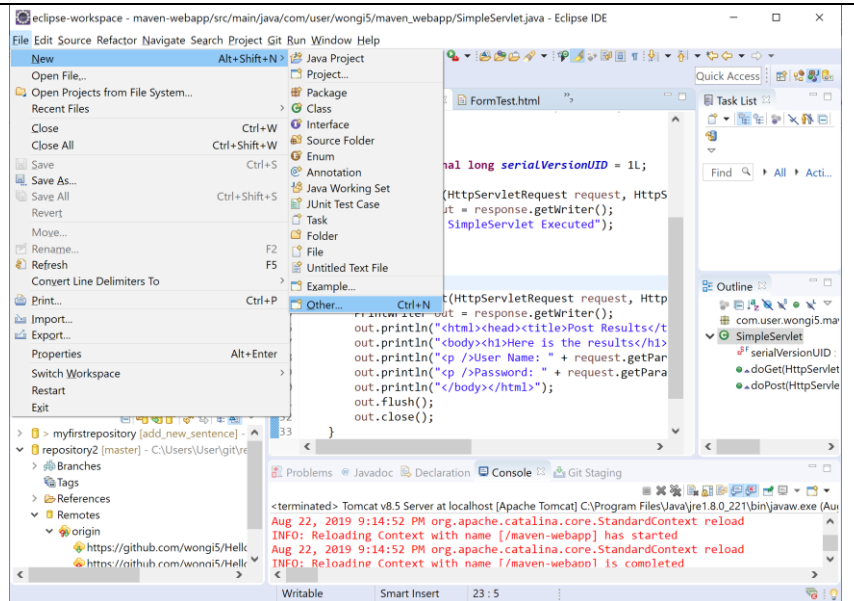


Lab 03: Applying Spring Framework

(Ref: <https://www.javaguides.net/2018/10/spring-mvc-5-hello-world-example.html>
<https://www.javaguides.net/2018/10/spring-mvc-sign-up-form-handling.html>)

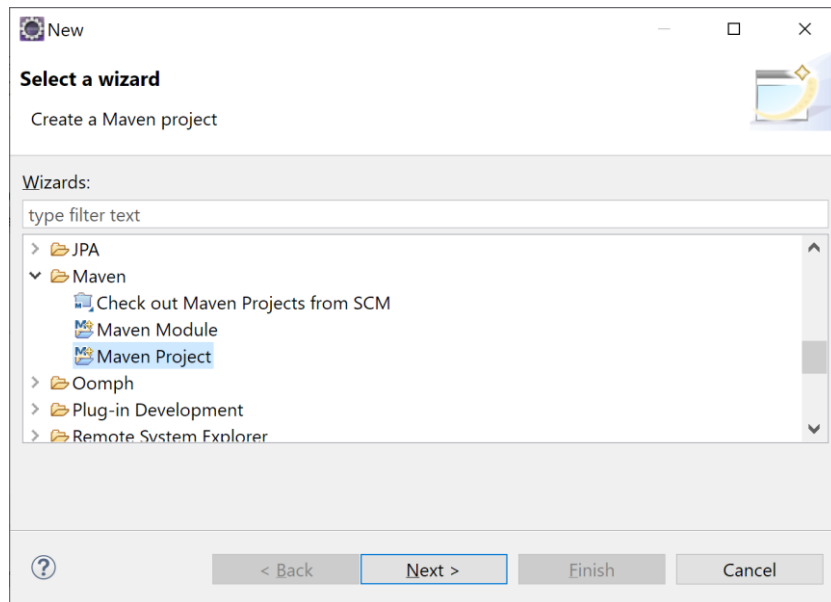
Step 1: Creating a Maven project

Start Eclipse



Choose File → New → Project

Choose Maven → Maven Project, then Next



Use the default Workspace location or choose another preferred one, then Next

Choose the item with Artifact Id “maven-archetype-webapp”, then Next

Fill in the Group Id and Artifact Id, which form the Package of the project, then Finish

New Maven Project

New Maven project

Select project name and location

☐ Create a simple project (skip archetype selection)

☒ Use default Workspace location

Location: C:\Users\User\workspace\maven-webapp\src\main\java\com\user\wongj5\maven_webapp\SimpleServlet.java [Browse...](#)

☐ Add project(s) to working set

Working set: [More...](#)

▶ Advanced

[?](#) < Back **Next >** Finish Cancel

New Maven Project

New Maven project

Select an Archetype

Catalog: All Catalogs [Configure...](#)

Filter:

Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-portlet	1.0.1
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	1.1
org.apache.maven.archetypes	maven-archetype-site-simple	1.1
org.apache.maven.archetypes	maven-archetype-webapp	1.0

An archetype which contains a sample Maven Webapp project.

☒ Show the last version of Archetype only ☐ Include snapshot archetypes [Add Archetype...](#)

▶ Advanced

[?](#) < Back **Next >** Finish Cancel

New Maven Project

New Maven project

Specify Archetype parameters

Group Id: com.user.wongj5

Artifact Id: **springdemo**

Version: 0.0.1-SNAPSHOT

Package: com.user.wongj5.springdemo

Properties available from archetype:

Name	Value

[Add...](#)
[Remove](#)

▶ Advanced

[?](#) < Back Next > **Finish** Cancel

<p>Step 2: Add dependencies in pom.xml</p> <p>Open pom.xml</p> <p>Add dependencies as shown</p>	<pre> <properties> <spring.version>5.1.12.RELEASE</spring.version> <jstl.version>1.2.1</jstl.version> <taglibs.version>1.1.2</taglibs.version> <servlet.version>3.1.0</servlet.version> <jsp.version>2.3.1</jsp.version> <jetty.version>9.3.28.v20191105</jetty.version> <jdk.version>1.8</jdk.version> <maven.version>3.5.1</maven.version> </properties> <dependencies> <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc --> <dependency> <groupId>org.springframework</groupId> <artifactId>spring-webmvc</artifactId> <version>\${spring.version}</version> </dependency> <dependency> <groupId>javax.annotation</groupId> <artifactId>javax.annotation-api</artifactId> <version>1.3.2</version> </dependency> <!-- JSTL Dependency --> <dependency> <groupId>javax.servlet.jsp.jstl</groupId> <artifactId>javax.servlet.jsp.jstl- api</artifactId> <version>\${jstl.version}</version> </dependency> <dependency> <groupId>taglibs</groupId> <artifactId>standard</artifactId> <version>\${taglibs.version}</version> </dependency> <!-- Servlet Dependency --> <dependency> <groupId>javax.servlet</groupId> <artifactId>javax.servlet-api</artifactId> <version>\${servlet.version}</version> <scope>provided</scope> </dependency> <!-- JSP Dependency --> <dependency> <groupId>javax.servlet.jsp</groupId> <artifactId>javax.servlet.jsp-api</artifactId> <version>\${jsp.version}</version> <scope>provided</scope> </dependency> </dependencies> </pre>
---	---

Add plugins as shown	<pre><plugins> <plugin> <artifactId>maven-compiler- plugin</artifactId> <version>\${maven.version}</version> <configuration> <source>\${jdk.version}</source> <target>\${jdk.version}</target> </configuration> </plugin> <plugin> <groupId>org.eclipse.jetty</groupId> <artifactId>jetty-maven- plugin</artifactId> <version>\${jetty.version}</version> </plugin> </plugins></pre>

Step 3: Configure Spring

Right-click the project, and add a new Java class called AppConfig.java as shown

```
package com.user.wongi5.springdemo.demo.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.view.InternalResourceViewResolver;
import org.springframework.web.servlet.view.JstlView;

//The @Configuration is a class-level annotation
//indicating
//that an object is a source of bean definitions.
//
//The @EnableWebMvc enables default Spring MVC
//configuration
//and provides the functionality equivalent to
//mvc:annotation-driven/ element in XML based
//configuration.
//
//The @ComponentScan scans the stereotype annotations
//(@Controller, @Service etc...) in a package
//specified by
//basePackages attribute.

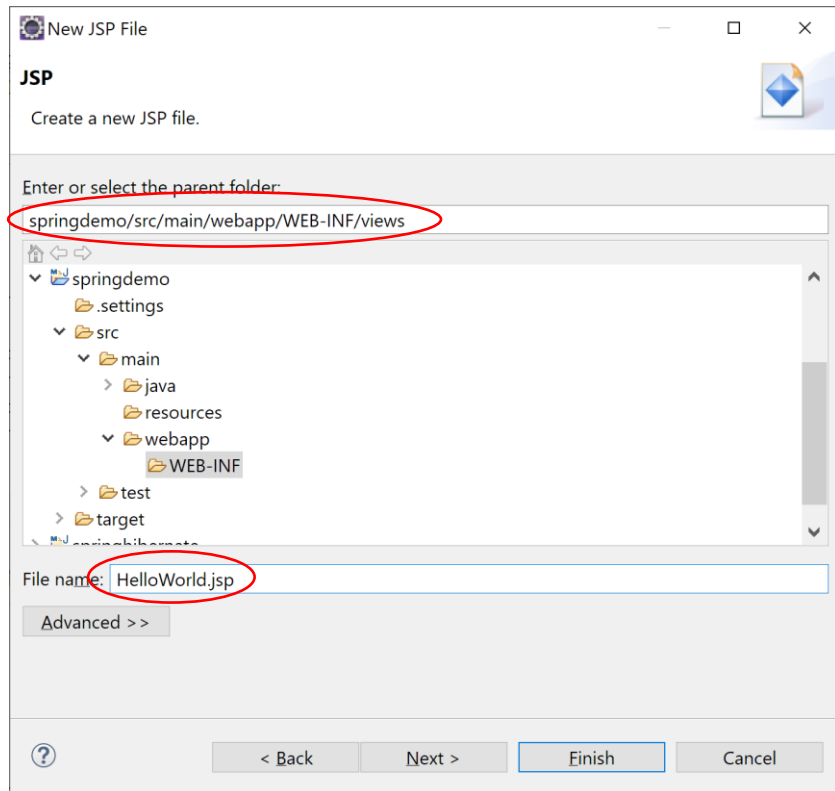
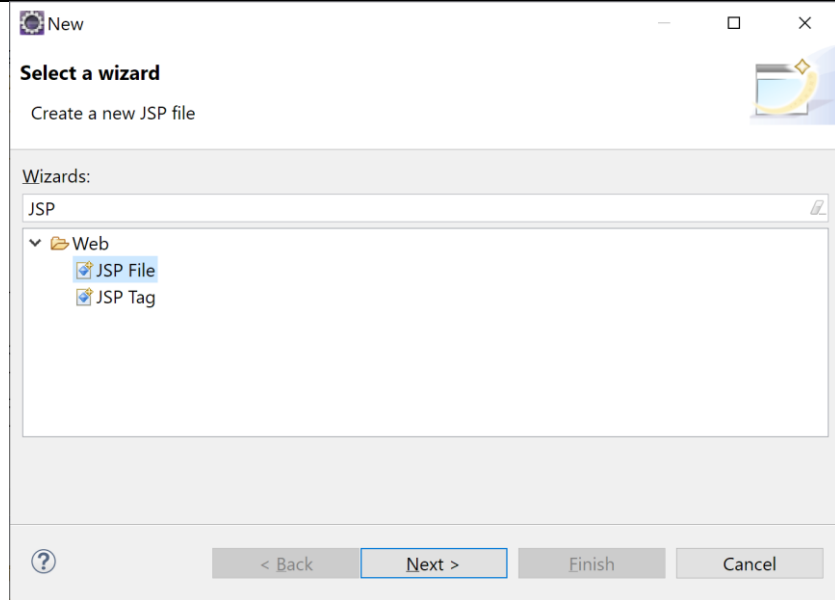
@Configuration
@EnableWebMvc
@ComponentScan(basePackages = {
    "com.user.wongi5.springdemo.demo"
})
public class AppConfig implements WebMvcConfigurer {

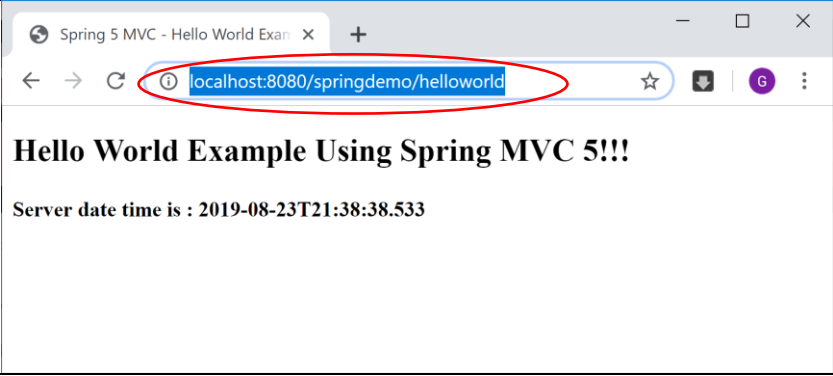
    @Bean
    public InternalResourceViewResolver resolver() {
        InternalResourceViewResolver resolver = new
        InternalResourceViewResolver();
        resolver.setViewClass(JstlView.class);
        resolver.setPrefix("/WEB-INF/views/");
        resolver.setSuffix(".jsp");
        return resolver;
    }
}
```

<p>Right-click the project and add another Java class called SpringMvcDispatcherServletInitializer.java as shown</p>	<pre> package com.user.wongi5.springdemo.demo.config; import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer; public class SpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer { @Override protected Class<?>[] getRootConfigClasses() { // TODO Auto-generated method stub return null; } @Override protected Class<?>[] getServletConfigClasses() { return new Class[] { AppConfig.class }; } @Override protected String[] getServletMappings() { return new String[] { "/" }; } } </pre>
<p>Step 4: Add Model class</p> <p>Right-click the project, and add a new Java class called HelloWorld.java as shown</p>	<pre> package com.user.wongi5.springdemo.demo.model; public class HelloWorld { private String message; private String dateTime; public String getMessage() { return message; } public void setMessage(String message) { this.message = message; } public String getDateTime() { return dateTime; } public void setDateTime(String dateTime) { this.dateTime = dateTime; } } </pre>

<p>Step 5: Add Controller class</p> <p>Right-click the project, and add a new Java class called HelloWorldController.java as shown</p>	<pre> package com.user.wongi5.springdemo.demo.controller; import java.time.LocalDateTime; import org.springframework.stereotype.Controller; import org.springframework.ui.Model; import org.springframework.web.bind.annotation.RequestMapping; import com.user.wongi5.springdemo.demo.model.HelloWorld; @Controller public class HelloWorldController { @RequestMapping("/helloworld") public String handler(Model model) { HelloWorld helloWorld = new HelloWorld(); helloWorld.setMessage("Hello World Example Using Spring MVC 5!!!"); helloWorld.setDateTime(LocalDateTime.now().toSt ring()); model.addAttribute("helloworld", helloWorld); return "helloworld"; } } </pre>
<p>Step 6: Add view JSP</p> <p>Right-click the project, choose New → Other</p> <p>Choose JSP File under Web, then Next</p>	

Create an helloworld.jsp file under
src/main/webapp/WEB-INF/views folder as shown,
then Finish



	<pre><%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%> <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%> <!DOCTYPE html> <html> <head><%@ page isELIgnored="false"%> <meta charset="ISO-8859-1"> <title>Spring 5 MVC - Hello World Example javaguides.net</title> </head> <body> <h2>\${helloWorld.message}</h2> <h4>Server date time is : \${helloWorld.dateTime}</h4> </body> </html></pre>
<p>Step 7: Test the demo</p> <p>Run the application in tomcat</p> <p>Open browser and go to the URL: http://localhost:8080/springdemo/helloworld</p>	 <p>Spring 5 MVC - Hello World Exan x +</p> <p>localhost:8080/springdemo/helloworld</p> <p>Hello World Example Using Spring MVC 5!!!</p> <p>Server date time is : 2019-08-23T21:38:38.533</p>

<p>Step 8: Add another model</p> <p>Create another model called SignUpForm.java</p>	<pre>package com.user.wongi5.springdemo.demo.model; public class SignUpForm { private String firstName; private String lastName; private String email; private String userName; private String password; 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 getEmail() { return email; } public void setEmail(String email) { this.email = email; } 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; } }</pre>
---	---

Step 9: Create another controller

```
package com.user.wongi5.springdemo.demo.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import com.user.wongi5.springdemo.demo.model.SignUpForm;

@Controller
public class SignUpController {

    /**
     * Create new signUpForm object for empty form
     *
     * @return
     */
    @ModelAttribute("signUpForm")
    public SignUpForm setSignUpForm() {
        return new SignUpForm();
    }

    /**
     * Method to show the initial HTML form
     *
     * @return
     */
    @GetMapping("/showSignUpForm")
    public String showForm() {
        return "signup-form";
    }
}
```

	<pre> /** * Save User sign up form * * @param signUpForm * @param model * @return */ @PostMapping("/saveSignUpForm") public String saveUser(@ModelAttribute("signUpForm") SignUpForm signUpForm, Model model) { // Implement business logic to save user details into a database // System.out.println("FirstName : " + signUpForm.getFirstName()); System.out.println("LastName : " + signUpForm.getLastName()); System.out.println("Username : " + signUpForm.getUserName()); System.out.println("Password : " + signUpForm.getPassword()); System.out.println("Email : " + signUpForm.getEmail()); model.addAttribute("message", "User SignUp successfully."); model.addAttribute("user", signUpForm); return "signup-success"; } </pre>
Step 10: Create signup-form.jsp	<pre> <%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%> <%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%> <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%> <!DOCTYPE html> <html> <head> <meta charset="ISO-8859-1"> <title>Spring MVC 5 - form handling Java Guides</title> <link href="<c:url value="/resources/css/bootstrap.min.css" />" rel="stylesheet"> <script src="<c:url value="/resources/js/jquery- 1.11.1.min.js" />"></script> </pre>

```

<script src="<c:url
value="/resources/js/bootstrap.min.js" />"></script>

</head>
<body>
<div class="container">
  <div class="col-md-offset-2 col-md-7">
    <h2 class="text-center">Spring MVC 5 Form Handling
Example -
    Sign up Form</h2>
    <div class="panel panel-info">
      <div class="panel-heading">
        <div class="panel-title">Sign Up</div>
      </div>
      <div class="panel-body">
        <form:form action="saveSignUpForm" cssClass="form-
horizontal"
        method="post" modelAttribute="signUpForm">

          <div class="form-group">
            <label for="firstname" class="col-md-3 control-
label">First
            Name</label>
            <div class="col-md-9">
              <form:input path="firstName" cssClass="form-
control" />
            </div>
          </div>
          <div class="form-group">
            <label for="lastname" class="col-md-3 control-
label">Last
            Name</label>
            <div class="col-md-9">
              <form:input path="lastName" cssClass="form-
control" />
            </div>
          </div>
          <div class="form-group">
            <label for="icode" class="col-md-3 control-
label">User
            Name </label>
            <div class="col-md-9">
              <form:input path="userName" cssClass="form-
control" />
            </div>
          </div>
          <div class="form-group">
            <label for="password" class="col-md-3 control-
label">Password</label>
            <div class="col-md-9">
              <form:password path="password" cssClass="form-
control" />
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>

```

	<pre> </div> <div class="form-group"> <label for="email" class="col-md-3 control-label">Email</label> <div class="col-md-9"> <form:input path="email" cssClass="form-control" /> </div> </div> <div class="form-group"> <!-- Button --> <div class="col-md-offset-3 col-md-9"> <form:button cssClass="btn btn-primary">Submit</form:button> </div> </div> </form:form> </div> </div> </div> </body> </html> </pre>
Step 11: Create signup-success.jsp	<pre> <%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%> <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%> <!DOCTYPE html> <html> <head> <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"> <title>javaguides.net</title> <link href="<c:url value="/resources/css/bootstrap.min.css" />" rel="stylesheet"> <script src="<c:url value="/resources/js/jquery- 1.11.1.min.js" />"></script> <script src="<c:url value="/resources/js/bootstrap.min.js" />"></script> <%@ page isELIgnored="false" %> </head> <body> <div class="container"> <div class="col-md-offset-2 col-md-7"> <h1>\${message}</h1> <hr /> </pre>

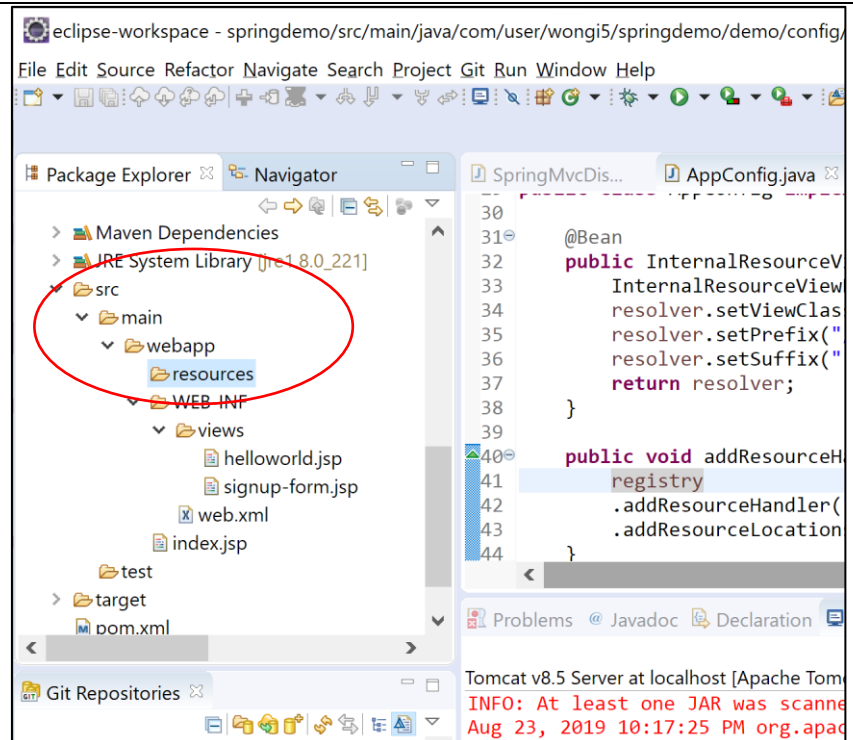
```

<table class="table table-striped table-bordered">
<tr>
<td><b>First Name </b>: ${user.firstName}</td>
</tr>
<tr>
<td><b>Last Name </b> : ${user.lastName}</td>
</tr>
<tr>
<td><b>UserName </b> : ${user.userName}</td>
</tr>
<tr>
<td><b>Email </b>: ${user.email}</td>
</tr>
</table>
</div>
</div>
</body>
</html>

```

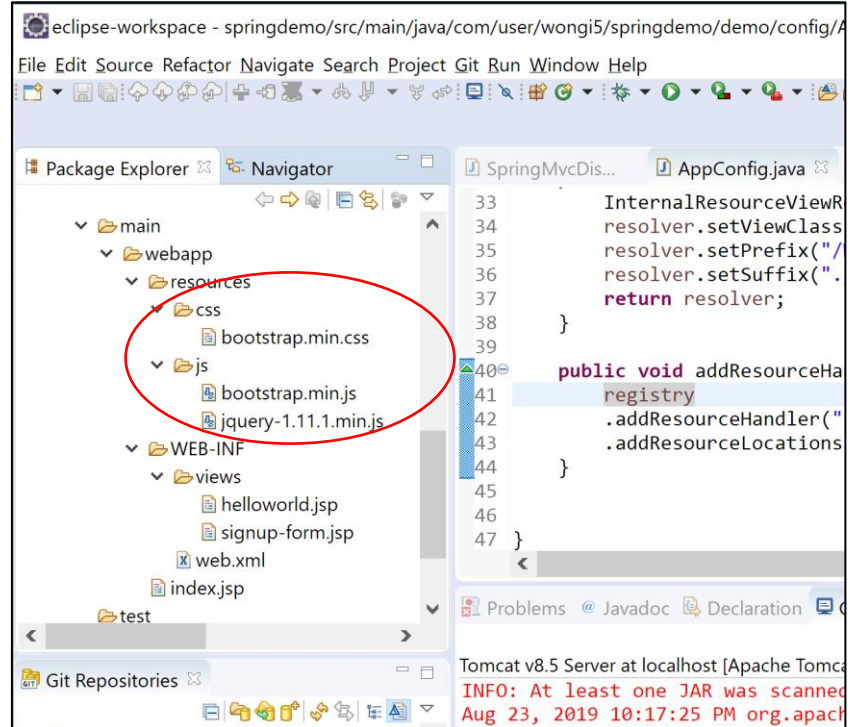
Step 12: Add CSS and JS resources

Right-click
src/main/webapp/, choose
New → Folder and create a
folder called resources



Create two sub-folders called css and js under src/main/webapp/resources

Drag-and-drop the following files to the folders as shown:
bootstrap.min.css
bootstrap.min.js
jquery-1.11.1.min.js



Add the
addResourceHandlers
method in AppConfig.java

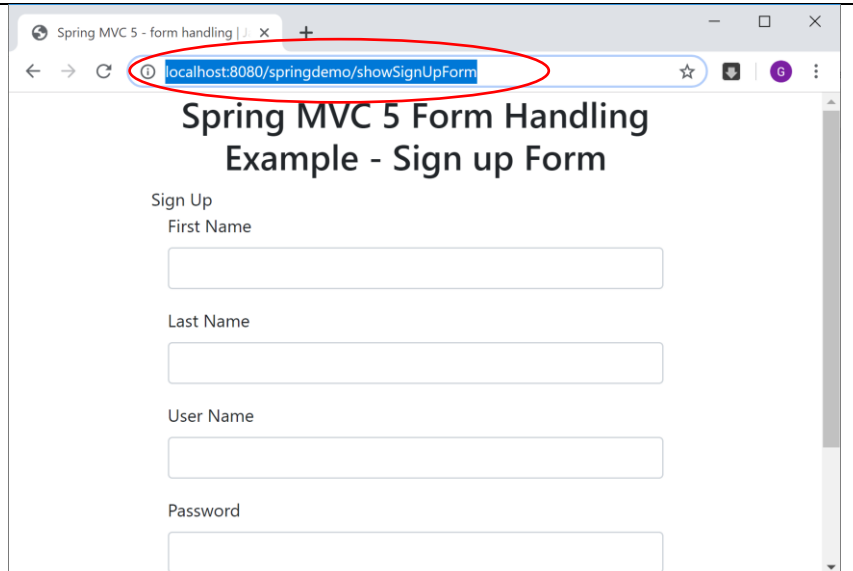
```
public void
addResourceHandlers(ResourceHandlerRegistry registry)
{
    registry
        .addResourceHandler("/resources/**")
        .addResourceLocations("/resources/");
}
```

Step 13: Test the signup
demo

Run the application in
tomcat

Open browser and go to the
URL:
<http://localhost:8080/springdemo/showSignUpForm>

Fill in the info and test



	<div><div>javaguides.net</div><div>localhost:8080/springdemo/saveSignUpForm</div><div>User SignUp successfully.</div><div><div>First Name : Ivan</div><div>Last Name : Wong</div><div>UserName : wongi5</div><div>Email : wongi5@douglascollege.ca</div></div></div>