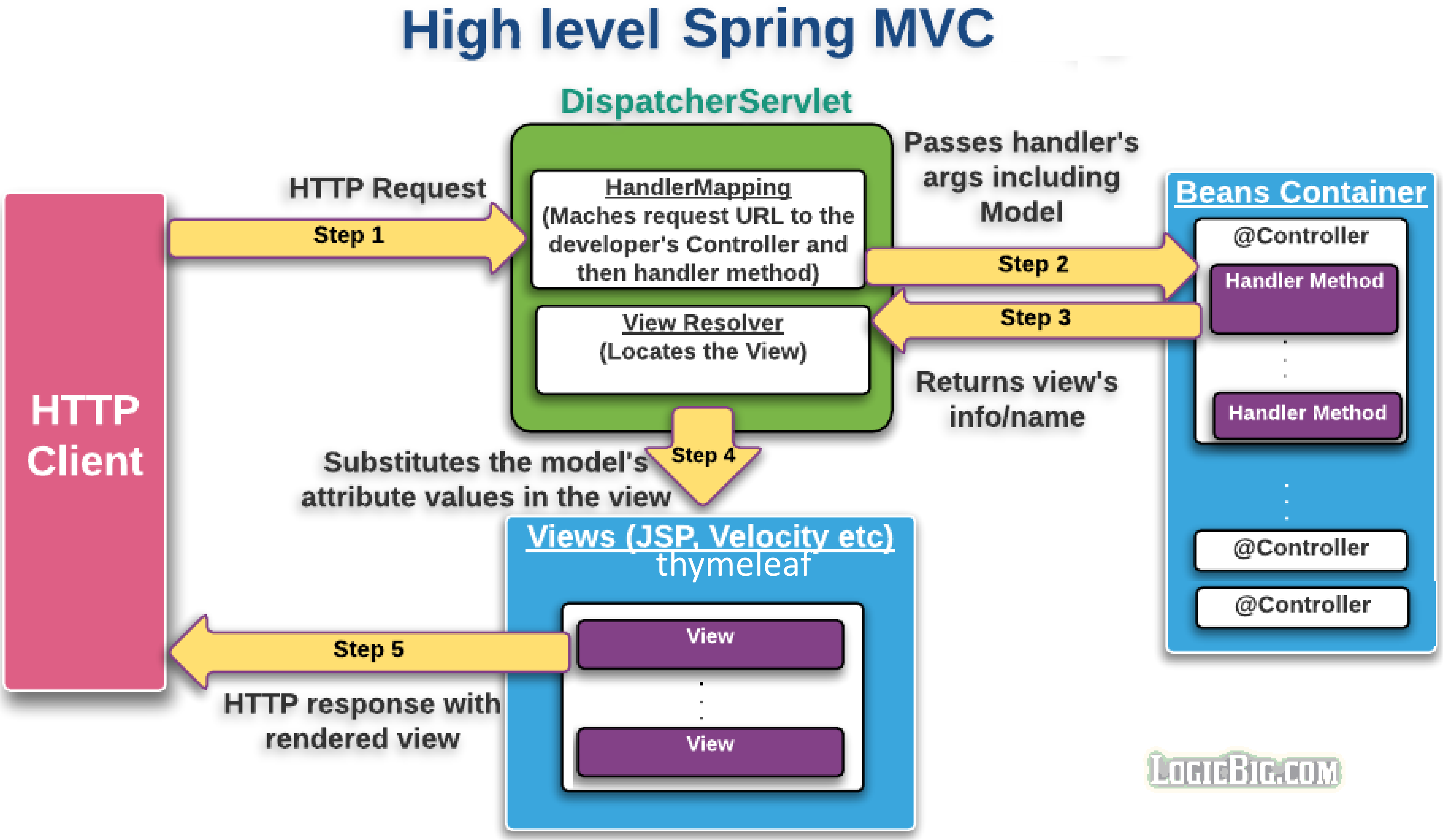
Spring MVC

An intro to Spring MVC with template engine

A spring application

- Controllers
  - Handle the logic
  - Receive HTTP requests
  - Manipulate data/objects and store them (session, cookies, database…)
  - Inject data into a view
  - Returns the view (response)
- Views
  - Html pages
  - Augmented html: special tags to insert dynamic content (the data injected from the controllers)
- Model
  - The objects used to pass data to the view

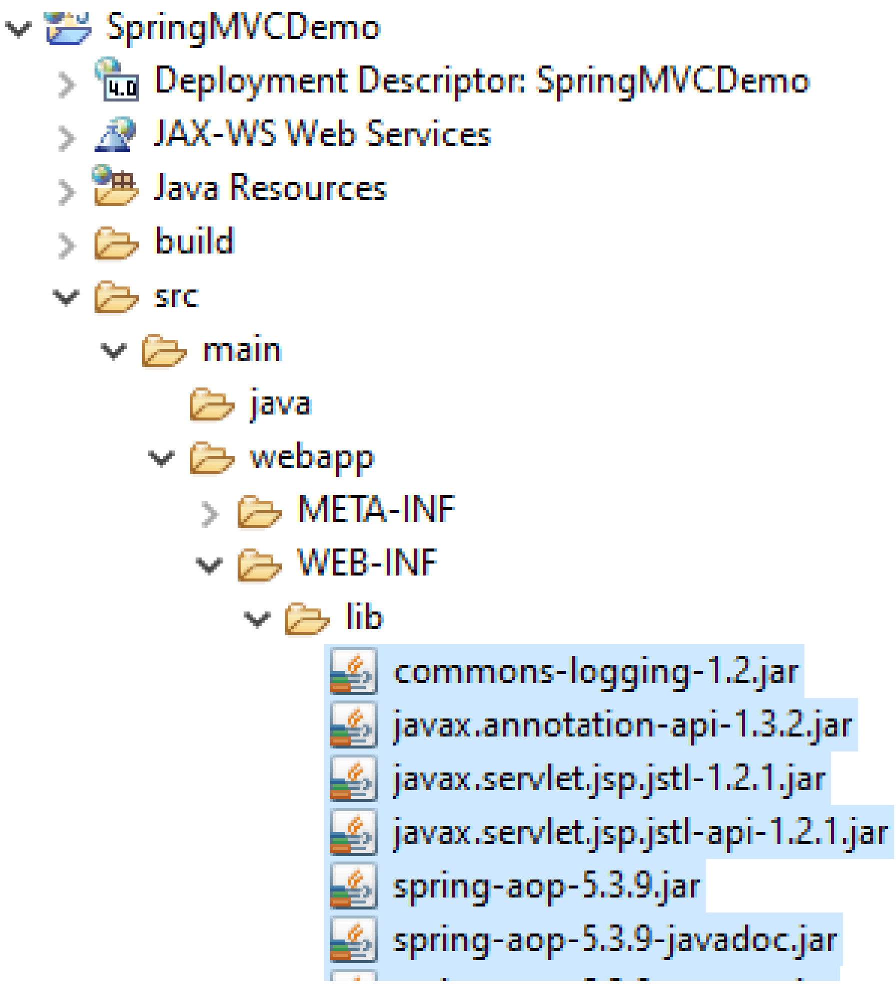
Request / response



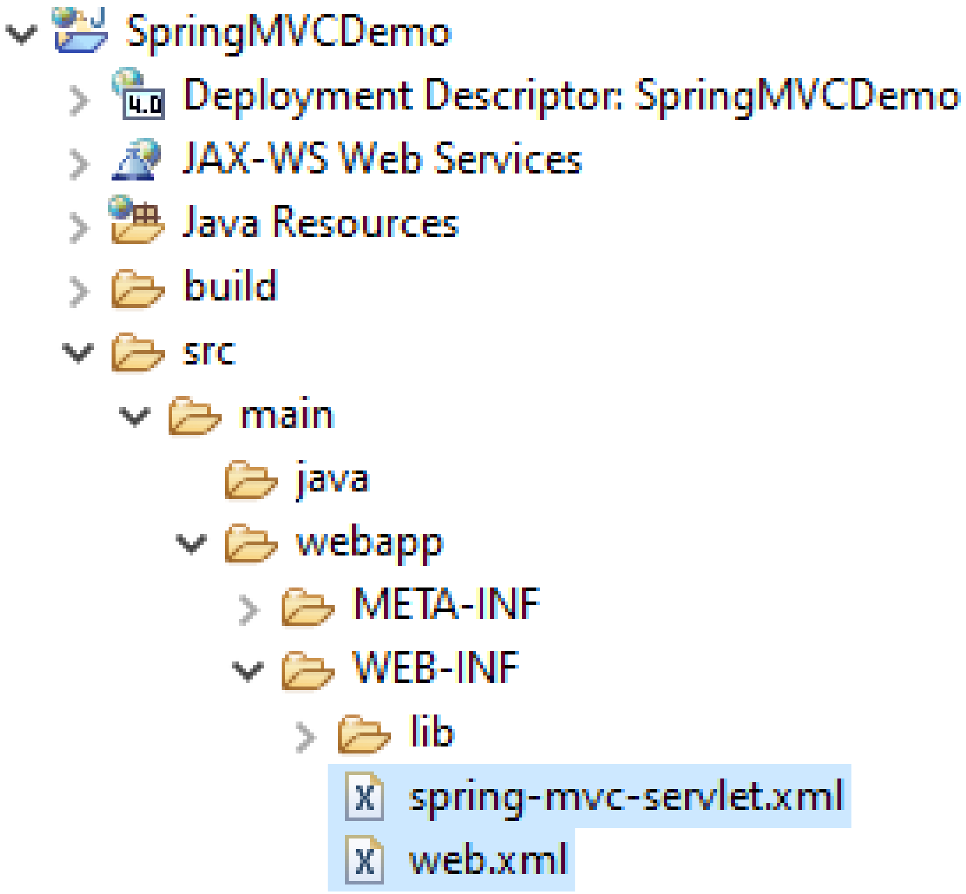
Project Setup

- Create a Dynamic Web Project
- Add spring jars in WEB-INF/lib
- Add java-servlet-api jars also in WEB-INF/lib
    - Javax.servlet.jsp.jstl-1.2.1.jar
    - Javax.servlet.jsp.jstl-api-1.2.1.jar
- Add Configuration files in WEB-INF
    - spring-mvc-servlet.xml and web.xml

Project Setup with spring jars and config files



```
SpringMVCDemo
  Deployment Descriptor: SpringMVCDemo
  JAX-WS Web Services
  Java Resources
  build
  src
    main
      java
      webapp
        META-INF
        WEB-INF
          lib
            commons-logging-1.2.jar
            javax.annotation-api-1.3.2.jar
            javax.servlet.jsp.jstl-1.2.1.jar
            javax.servlet.jsp.jstl-api-1.2.1.jar
            spring-aop-5.3.9.jar
            spring-aop-5.3.9-javadoc.jar
```



```
SpringMVCDemo
  Deployment Descriptor: SpringMVCDemo
  JAX-WS Web Services
  Java Resources
  build
  src
    main
      java
      webapp
        META-INF
        WEB-INF
          lib
          spring-mvc-servlet.xml
          web.xml
```

Project Setup :

Instead of adding jars, we can do it with maven as follows:

- Create a Dynamic Web Project
- Rclick Project->Configure->Convert To maven
- Include following dependencies
  - spring-webmvc
  - jakarta.servlet-api
  - javax.annotation-api
  - commons-logging
  - jakarta.servlet.jsp.jstl-api
- Add Configuration files in WEB-INF
  - spring-mvc-servlet.xml and web.xml

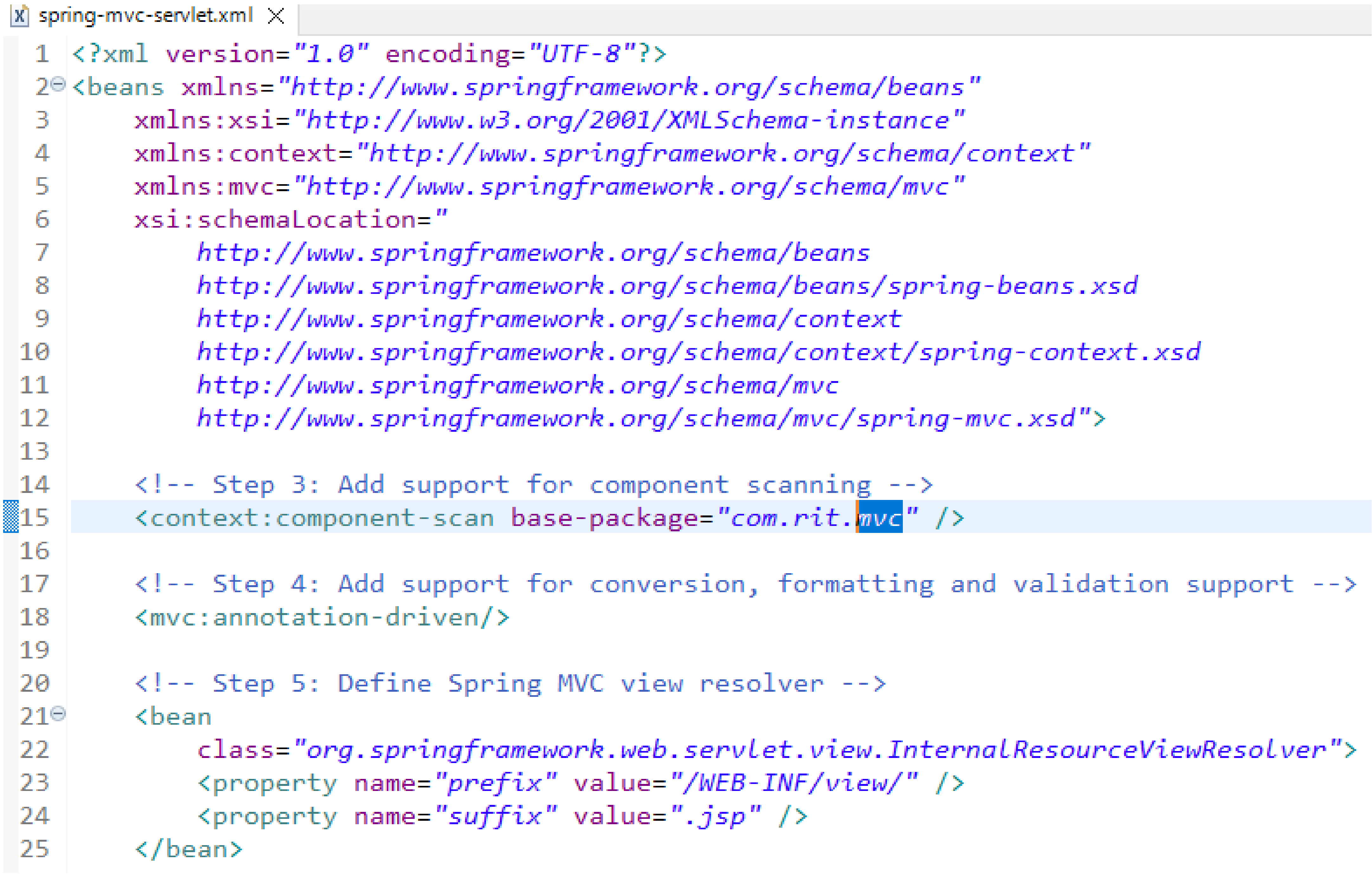
Alternative Maven Project Setup

Maven dependencies

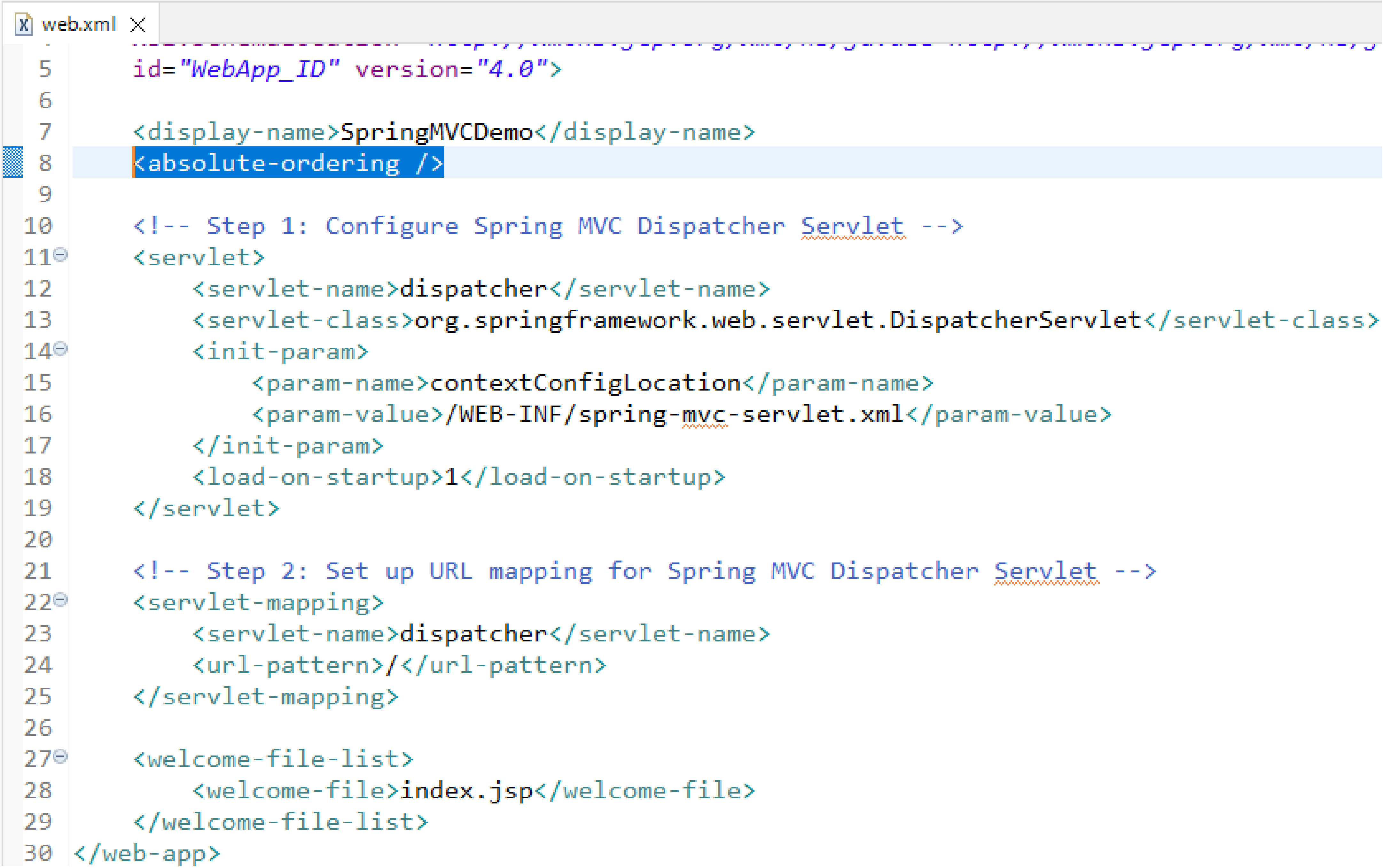
Alternative Maven Project Setup

```
<dependency>
	<groupId>org.springframework</groupId>
	<artifactId>spring-web</artifactId>
	<version>6.0.11</version>
</dependency>
<dependency>
	<groupId>org.springframework</groupId>
	<artifactId>spring-webmvc</artifactId>
	<version>6.0.11</version>
</dependency>
<dependency>
	<groupId>jakarta.servlet</groupId>
	<artifactId>jakarta.servlet-api</artifactId>
	<version>6.0.0</version>
	<scope>provided</scope>
</dependency>
```

```
<dependency>
	<groupId>javax.annotation</groupId>
	<artifactId>javax.annotation-api</artifactId>
	<version>1.3.2</version>
</dependency>
<dependency>
	<groupId>commons-logging</groupId>
	<artifactId>commons-logging</artifactId>
	<version>1.1.2</version>
</dependency>
<dependency>
	<groupId>jakarta.servlet.jsp.jstl</groupId>
	<artifactId>jakarta.servlet.jsp.jstl-api</artifactId>
	<version>2.0.0</version>
</dependency>
```



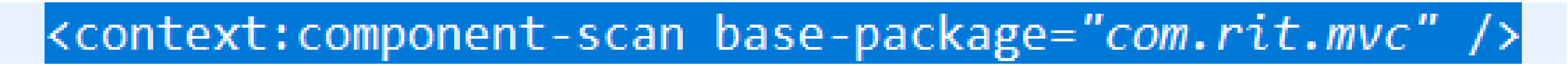
```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context.xsd
        http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc.xsd">

    <!-- Step 3: Add support for component scanning -->
    <context:component-scan base-package="com.rit.mvc" />

    <!-- Step 4: Add support for conversion, formatting and validation support -->
    <mvc:annotation-driven/>

    <!-- Step 5: Define Spring MVC view resolver -->
    <bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/view/" />
        <property name="suffix" value=".jsp" />
    </bean>
```



```xml
    id="WebApp_ID" version="4.0">

    <display-name>SpringMVCDemo</display-name>
    <absolute-ordering />

    <!-- Step 1: Configure Spring MVC Dispatcher Servlet -->
    <servlet>
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-mvc-servlet.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

    <!-- Step 2: Set up URL mapping for Spring MVC Dispatcher Servlet -->
    <servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>

    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

Project Setup

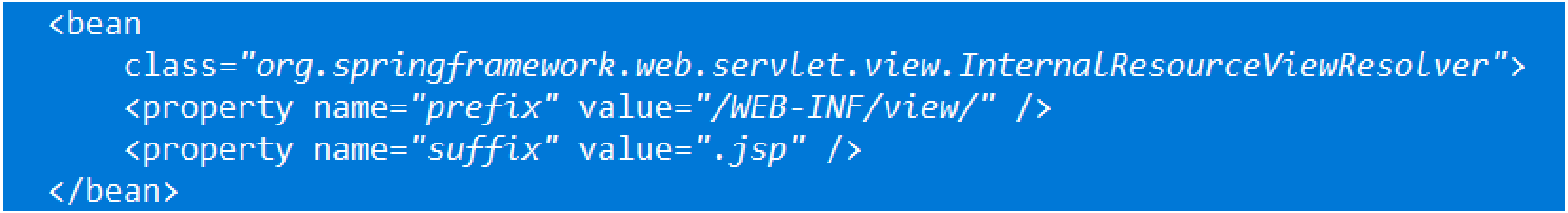
- Create a package com.rit.mvc as defined in servlet config file



```
<context:component-scan base-package="com.rit.mvc" />
```

  - We will create all controller, model & service classes here

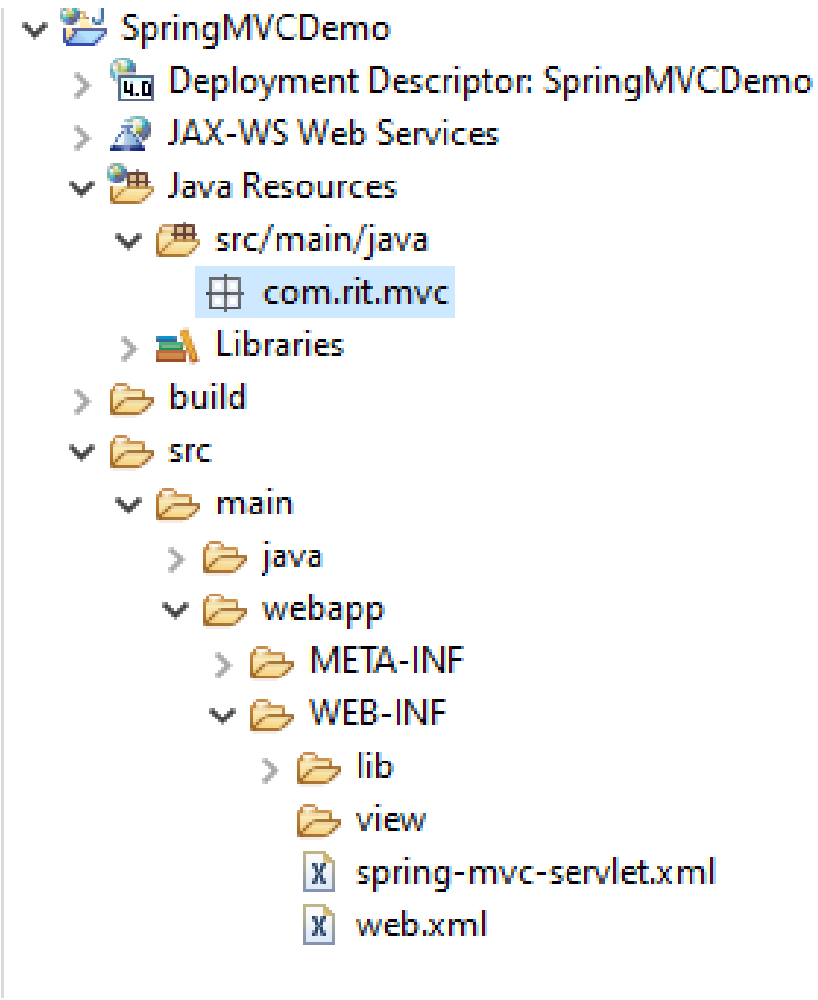
- Create a folder view in WEB-INF folder as defined in servlet config file



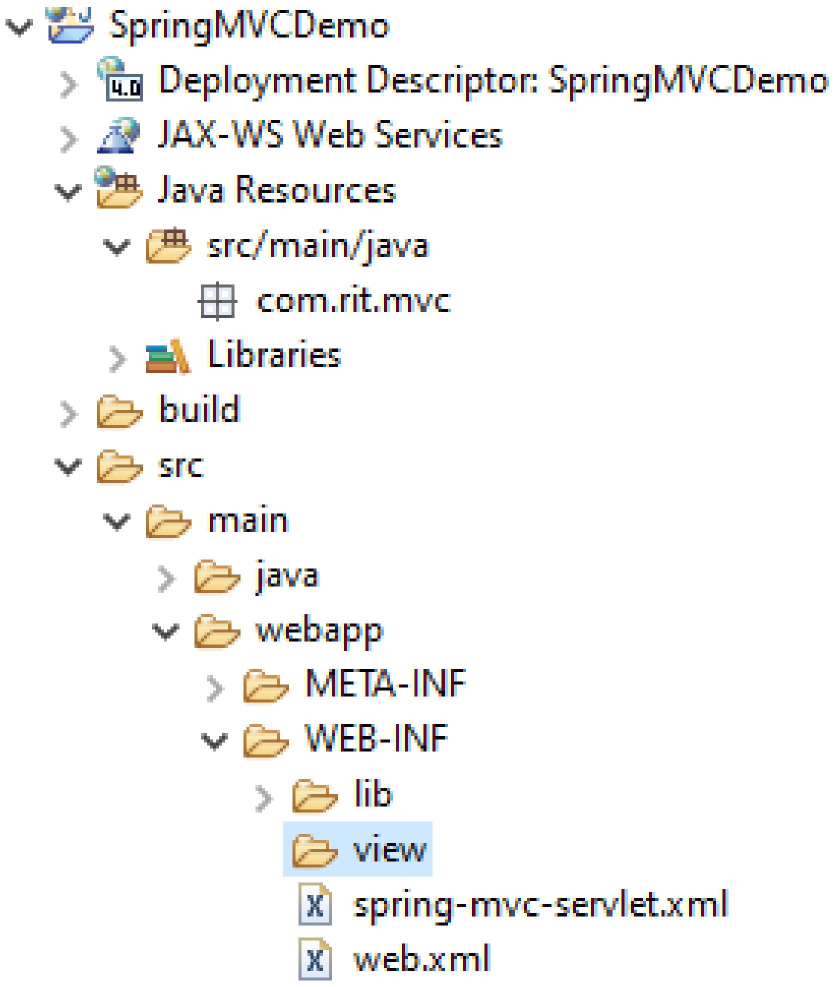
```
<bean
    class="org.springframework.web.servlet.view.InternalResourceViewResolver">
    <property name="prefix" value="/WEB-INF/view/" />
    <property name="suffix" value=".jsp" />
</bean>
```

  - We will create all the jsp files within this folder

Package & View folder created

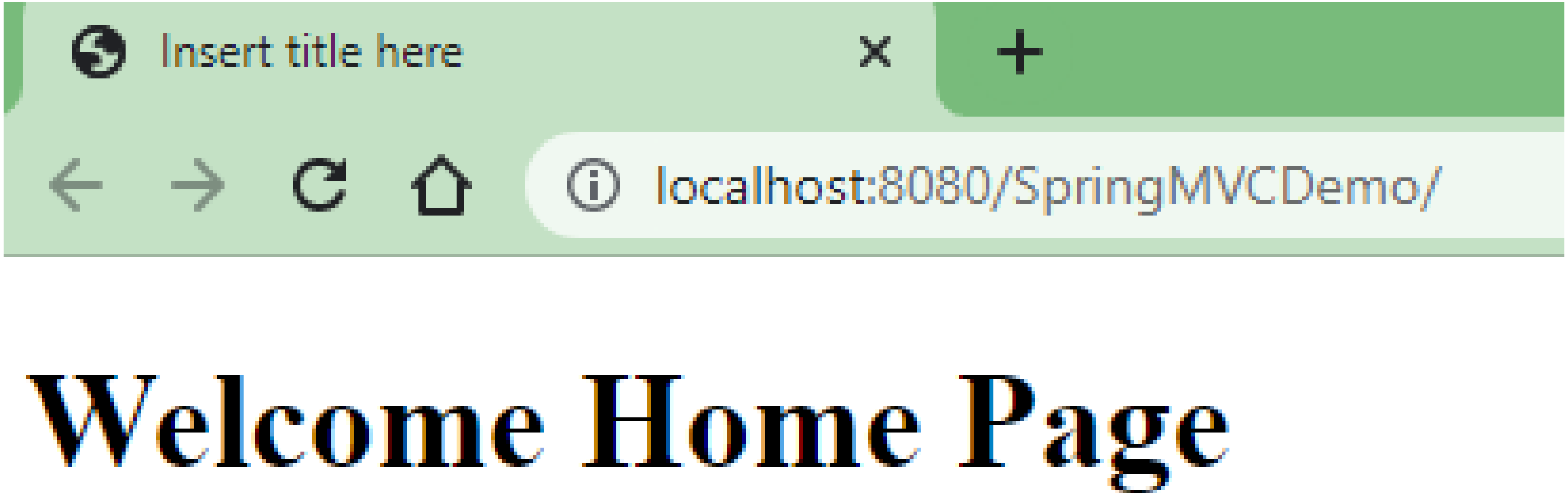


```
SpringMVCDemo
  Deployment Descriptor: SpringMVCDemo
  JAX-WS Web Services
  Java Resources
    src/main/java
      com.rit.mvc
    Libraries
  build
  src
    main
      java
      webapp
        META-INF
        WEB-INF
          lib
          view
          spring-mvc-servlet.xml
          web.xml
```



```
SpringMVCDemo
  Deployment Descriptor: SpringMVCDemo
  JAX-WS Web Services
  Java Resources
    src/main/java
      com.rit.mvc
    Libraries
  build
  src
    main
      java
      webapp
        META-INF
        WEB-INF
          lib
          view
          spring-mvc-servlet.xml
          web.xml
```

Lets Develop this

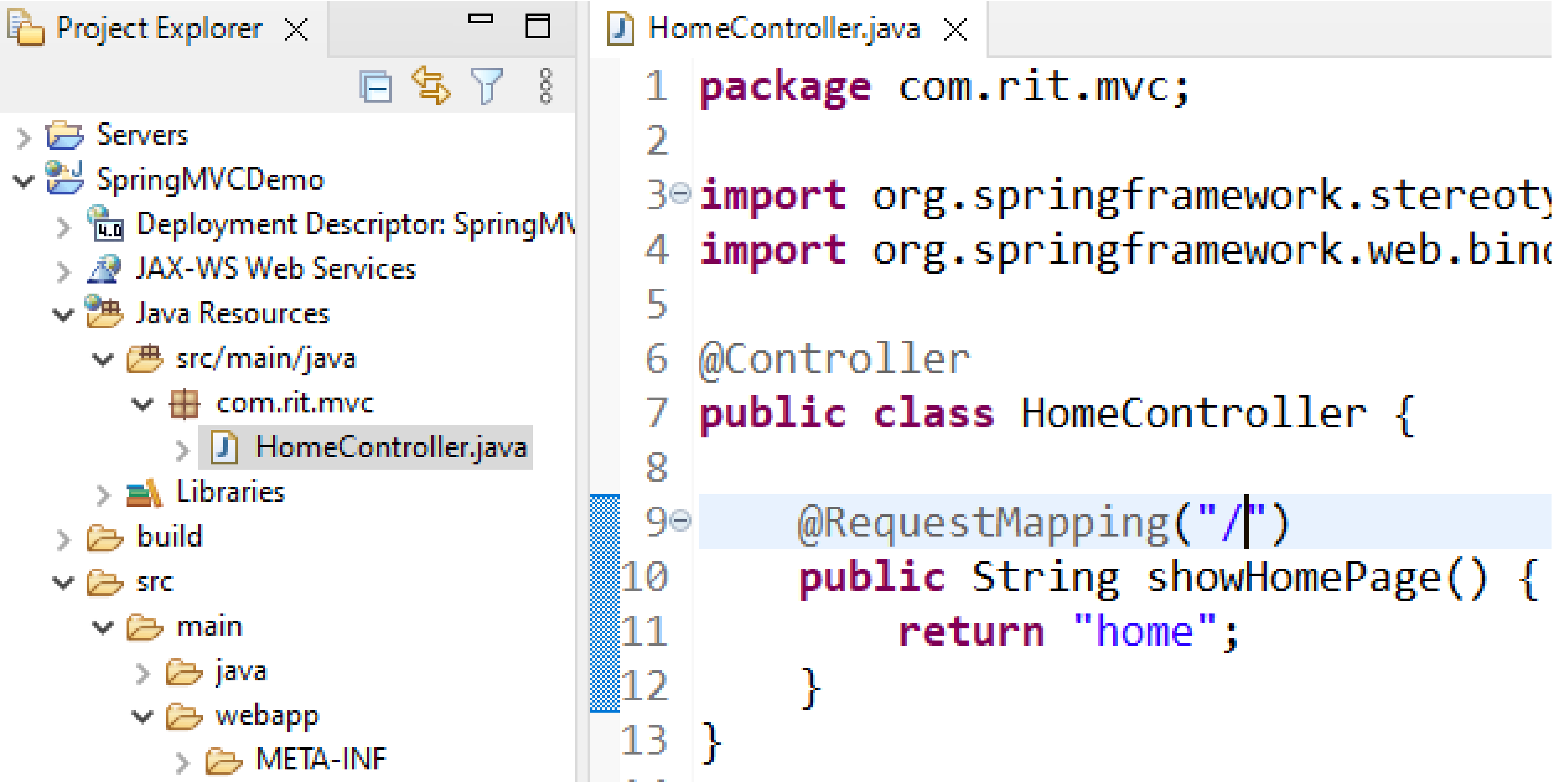


Welcome Home Page

Development Process : First Controller

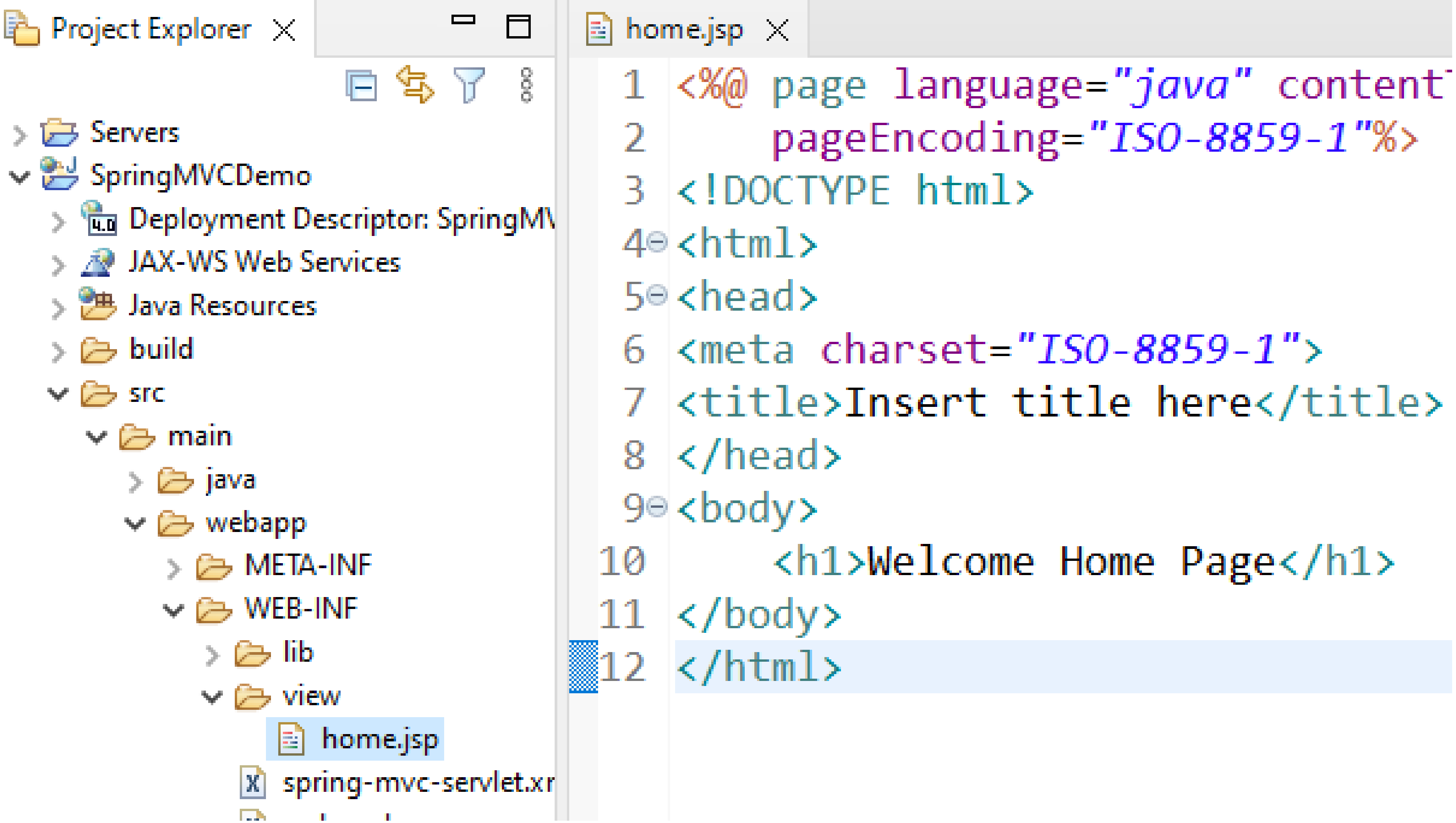
- Create a Controller Class
- Create a Controller Method
- Add Request Mapping to the Controller Method
- Return View Name
- Develop View Page

Controller Class



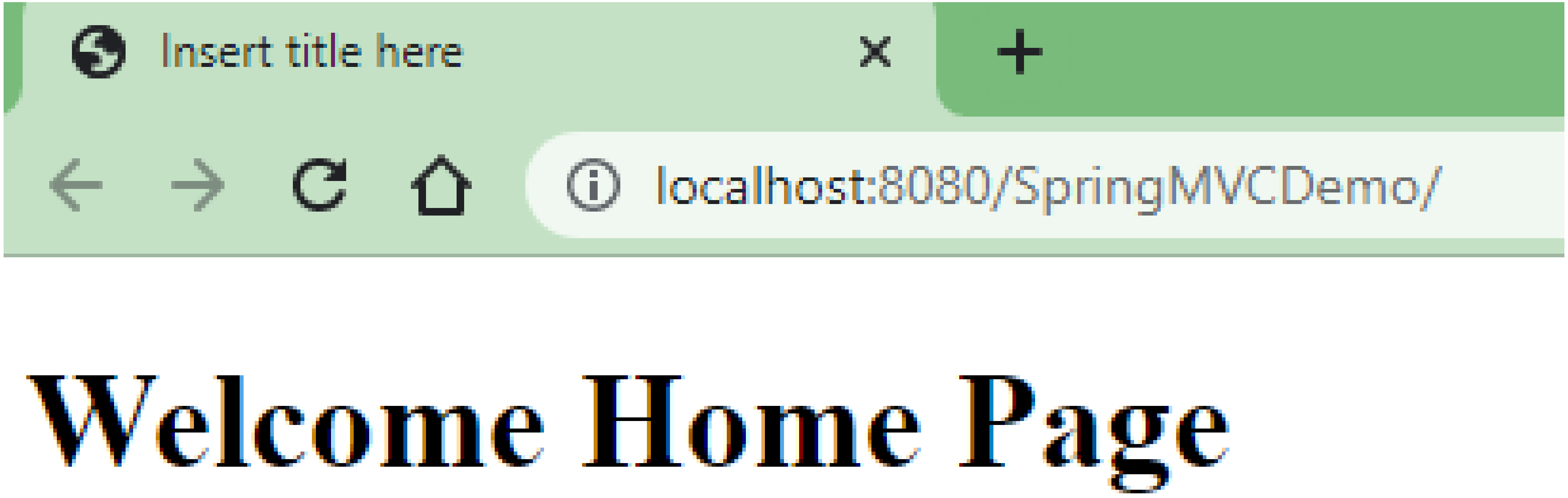
```java
package com.rit.mvc;

import org.springframework.stereoty
import org.springframework.web.binc

@Controller
public class HomeController {

    @RequestMapping("/")
    public String showHomePage() {
        return "home";
    }
}
```

View Page



```
<%@ page language="java" content
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <h1>Welcome Home Page</h1>
</body>
</html>
```

Project -> Run as Server

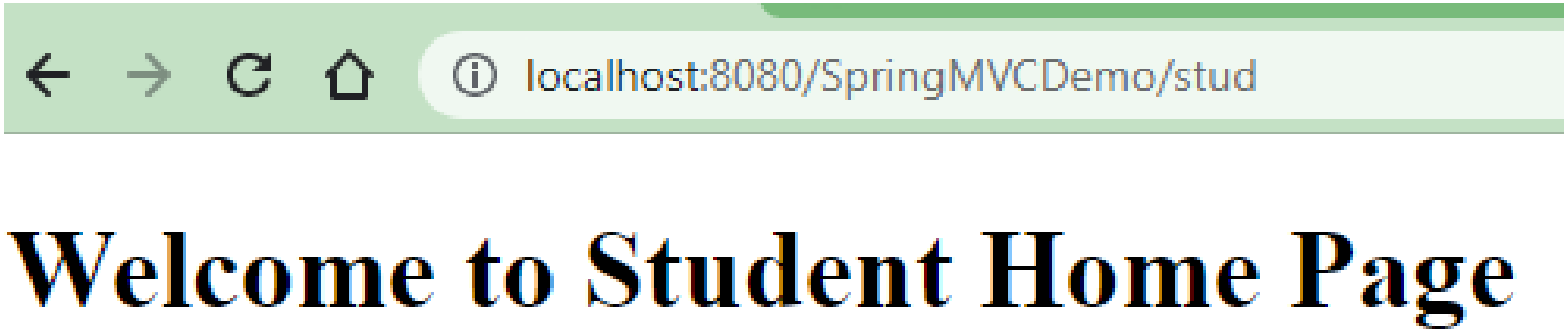


localhost:8080/SpringMVCDemo/

Welcome Home Page

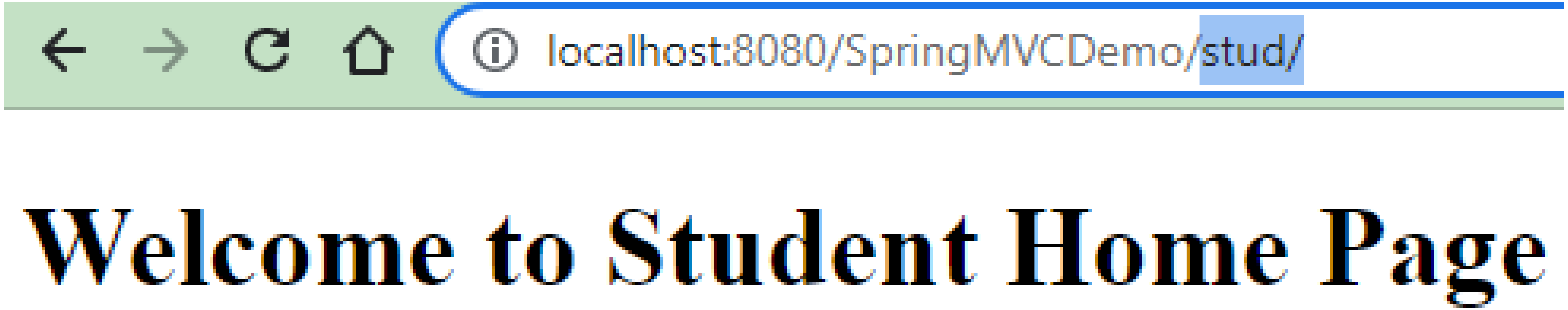
Controller Level Request Mapping

Lets develop it with controller level ReqMap…



localhost:8080/SpringMVCDemo/stud

Welcome to Student Home Page



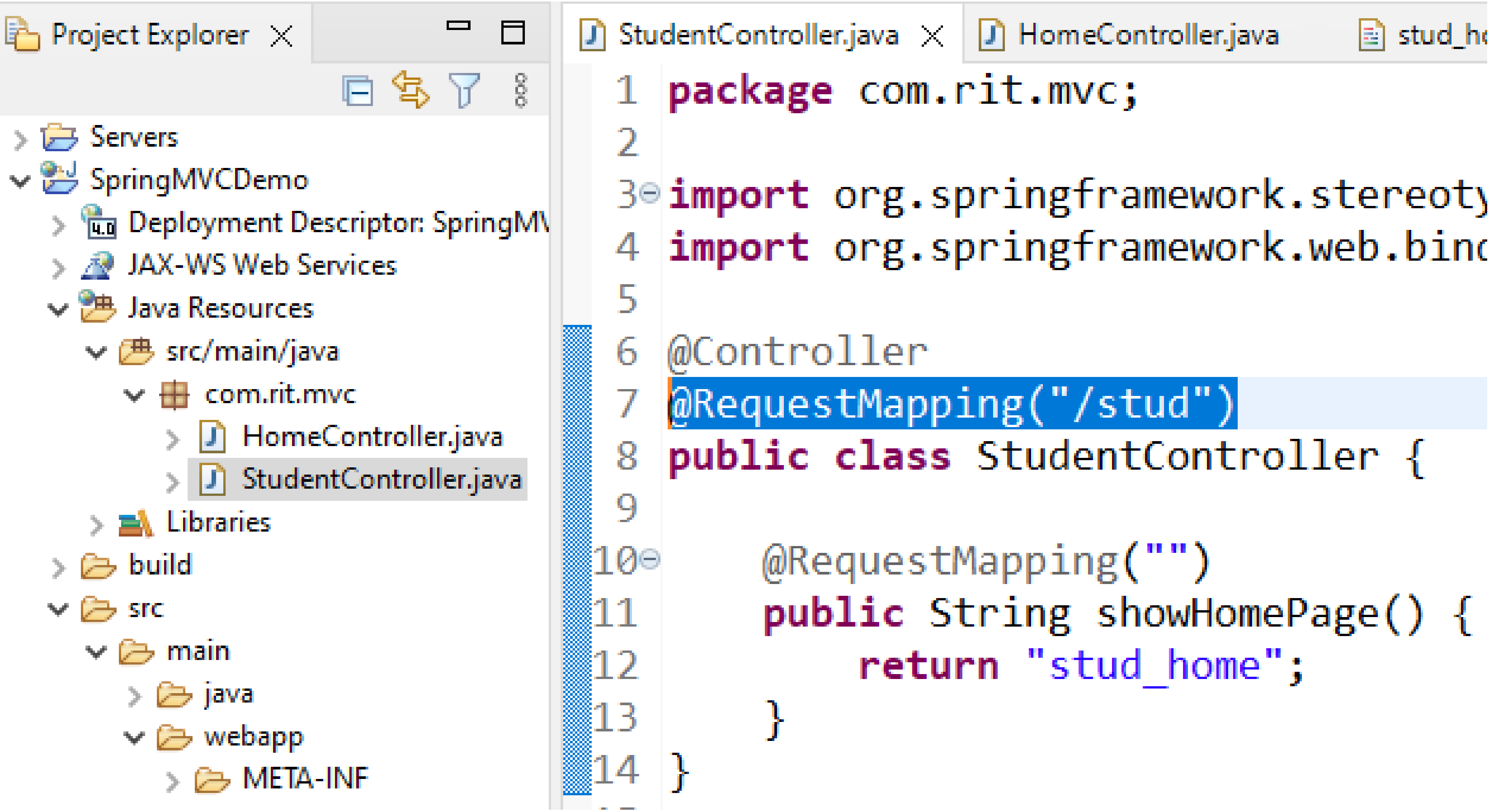
localhost:8080/SpringMVCDemo/stud/

Welcome to Student Home Page

Development Process : First Controller

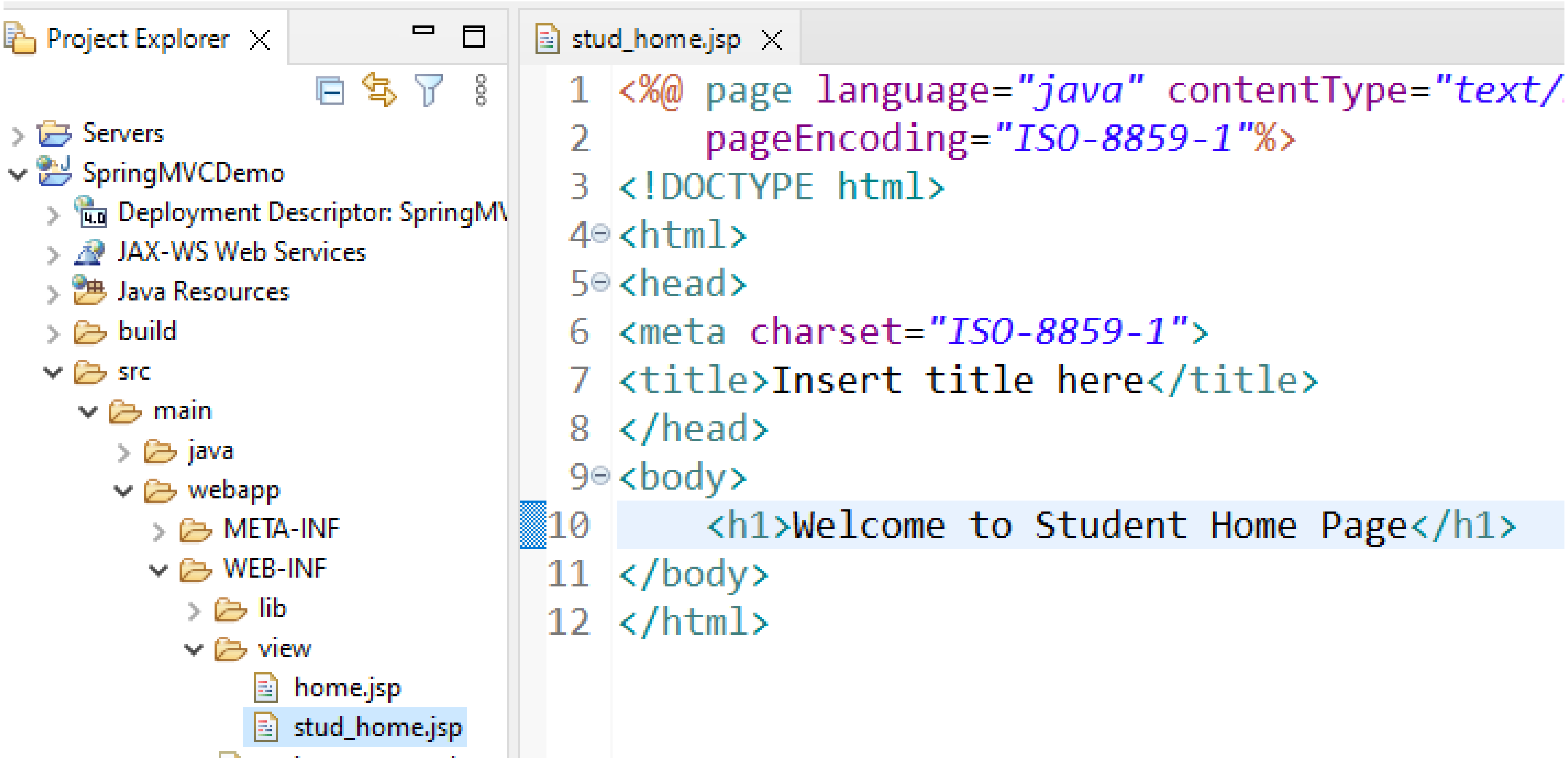
- Create a Controller Class
- Add Request Mapping to the Controller Class
- Create a Controller Method
- Add Request Mapping to the Controller Method
- Return View Name
- Develop View Page

Controller Class



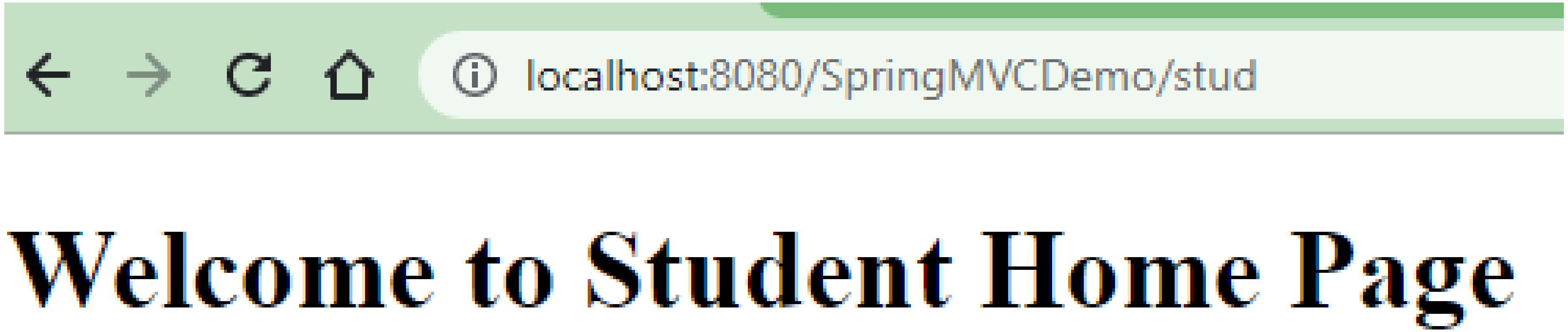
```java
package com.rit.mvc;

import org.springframework.stereoty
import org.springframework.web.binc

@Controller
@RequestMapping("/stud")
public class StudentController {

    @RequestMapping("")
    public String showHomePage() {
        return "stud_home";
    }
}
```

Stud view



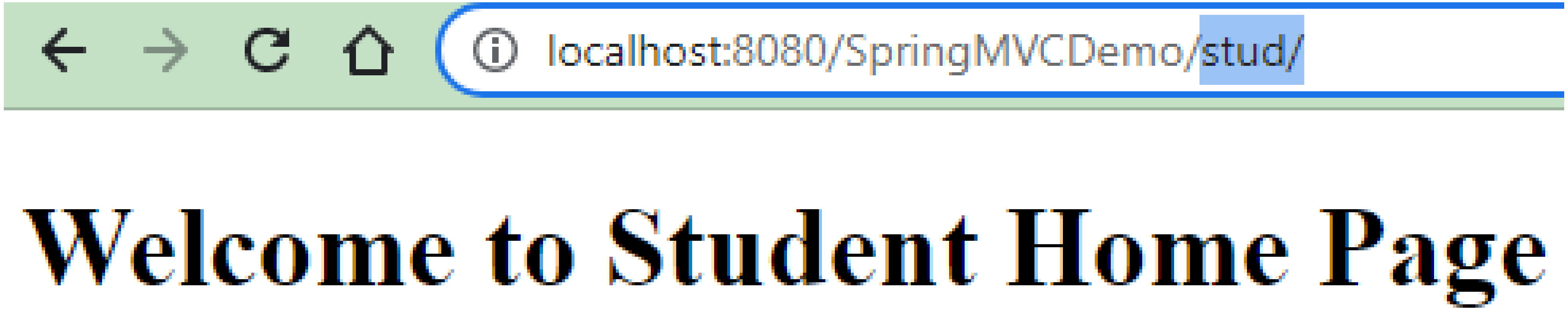
```
<%@ page language="java" contentType="text/
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <h1>Welcome to Student Home Page</h1>
</body>
</html>
```

Output



localhost:8080/SpringMVCDemo/stud

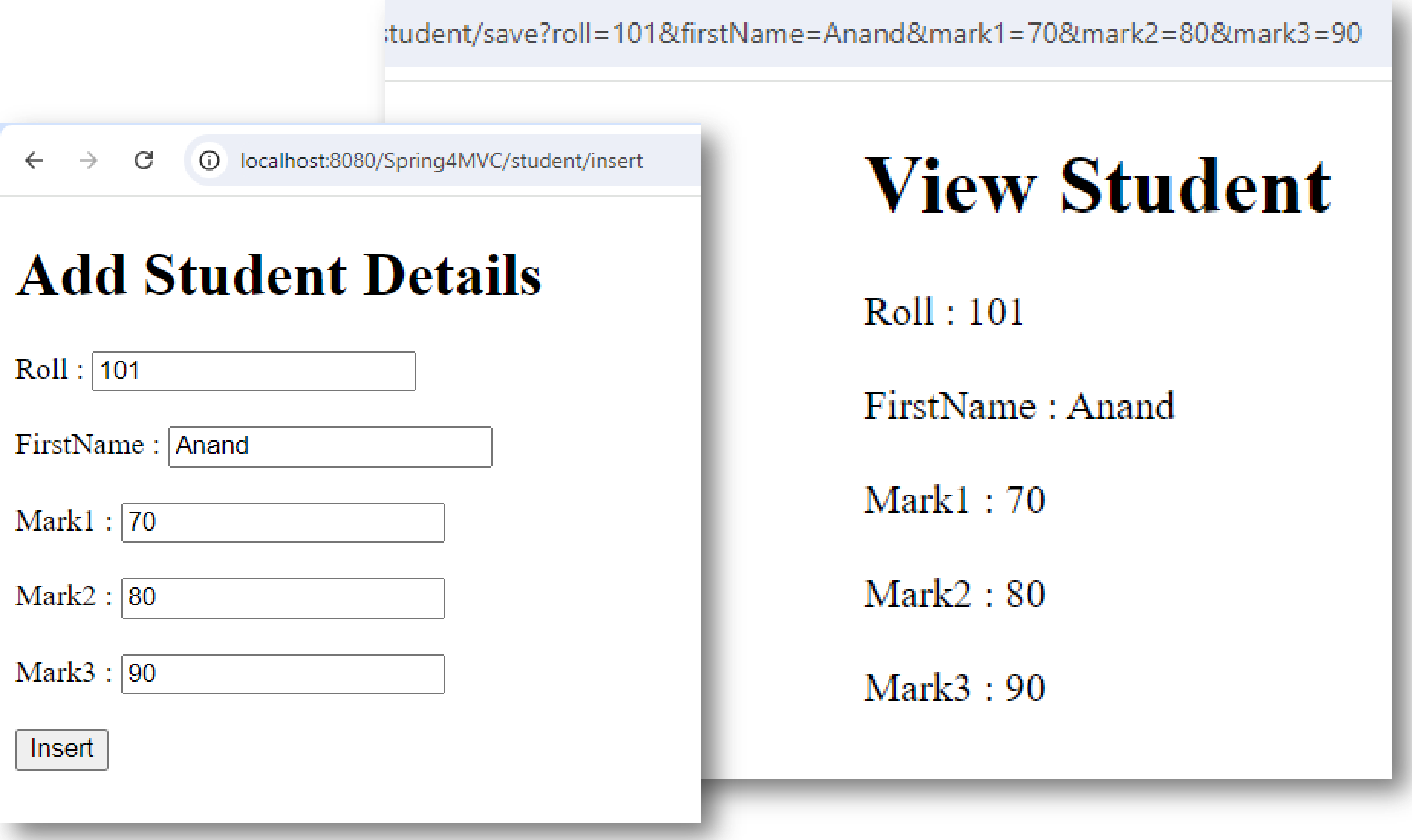
Welcome to Student Home Page



localhost:8080/SpringMVCDemo/stud/

Welcome to Student Home Page

Reading HTML Form data



localhost:8080/Spring4MVC/student/insert

Add Student Details

Roll : 101

FirstName : Anand

Mark1 : 70

Mark2 : 80

Mark3 : 90

tudent/save?roll=101&firstName=Anand&mark1=70&mark2=80&mark3=90

View Student

Roll : 101

FirstName : Anand

Mark1 : 70

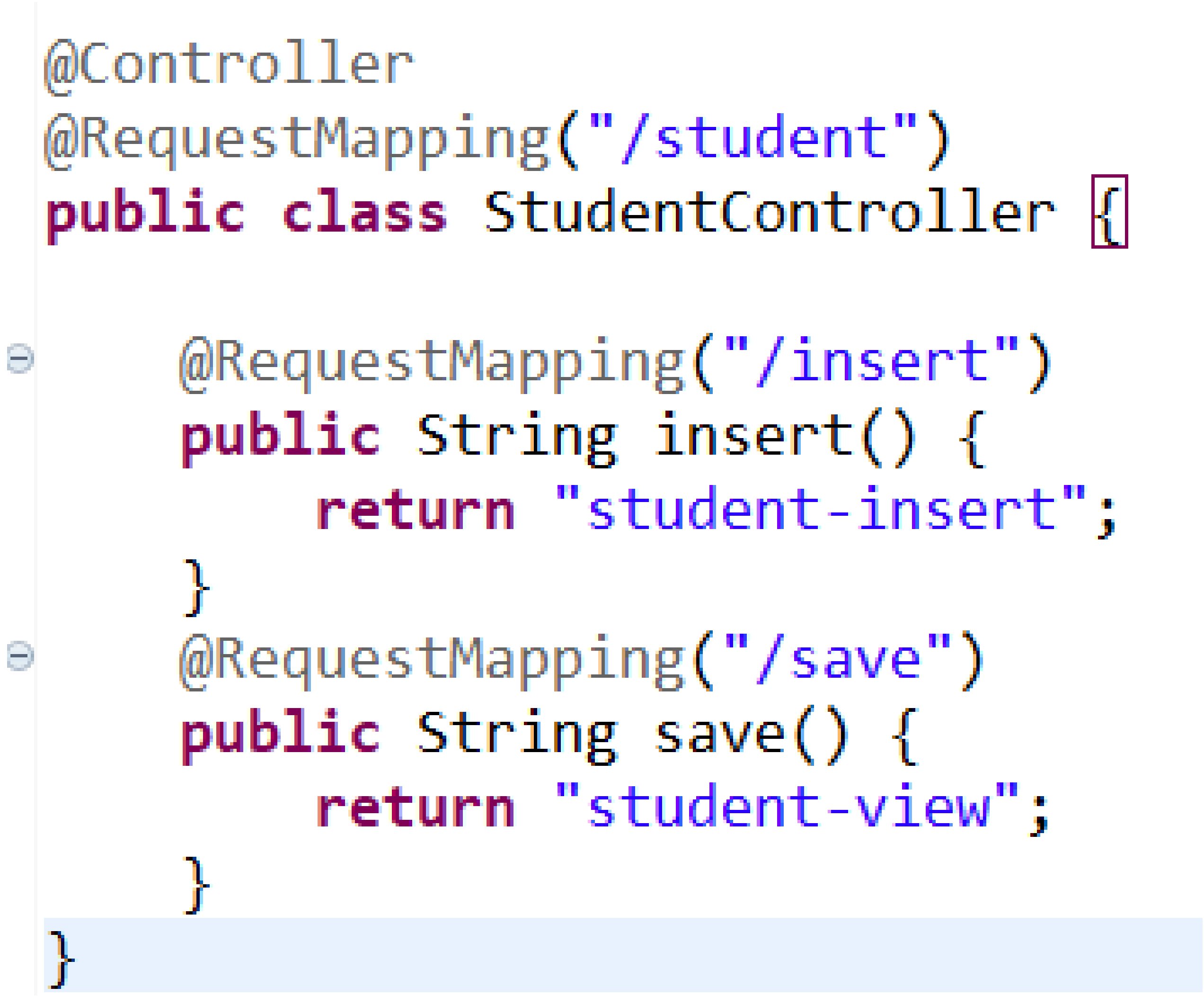
Mark2 : 80

Mark3 : 90

Development Process : First Controller

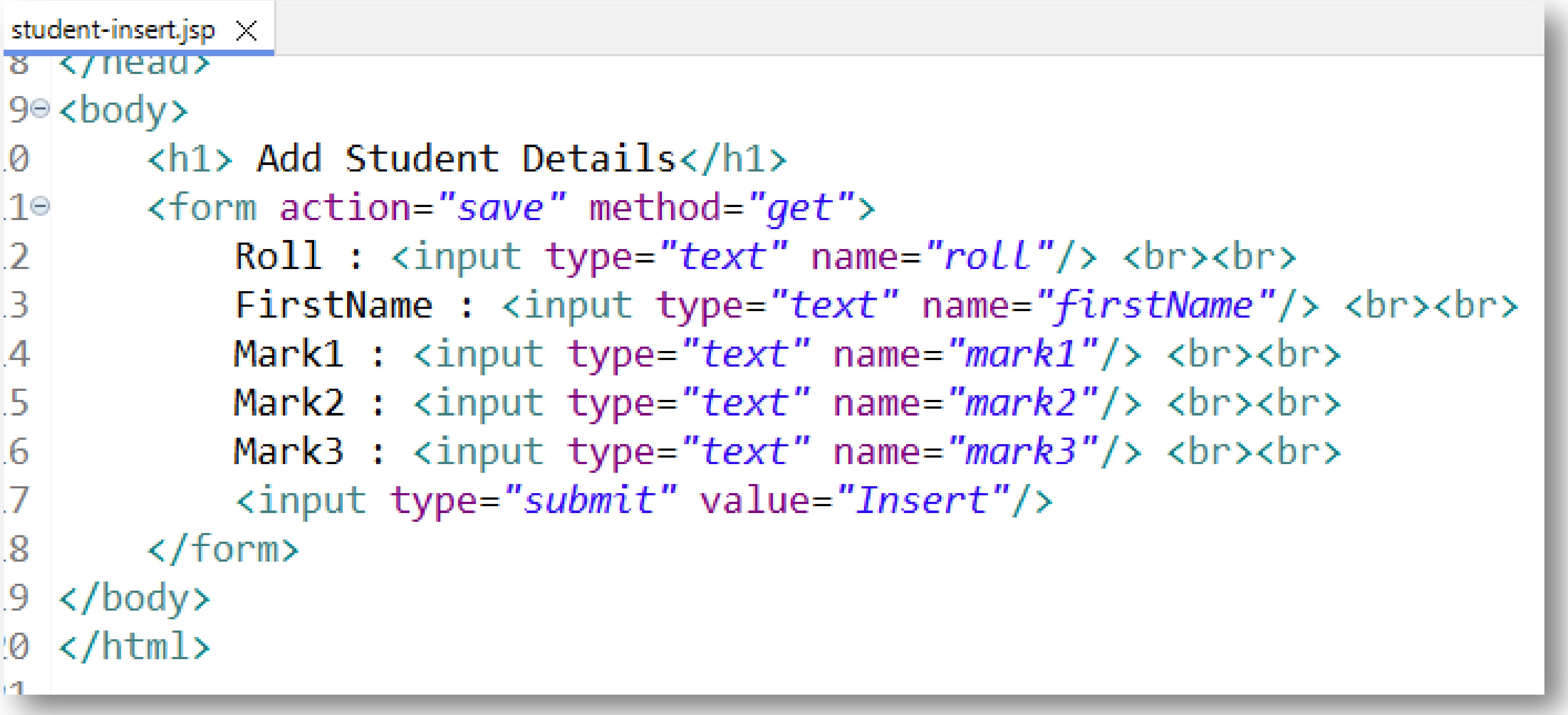
- Create a Controller Class
- Create Two Controller Methods
    - Insert()
    - Save()
- Develop View Pages
- Access the form data using ${param.<name>} in jsp

Controller



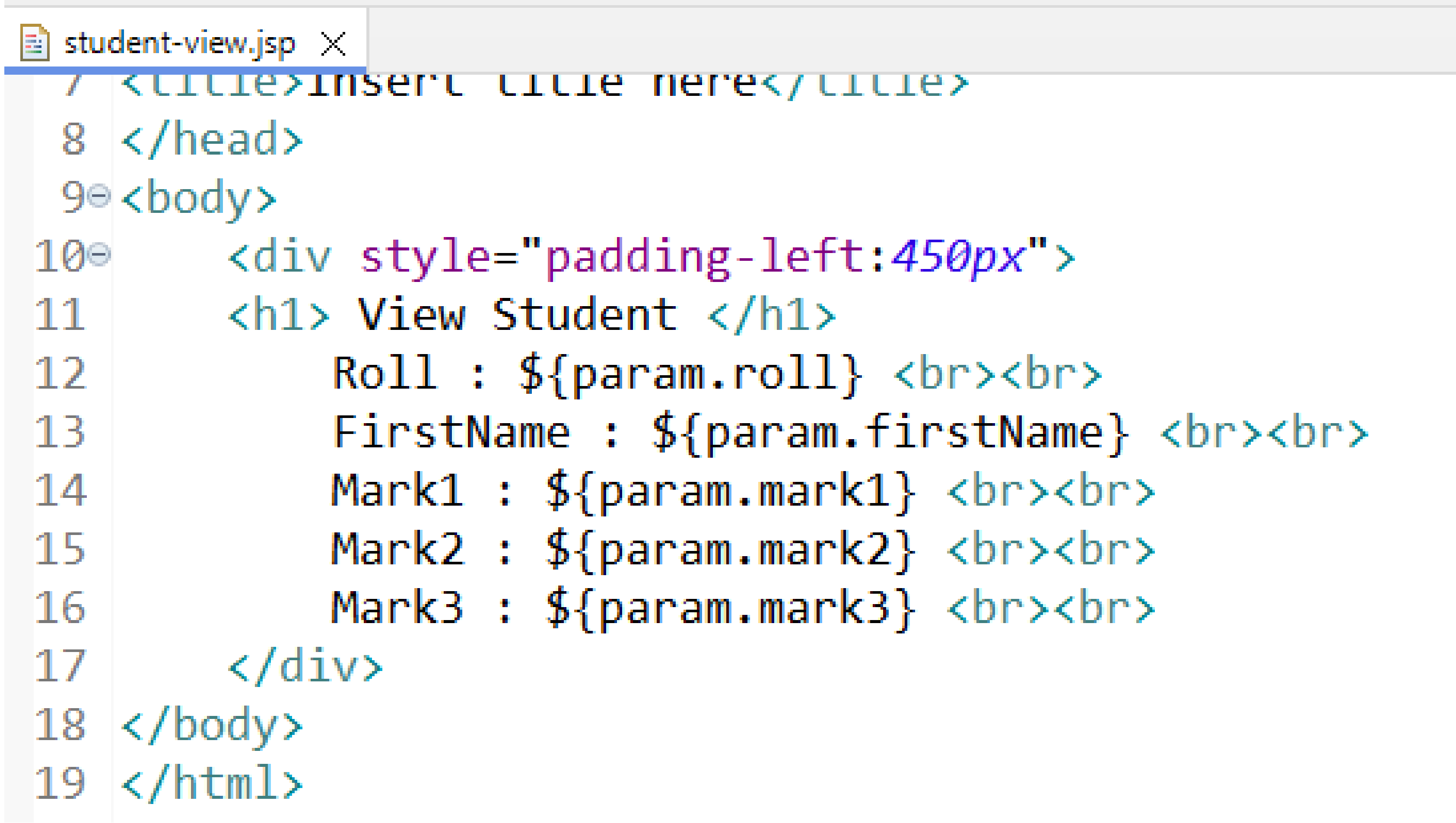
```java
@Controller
@RequestMapping("/student")
public class StudentController {

    @RequestMapping("/insert")
    public String insert() {
        return "student-insert";
    }
    @RequestMapping("/save")
    public String save() {
        return "student-view";
    }
}
```

Studuent-insert.jsp



```
</head>
<body>
    <h1> Add Student Details</h1>
    <form action="save" method="get">
        Roll : <input type="text" name="roll"/> <br><br>
        FirstName : <input type="text" name="firstName"/> <br><br>
        Mark1 : <input type="text" name="mark1"/> <br><br>
        Mark2 : <input type="text" name="mark2"/> <br><br>
        Mark3 : <input type="text" name="mark3"/> <br><br>
        <input type="submit" value="Insert"/>
    </form>
</body>
</html>
```

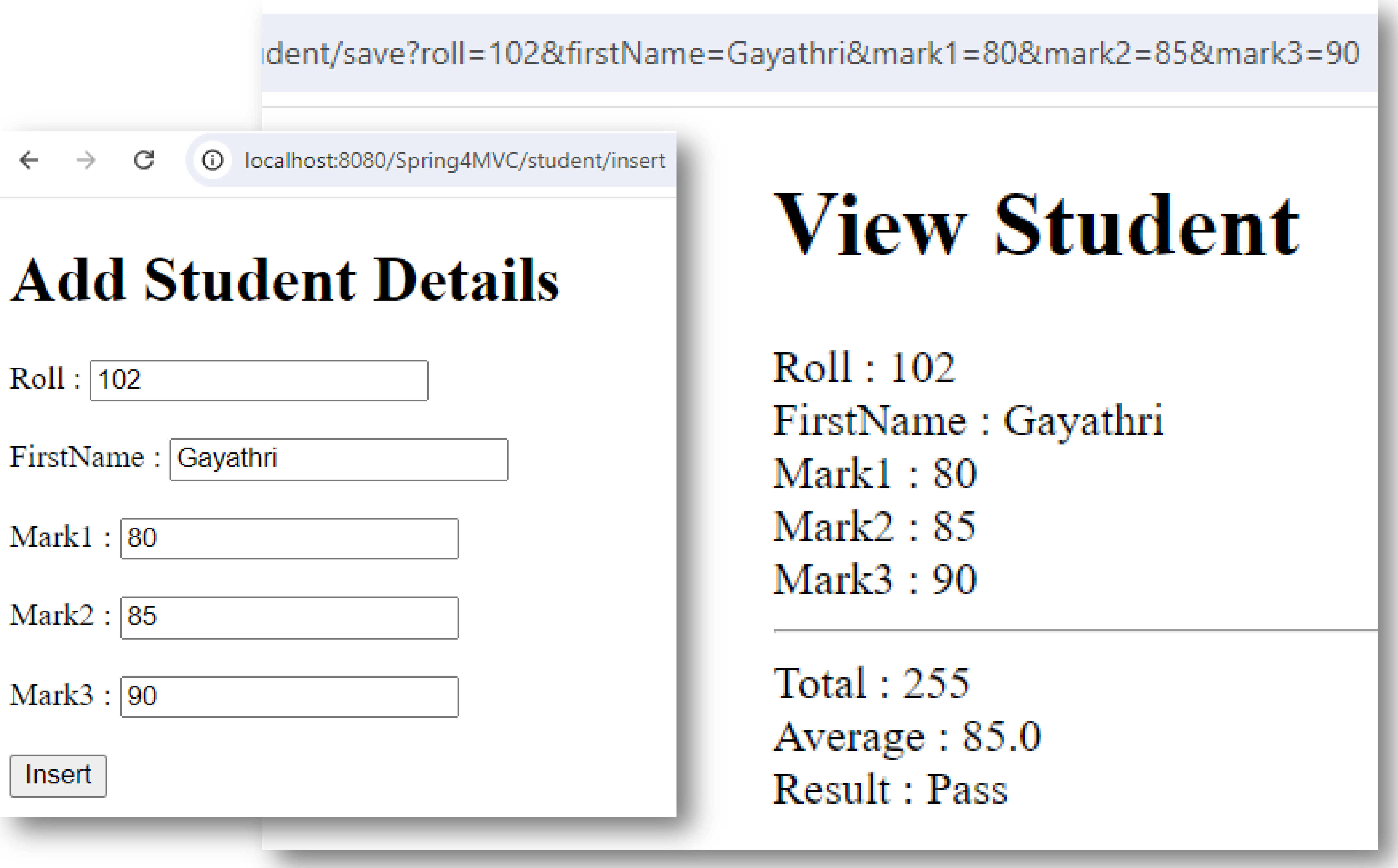
Studuent-view.jsp



```
<title>Insert title here</title>
</head>
<body>
    <div style="padding-left:450px">
    <h1> View Student </h1>
        Roll : ${param.roll} <br><br>
        FirstName : ${param.firstName} <br><br>
        Mark1 : ${param.mark1} <br><br>
        Mark2 : ${param.mark2} <br><br>
        Mark3 : ${param.mark3} <br><br>
    </div>
</body>
</html>
```

HttpRequest & Model

In Header -> Get data from VIEW FORM and set data to VIEW PAGE



dent/save?roll=102&firstName=Gayathri&mark1=80&mark2=85&mark3=90

localhost:8080/Spring4MVC/student/insert

Add Student Details

Roll : 102

FirstName : Gayathri

Mark1 : 80

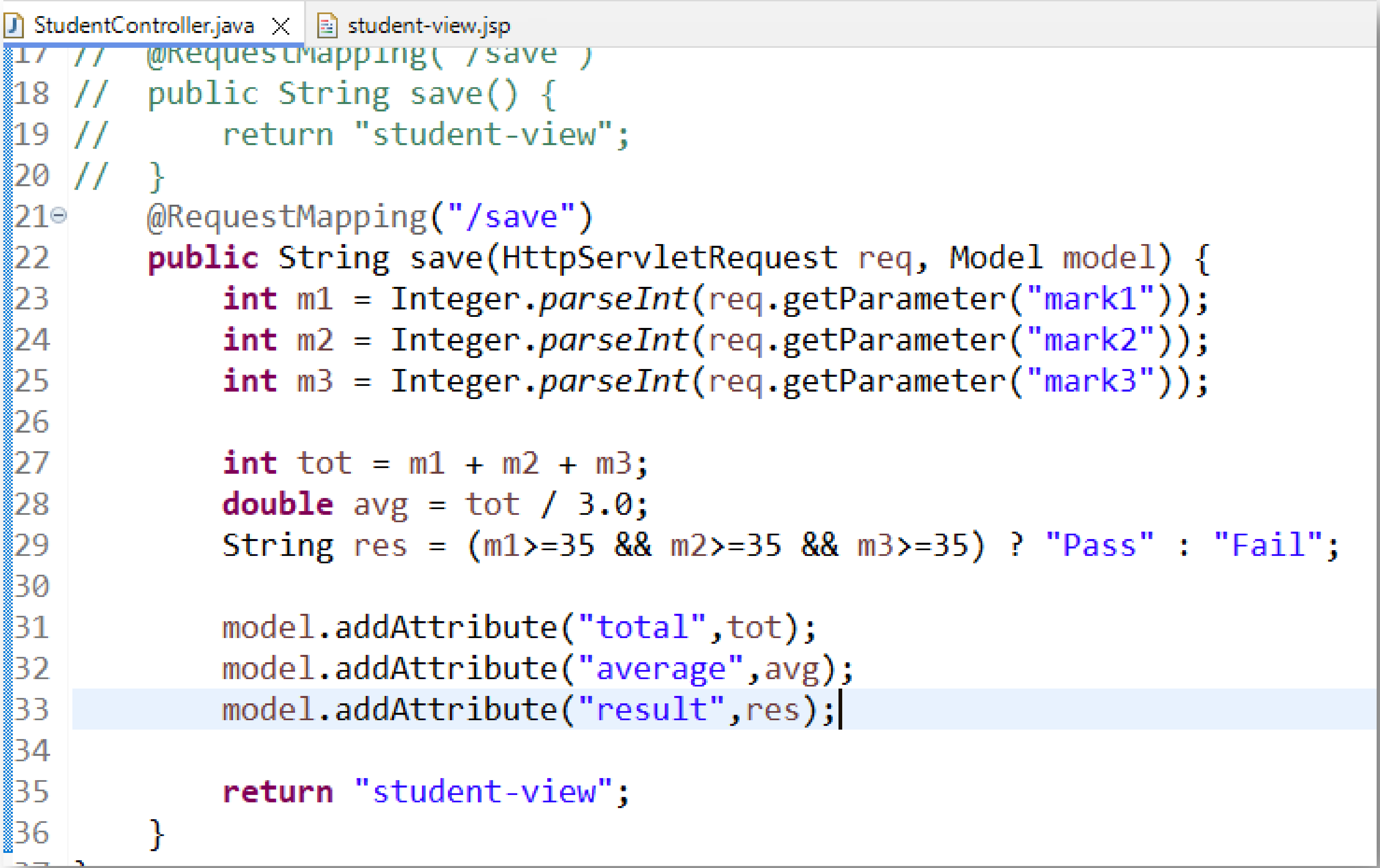
Mark2 : 85

Mark3 : 90

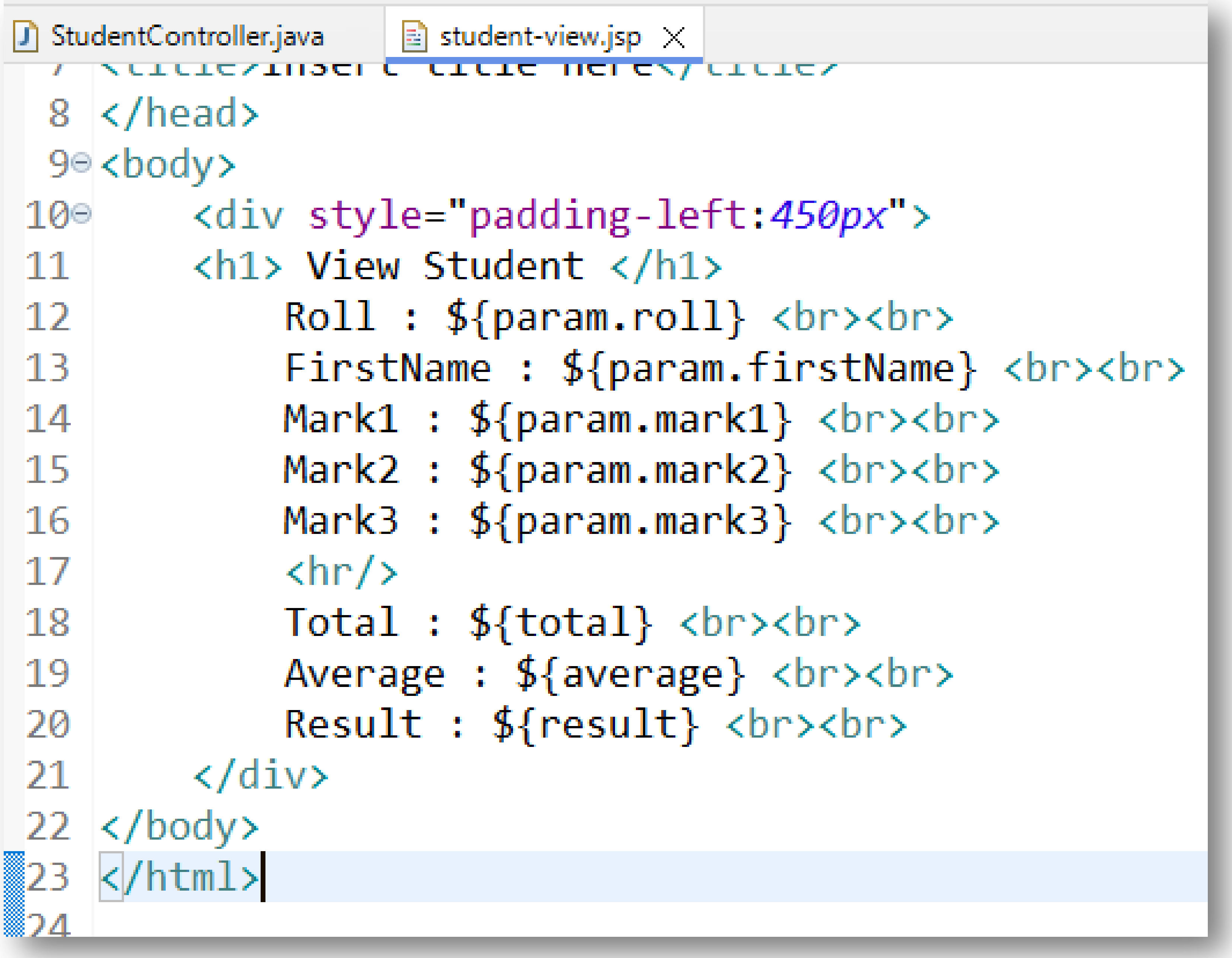
View Student

Roll : 102
FirstName : Gayathri
Mark1 : 80
Mark2 : 85
Mark3 : 90

Total : 255
Average : 85.0
Result : Pass



```java
//	@RequestMapping("/save")
//	public String save() {
//		return "student-view";
//	}
	@RequestMapping("/save")
	public String save(HttpServletRequest req, Model model) {
		int m1 = Integer.parseInt(req.getParameter("mark1"));
		int m2 = Integer.parseInt(req.getParameter("mark2"));
		int m3 = Integer.parseInt(req.getParameter("mark3"));

		int tot = m1 + m2 + m3;
		double avg = tot / 3.0;
		String res = (m1>=35 && m2>=35 && m3>=35) ? "Pass" : "Fail";

		model.addAttribute("total",tot);
		model.addAttribute("average",avg);
		model.addAttribute("result",res);

		return "student-view";
	}
}
```

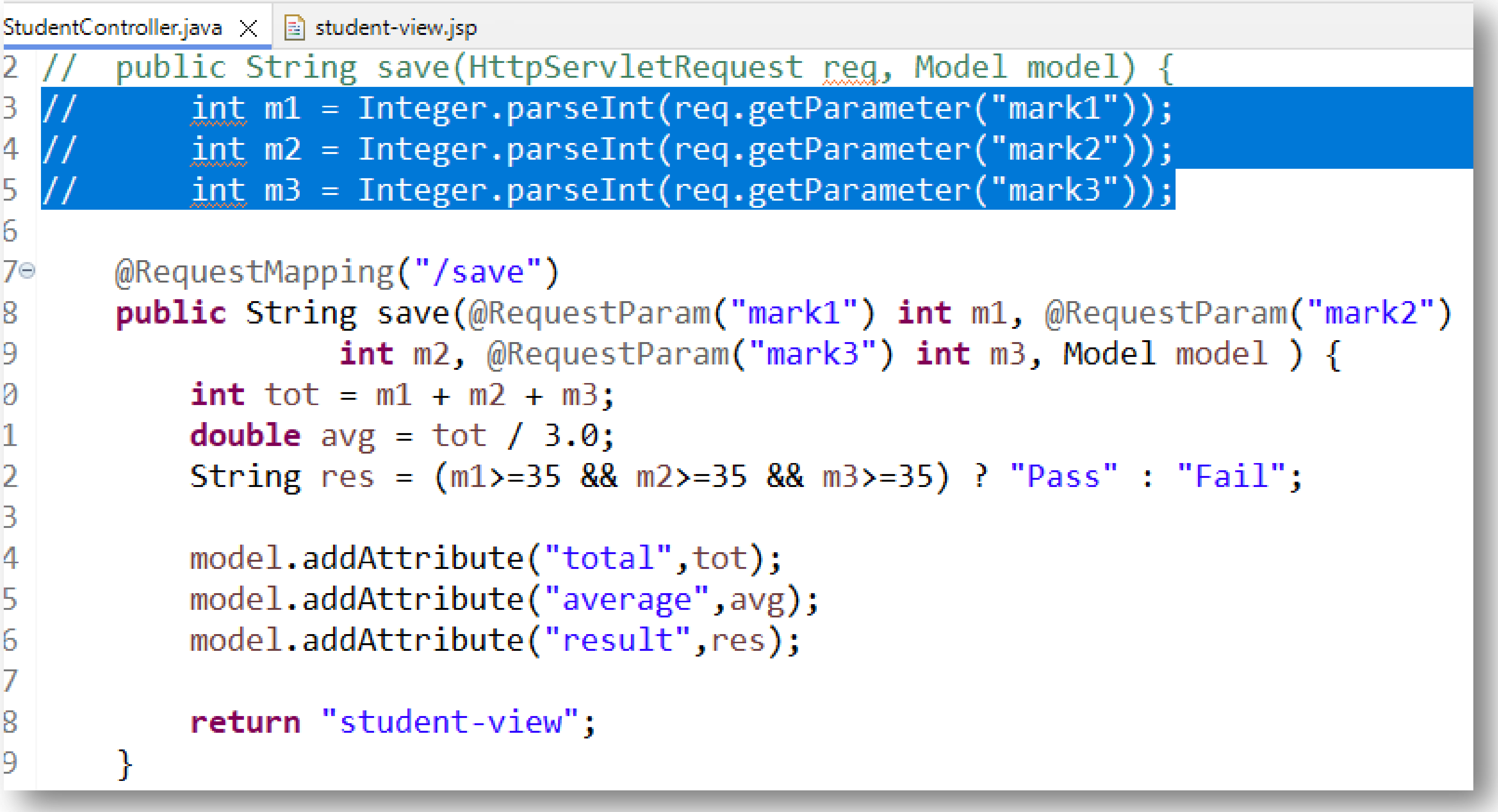


```
<title>Insert title here</title>
</head>
<body>
    <div style="padding-left:450px">
    <h1> View Student </h1>
        Roll : ${param.roll} <br><br>
        FirstName : ${param.firstName} <br><br>
        Mark1 : ${param.mark1} <br><br>
        Mark2 : ${param.mark2} <br><br>
        Mark3 : ${param.mark3} <br><br>
        <hr/>
        Total : ${total} <br><br>
        Average : ${average} <br><br>
        Result : ${result} <br><br>
    </div>
</body>
</html>
```

RequestParam

Instead of HttpServletRequest

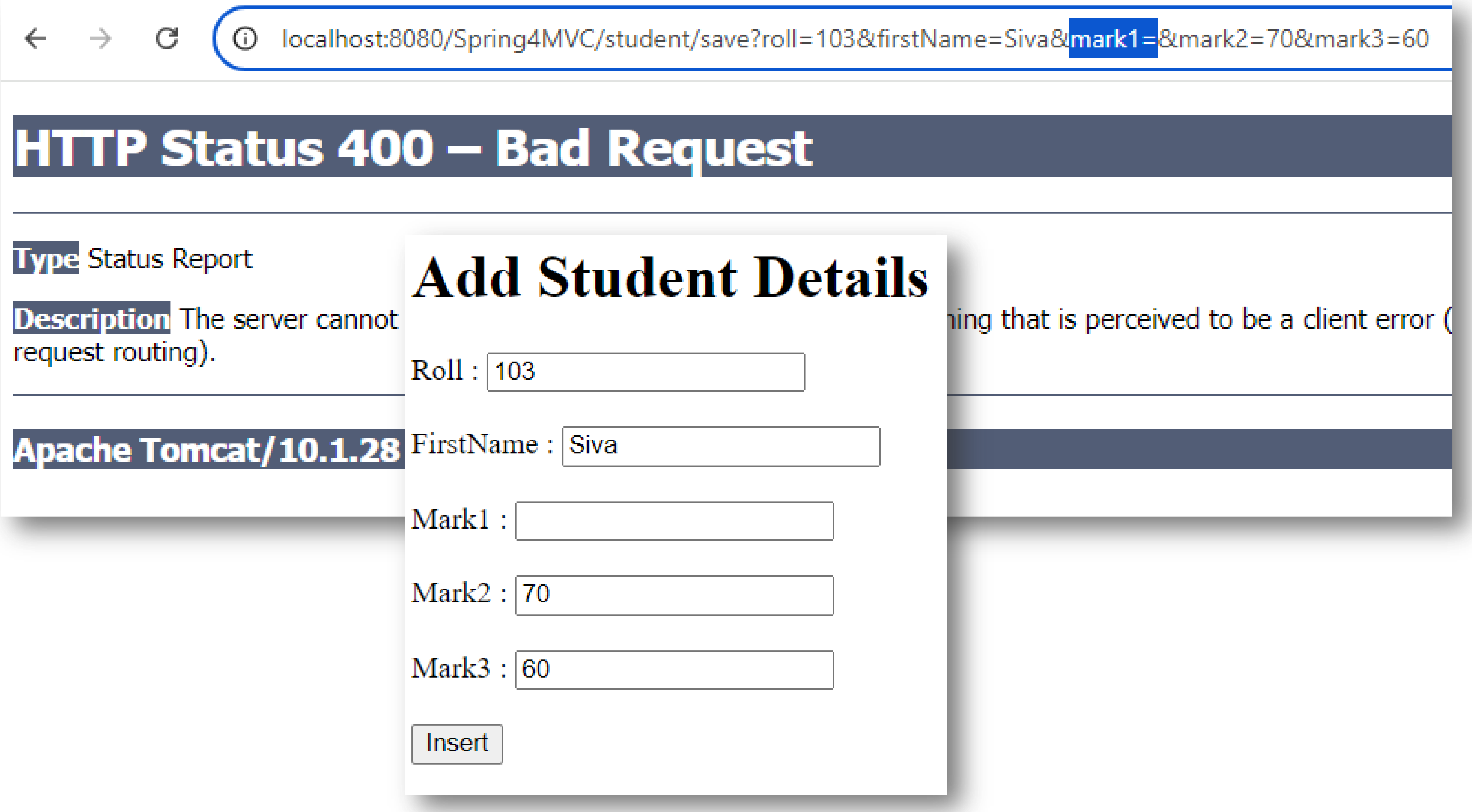
@RequestParam instead of HttpServletRequest



```
//   public String save(HttpServletRequest req, Model model) {
//       int m1 = Integer.parseInt(req.getParameter("mark1"));
//       int m2 = Integer.parseInt(req.getParameter("mark2"));
//       int m3 = Integer.parseInt(req.getParameter("mark3"));

    @RequestMapping("/save")
    public String save(@RequestParam("mark1") int m1, @RequestParam("mark2")
                int m2, @RequestParam("mark3") int m3, Model model ) {
        int tot = m1 + m2 + m3;
        double avg = tot / 3.0;
        String res = (m1>=35 && m2>=35 && m3>=35) ? "Pass" : "Fail";

        model.addAttribute("total",tot);
        model.addAttribute("average",avg);
        model.addAttribute("result",res);

        return "student-view";
    }
```

@RequestParam

- When we use @RequestParam we must pass the data for the param
- By default the required attribute is “true”
- We can set it false to make the param optional
- We can also set default value for the param



localhost:8080/Spring4MVC/student/save?roll=103&firstName=Siva&mark1=&mark2=70&mark3=60

HTTP Status 400 – Bad Request

Type Status Report

Description The server cannot ing that is perceived to be a client error (request routing).

Apache Tomcat/10.1.28

Add Student Details

Roll : 103

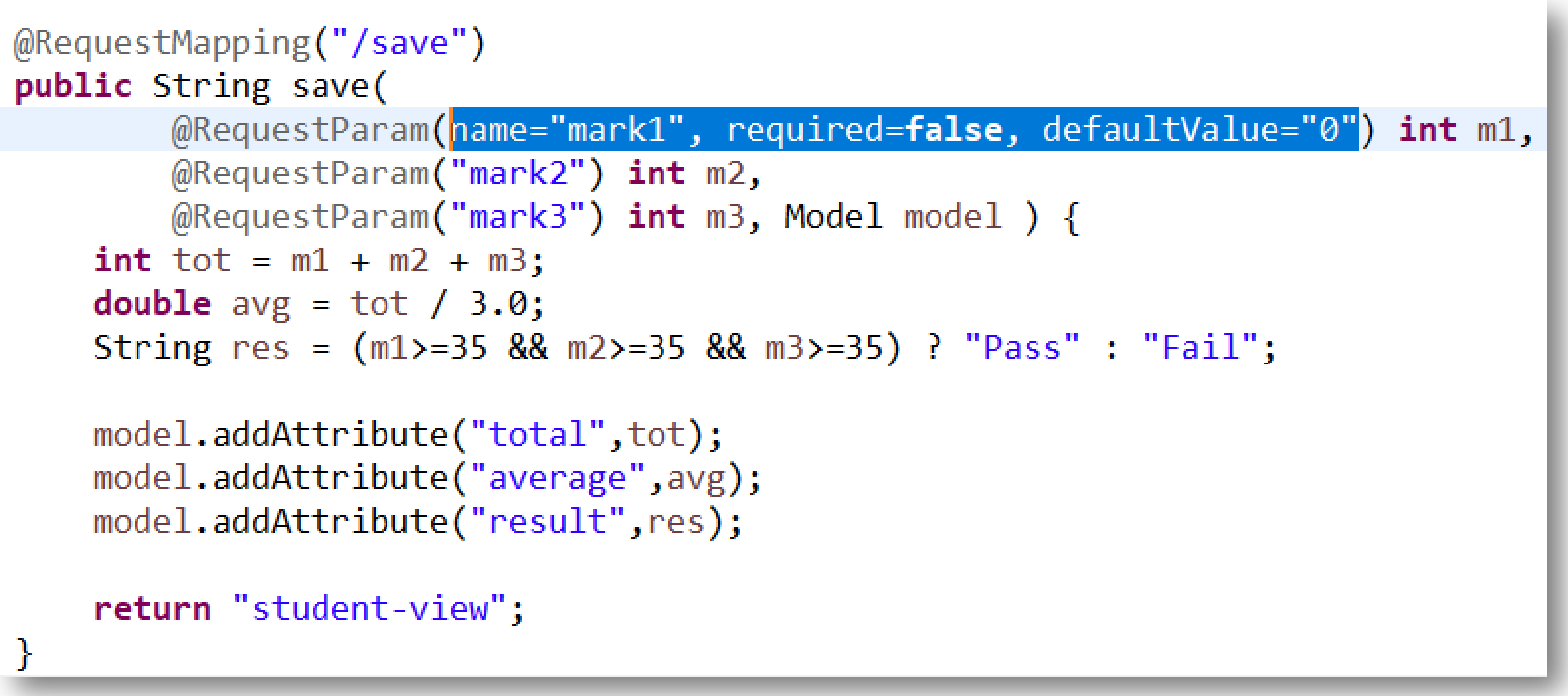
FirstName : Siva

Mark1 :

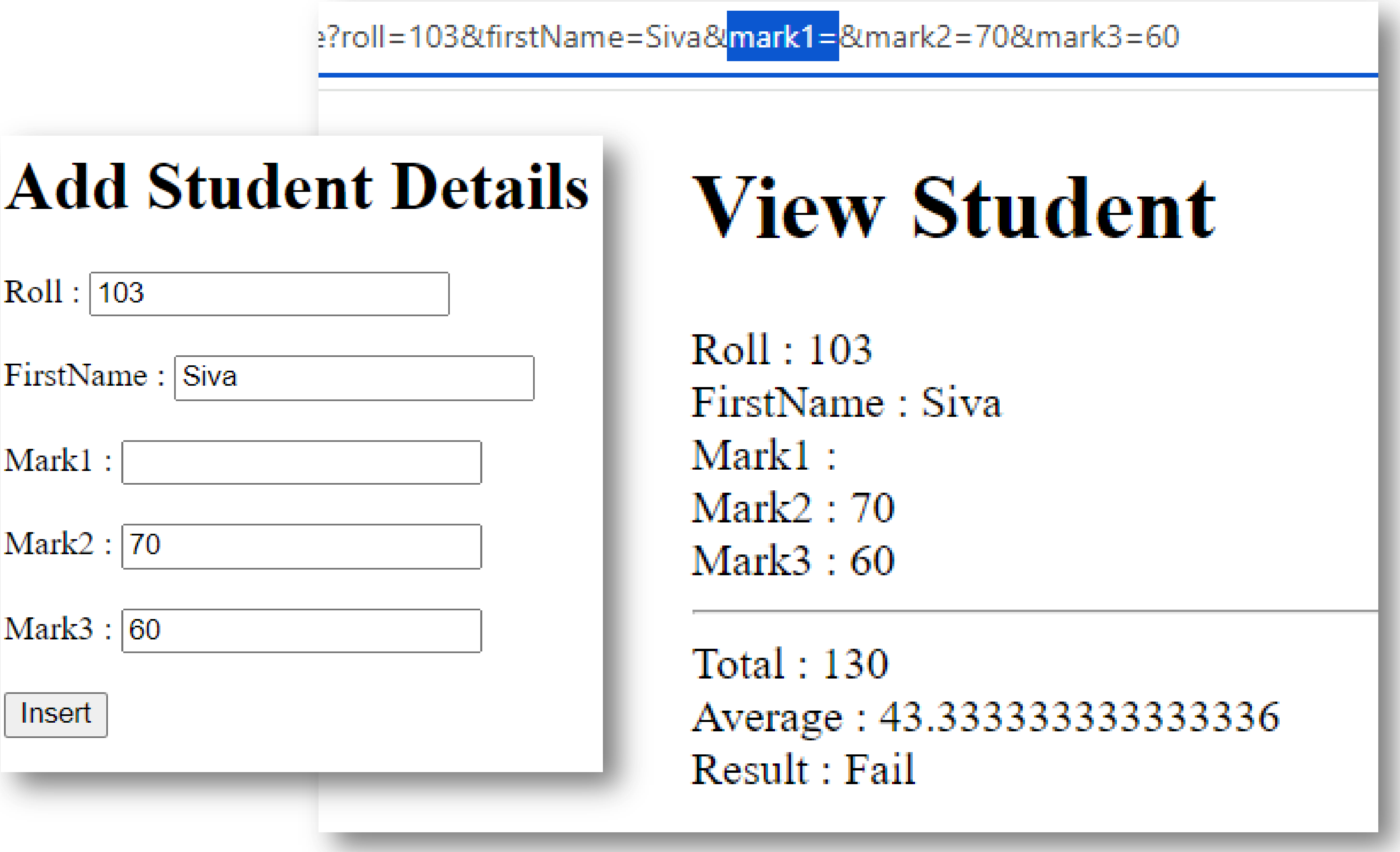
Mark2 : 70

Mark3 : 60

required & defaultValue attributes



```java
@RequestMapping("/save")
public String save(
        @RequestParam(name="mark1", required=false, defaultValue="0") int m1,
        @RequestParam("mark2") int m2,
        @RequestParam("mark3") int m3, Model model ) {
    int tot = m1 + m2 + m3;
    double avg = tot / 3.0;
    String res = (m1>=35 && m2>=35 && m3>=35) ? "Pass" : "Fail";

    model.addAttribute("total",tot);
    model.addAttribute("average",avg);
    model.addAttribute("result",res);

    return "student-view";
}
```



?roll=103&firstName=Siva&mark1=&mark2=70&mark3=60

Add Student Details

Roll : 103

FirstName : Siva

Mark1 :

Mark2 : 70

Mark3 : 60

View Student

Roll : 103
FirstName : Siva
Mark1 :
Mark2 : 70
Mark3 : 60

Total : 130
Average : 43.333333333333336
Result : Fail

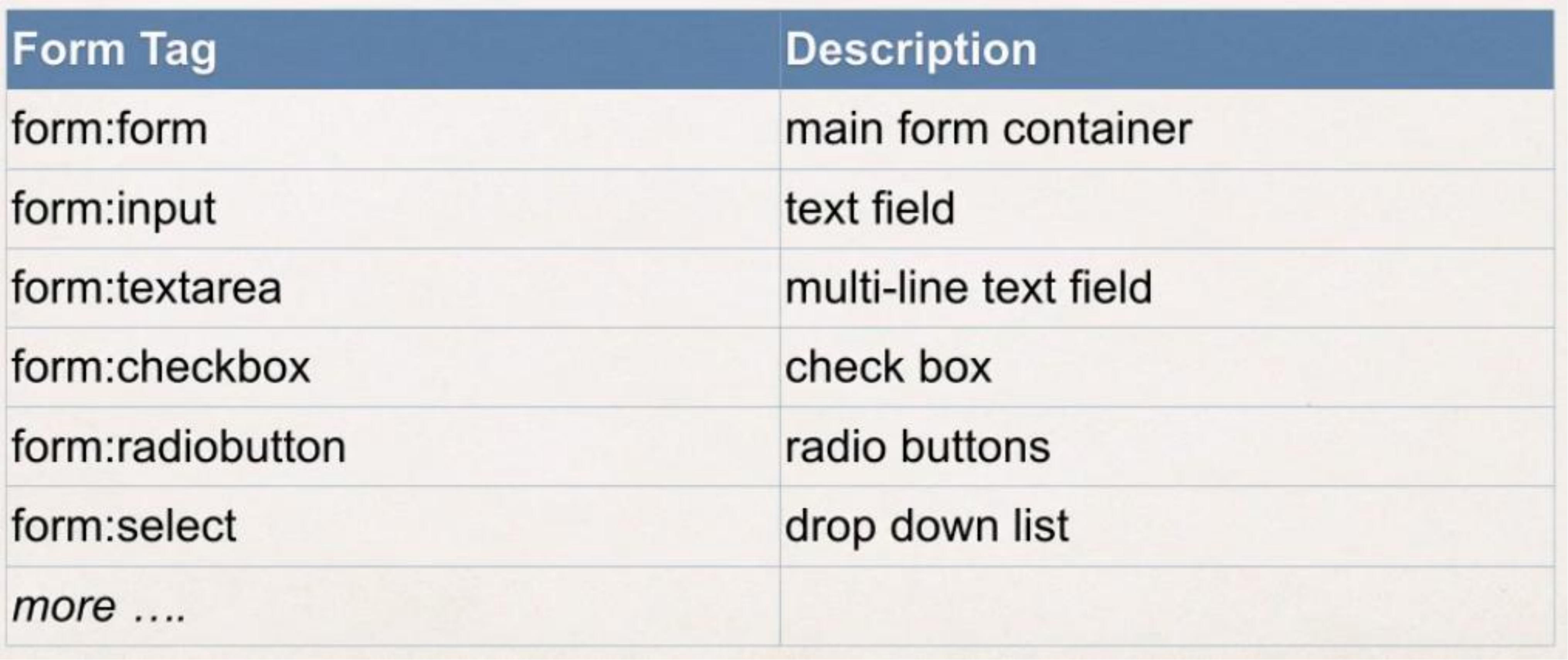
Spring MVC Form Tags

Spring MVC Form Tags

- Spring MVC Form Tags are the building blocks for a web page
- Form Tags are configurable and reusable for a web page
- Spring MVC Form Tags can make use of databinding
- Automatically Setting/Retrieving data from a java object/bean

Form Tags

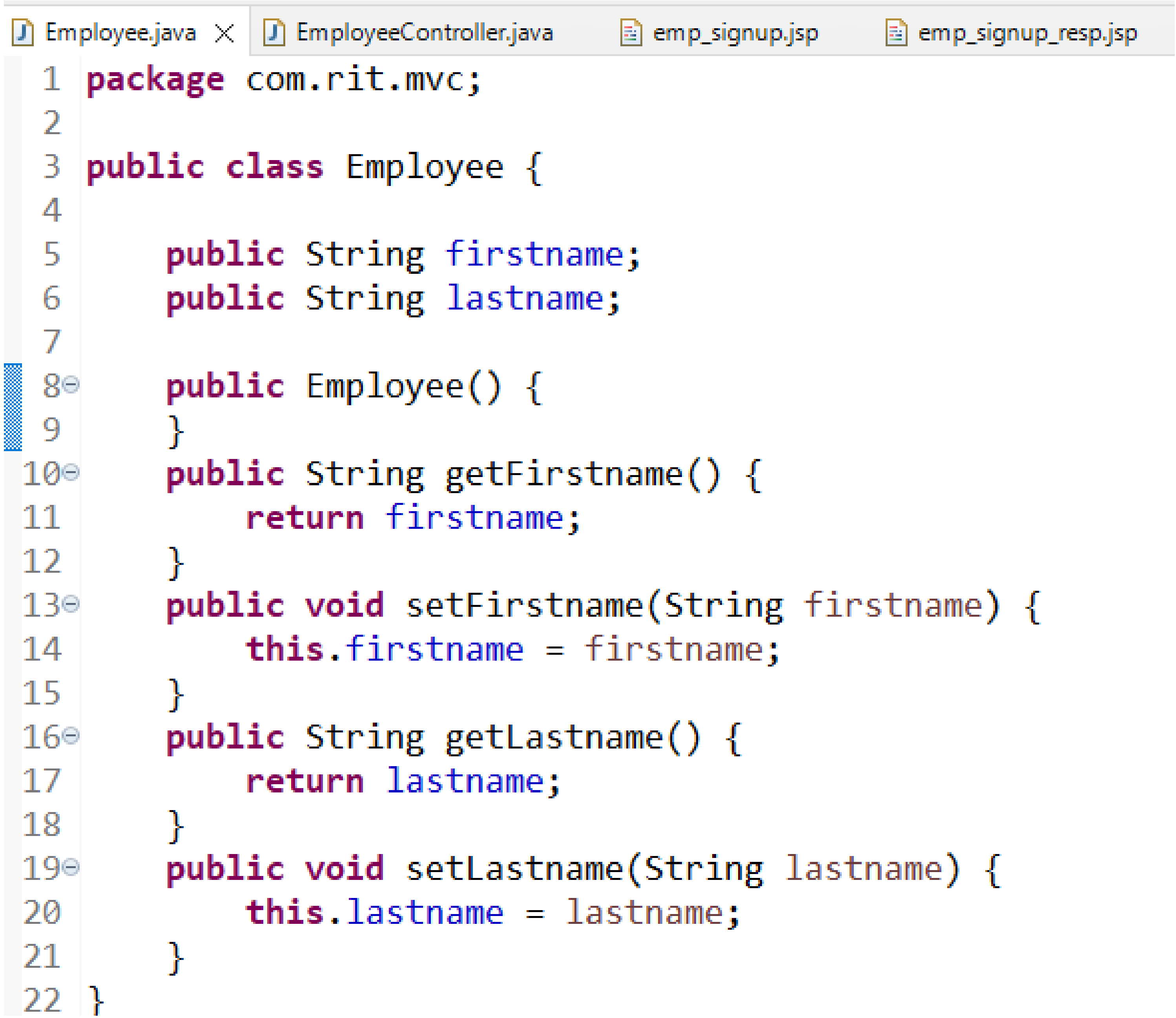
- Form Tags will generate HTML for us.



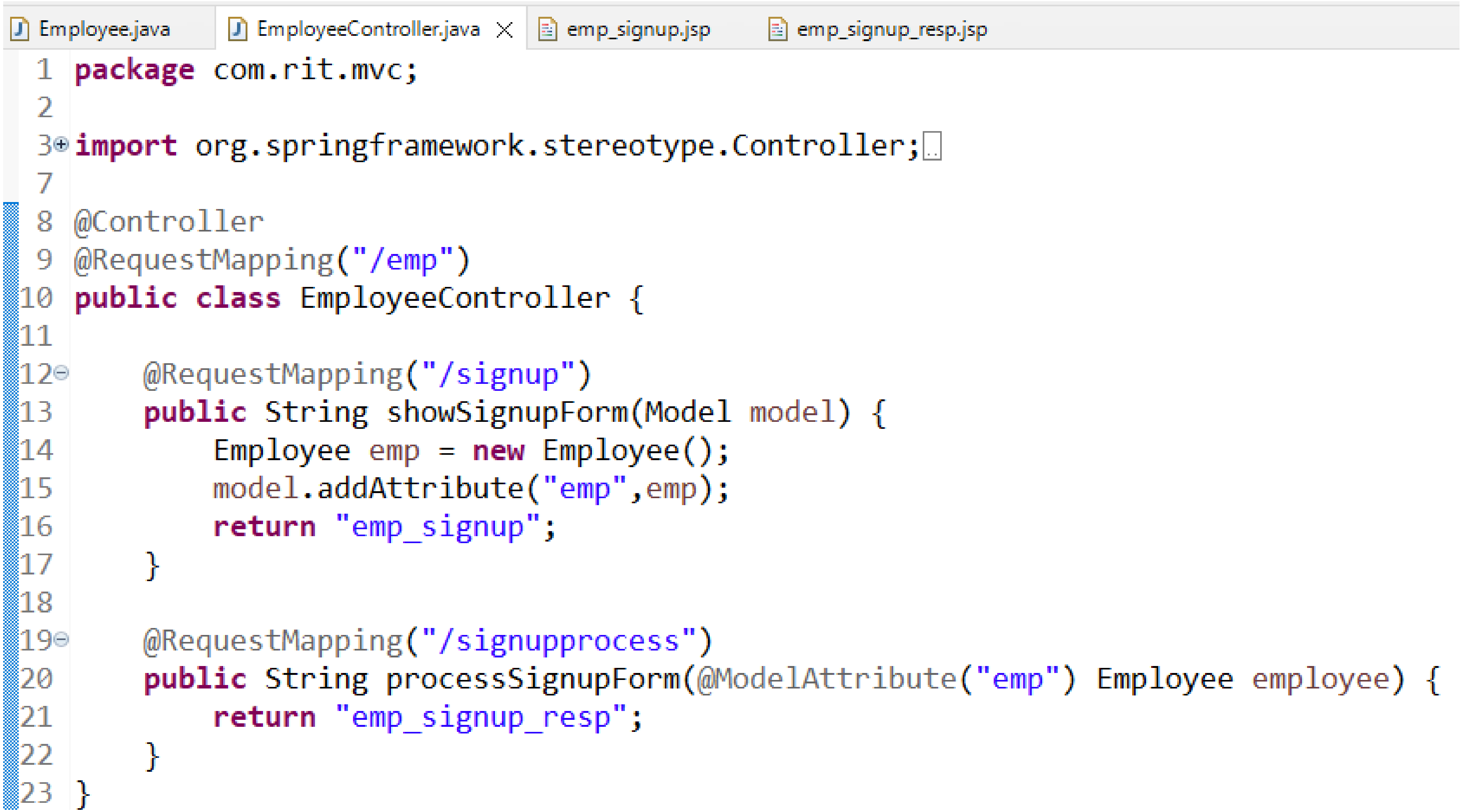
Form Tag	Description
form:form	main form container
form:input	text field
form:textarea	multi-line text field
form:checkbox	check box
form:radiobutton	radio buttons
form:select	drop down list
more ….	

Development Process

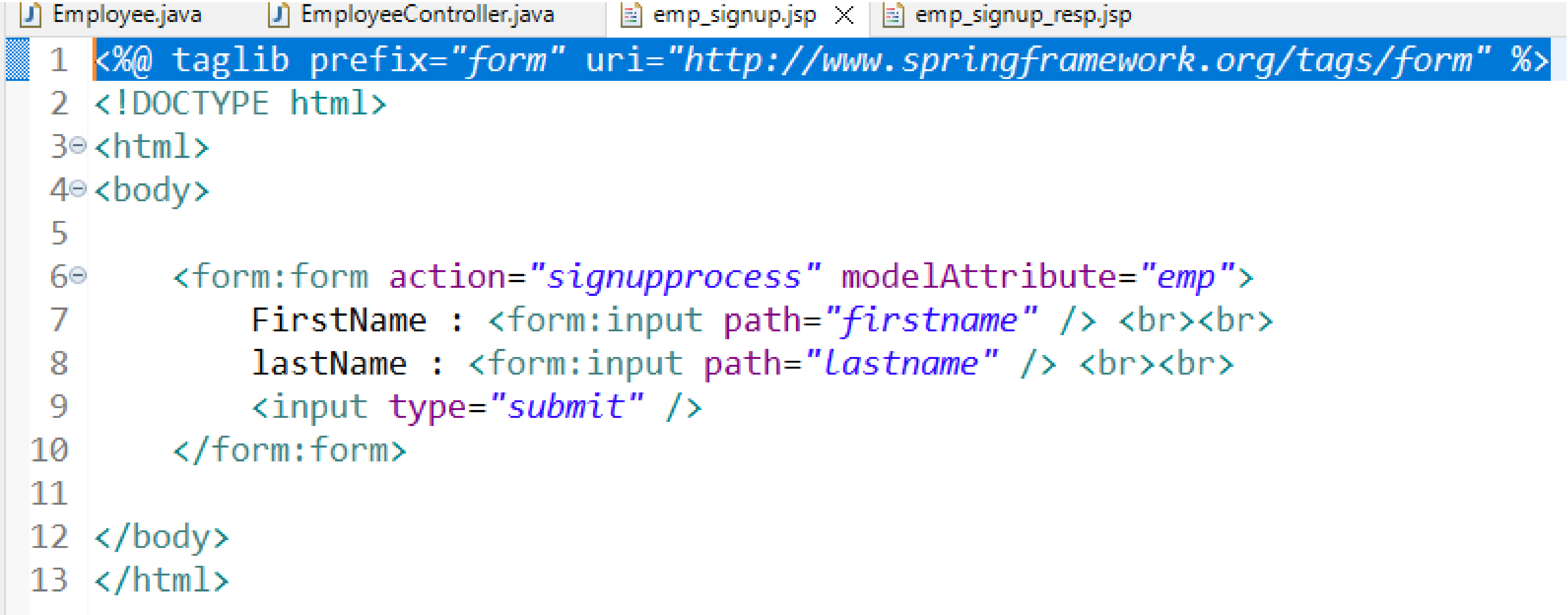
- Create a bean class (Employee)
- Create a Controller (EmpController)
  - Create a form method with model attribute
  - Create a process method with ModelAttribute
- Create a Form Jsp with formtags and path
- Create a Process Jsp



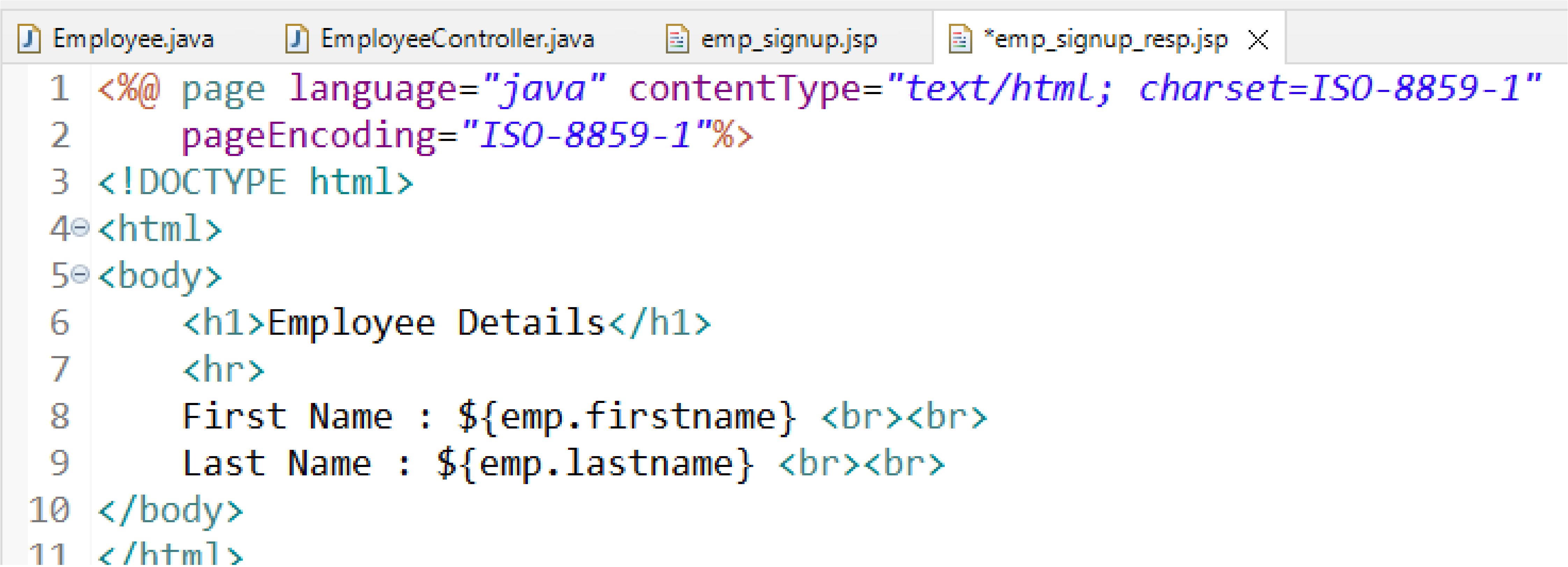
```java
package com.rit.mvc;

public class Employee {

    public String firstname;
    public String lastname;

    public Employee() {
    }
    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;
    }
}
```



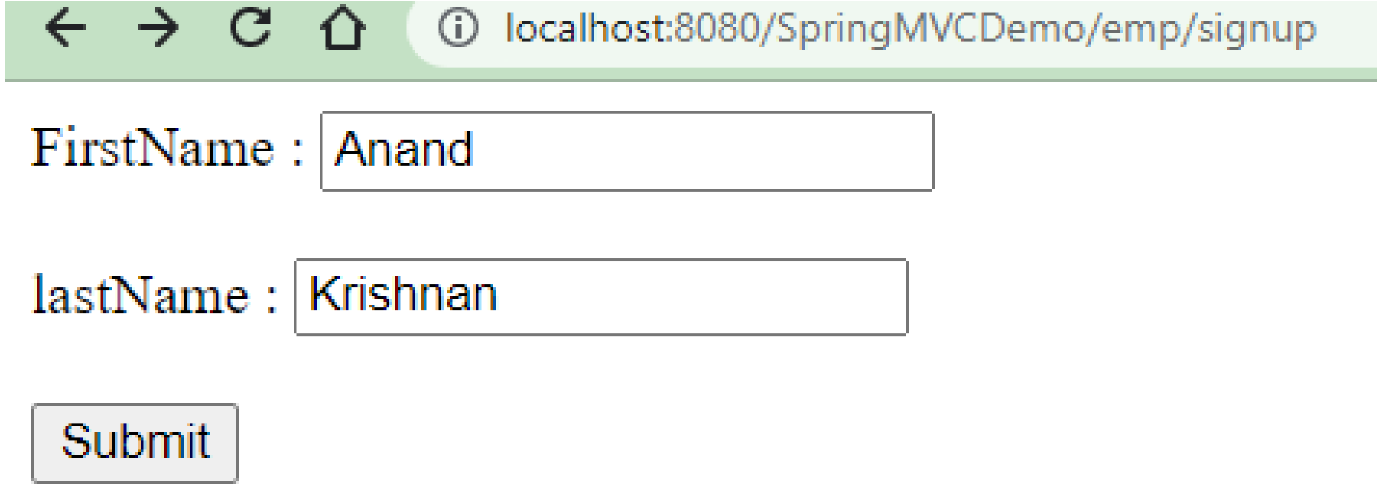
```java
package com.rit.mvc;

import org.springframework.stereotype.Controller;

@Controller
@RequestMapping("/emp")
public class EmployeeController {

    @RequestMapping("/signup")
    public String showSignupForm(Model model) {
        Employee emp = new Employee();
        model.addAttribute("emp",emp);
        return "emp_signup";
    }

    @RequestMapping("/signupprocess")
    public String processSignupForm(@ModelAttribute("emp") Employee employee) {
        return "emp_signup_resp";
    }
}
```



```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<!DOCTYPE html>
<html>
<body>

    <form:form action="signupprocess" modelAttribute="emp">
        FirstName : <form:input path="firstname" /> <br><br>
        lastName : <form:input path="lastname" /> <br><br>
        <input type="submit" />
    </form:form>

</body>
</html>
```



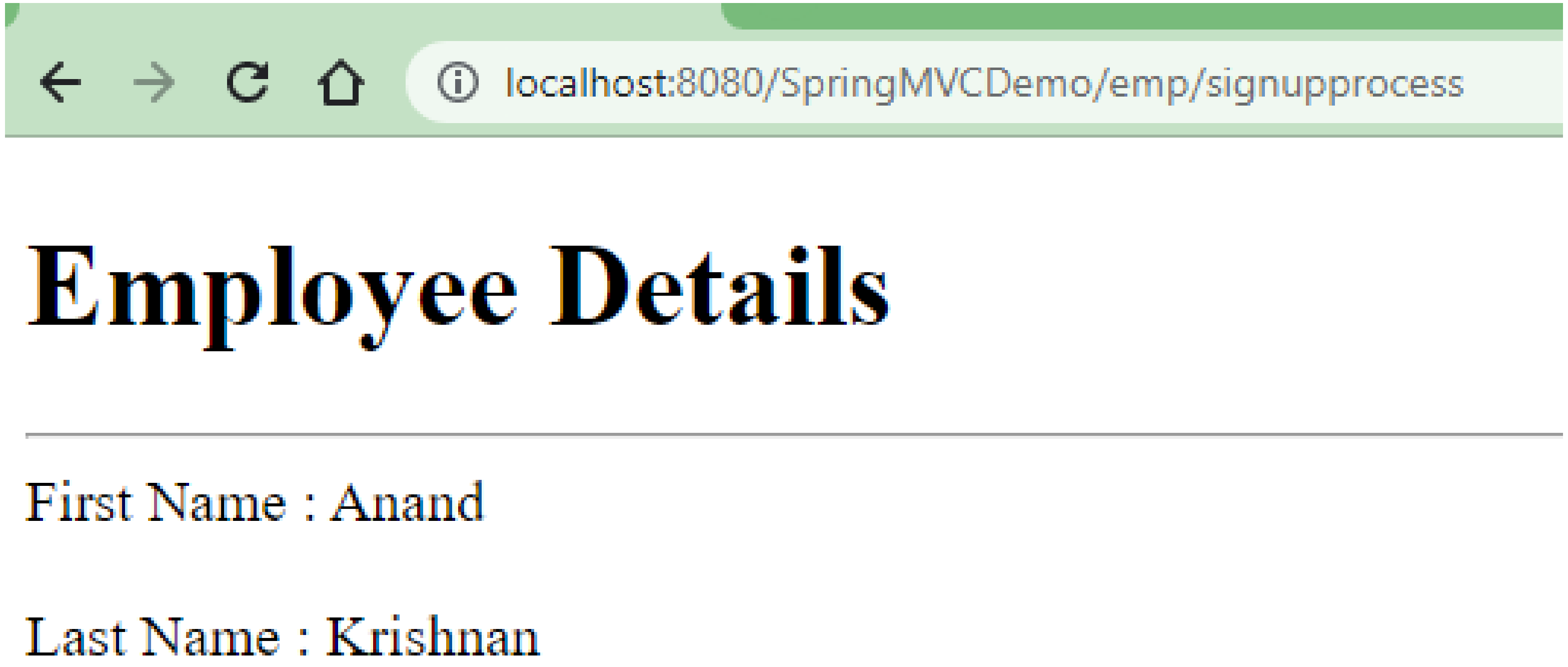
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<body>
    <h1>Employee Details</h1>
    <hr>
    First Name : ${emp.firstname} <br><br>
    Last Name : ${emp.lastname} <br><br>
</body>
</html>
```



localhost:8080/SpringMVCDemo/emp/signup

FirstName : Anand

lastName : Krishnan



localhost:8080/SpringMVCDemo/emp/signupprocess

Employee Details

First Name : Anand

Last Name : Krishnan

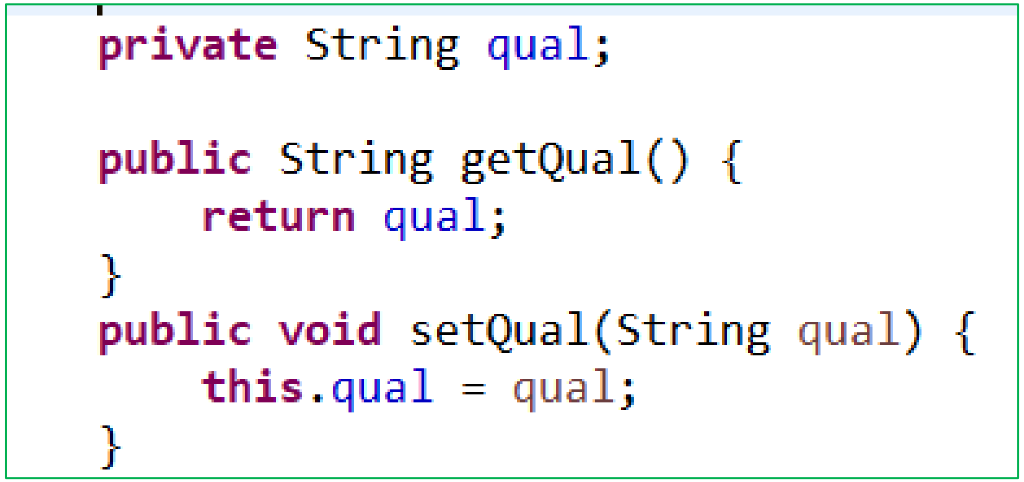
MVC Form Tag - Select

Development Process

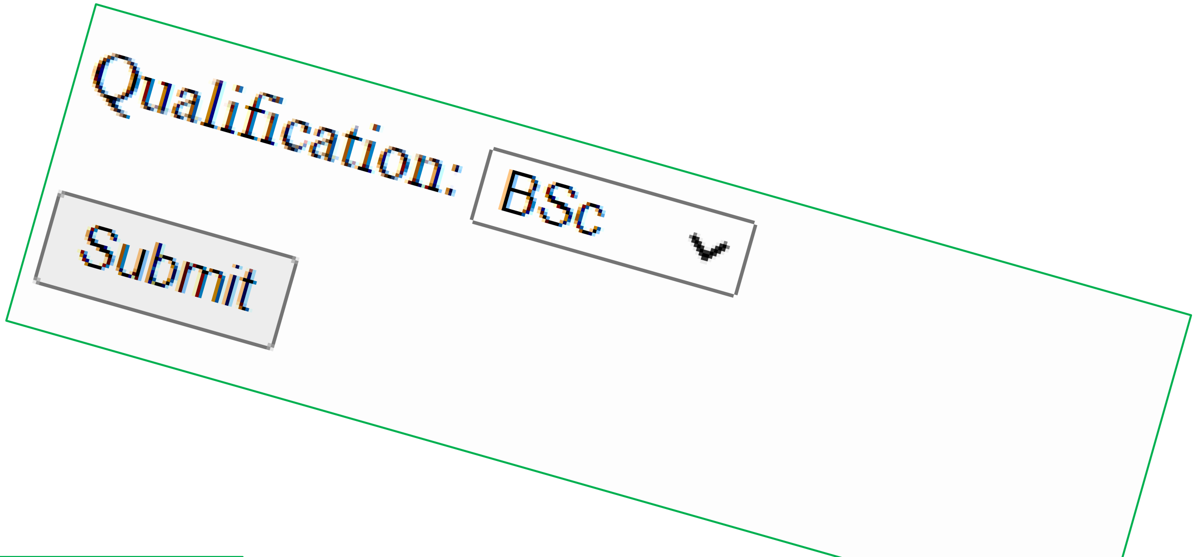
- Add a property in Employee (bean) class
- Generate Getter and Setter
- Add the dropdown element

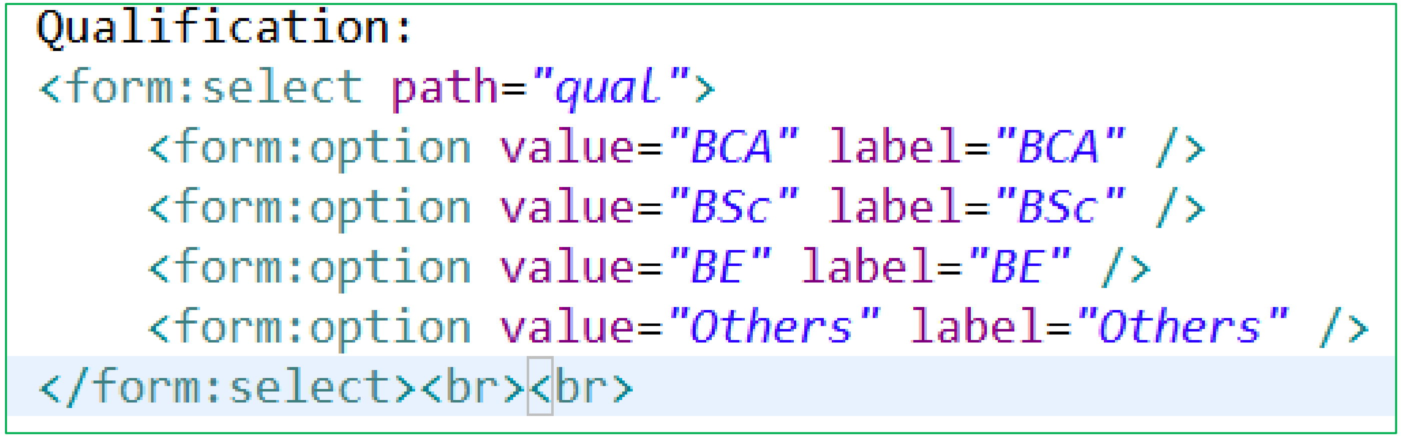
  <form:select..> tag in emp_signup.jsp
- Get the value in the response page

  ${emp.--------}

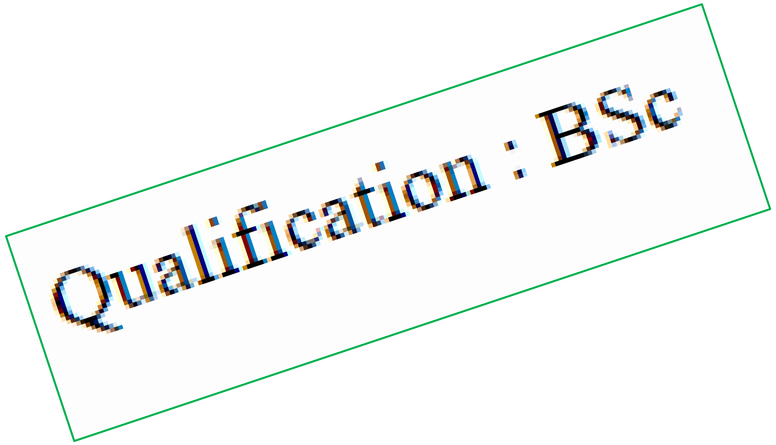


```
    private String qual;

    public String getQual() {
        return qual;
    }
    public void setQual(String qual) {
        this.qual = qual;
    }
```

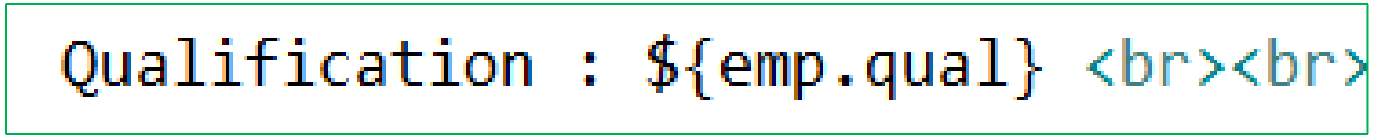




```
Qualification:
<form:select path="qual">
    <form:option value="BCA" label="BCA" />
    <form:option value="BSc" label="BSc" />
    <form:option value="BE" label="BE" />
    <form:option value="Others" label="Others" />
</form:select><br><br>
```

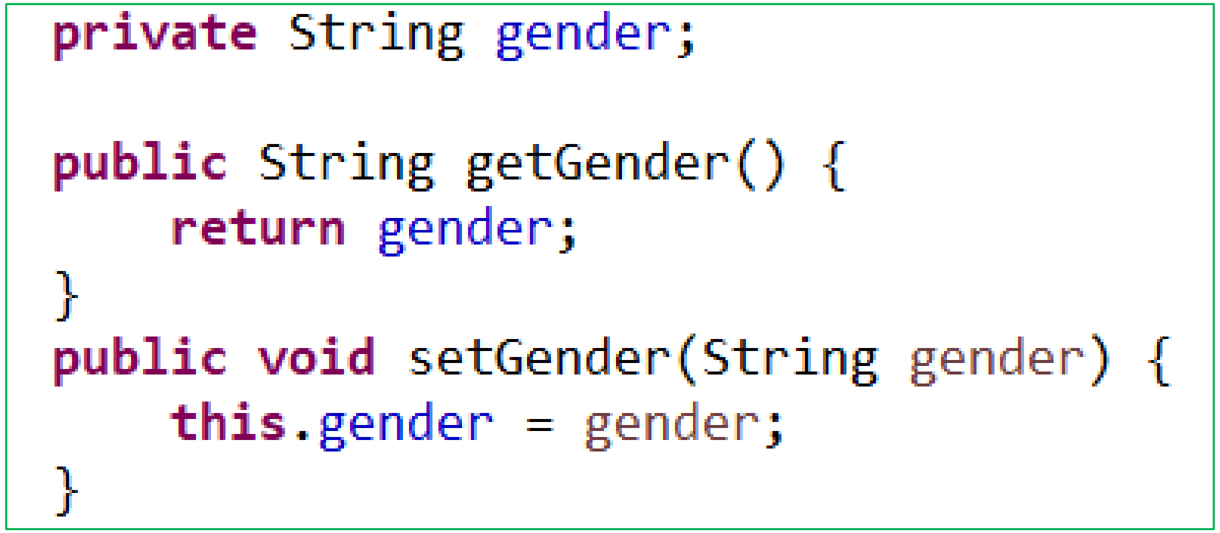


Qualification : BSc

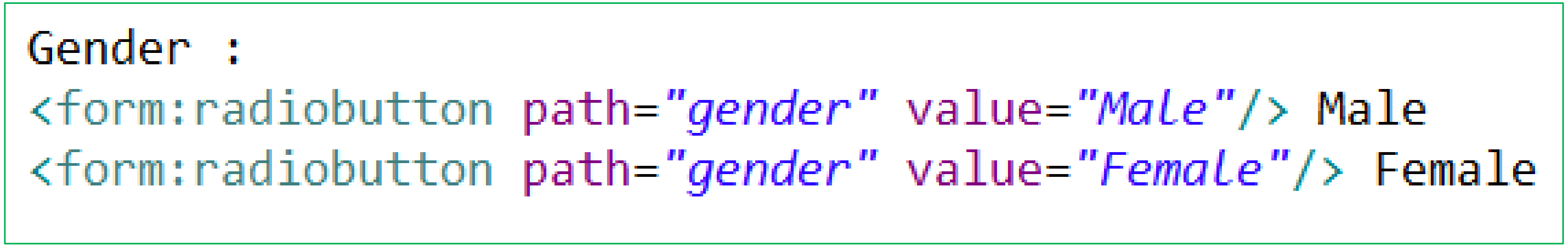


```
 Qualification : ${emp.qual} <br><br>
```

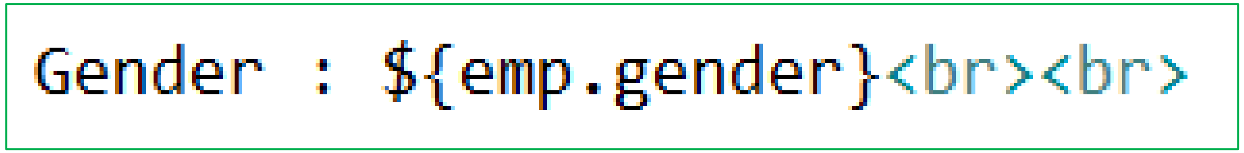
MVC Form Tag - Radio



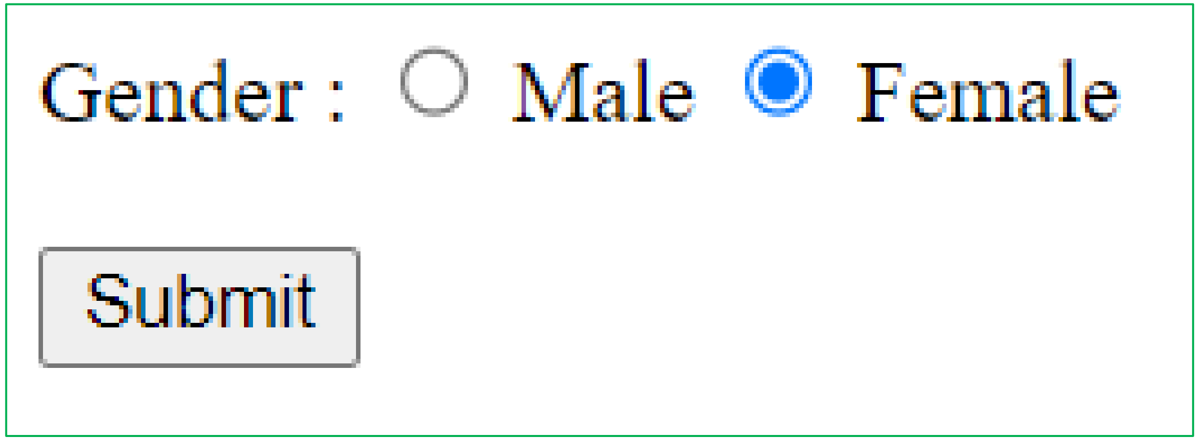
```
private String gender;

public String getGender() {
    return gender;
}
public void setGender(String gender) {
    this.gender = gender;
}
```



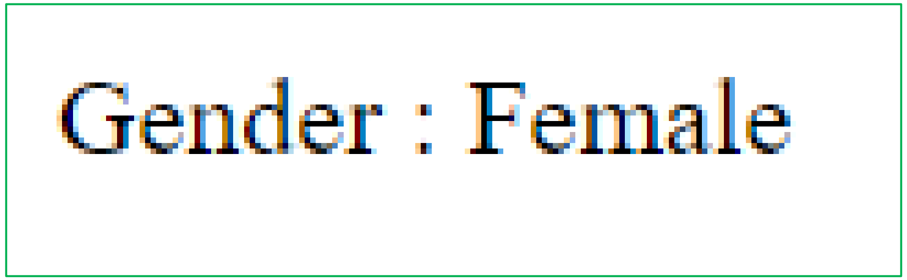
```
Gender :
<form:radiobutton path="gender" value="Male"/> Male
<form:radiobutton path="gender" value="Female"/> Female
```



```
Gender : ${emp.gender}<br><br>
```

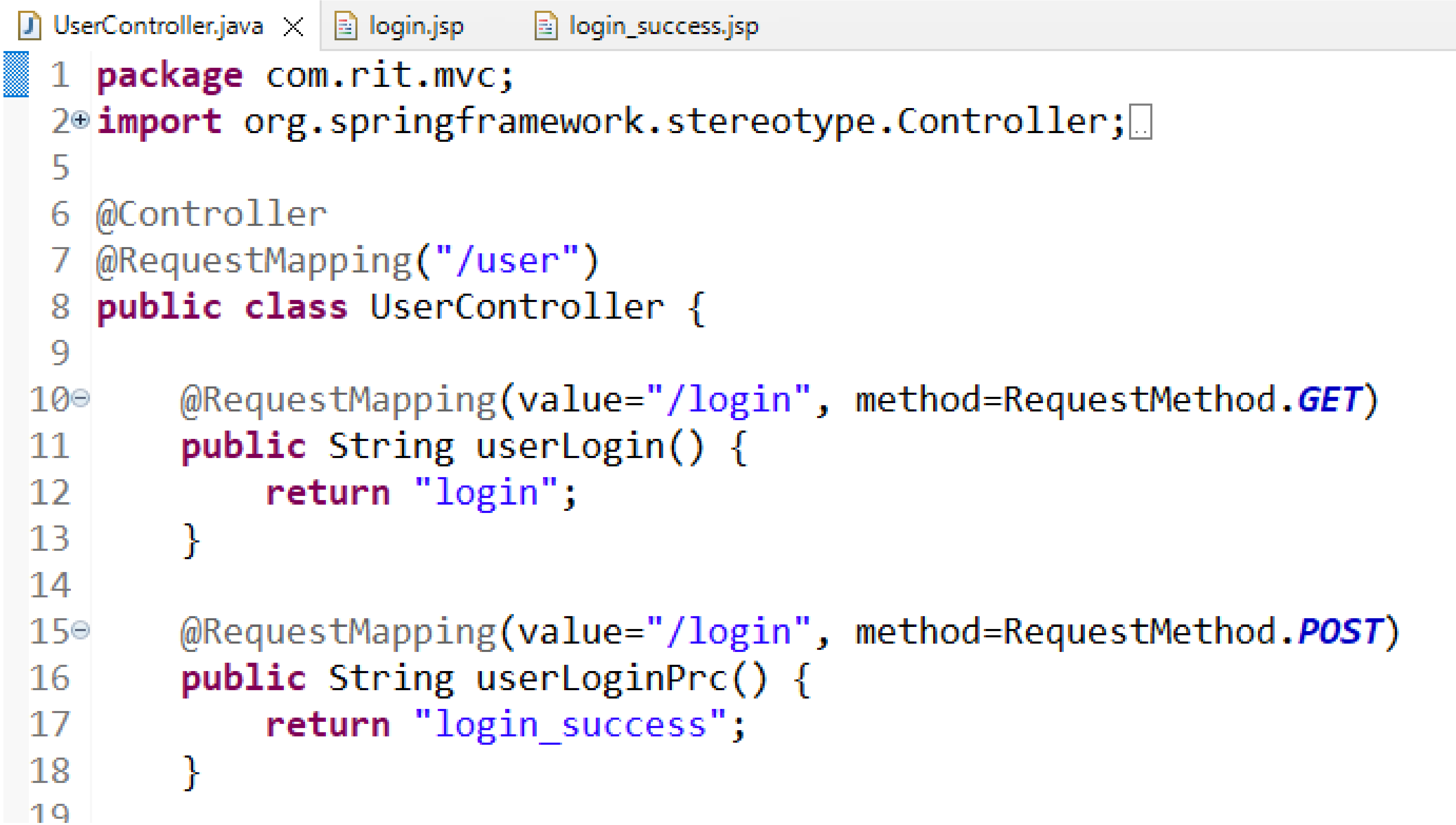


Gender : ○ Male ● Female

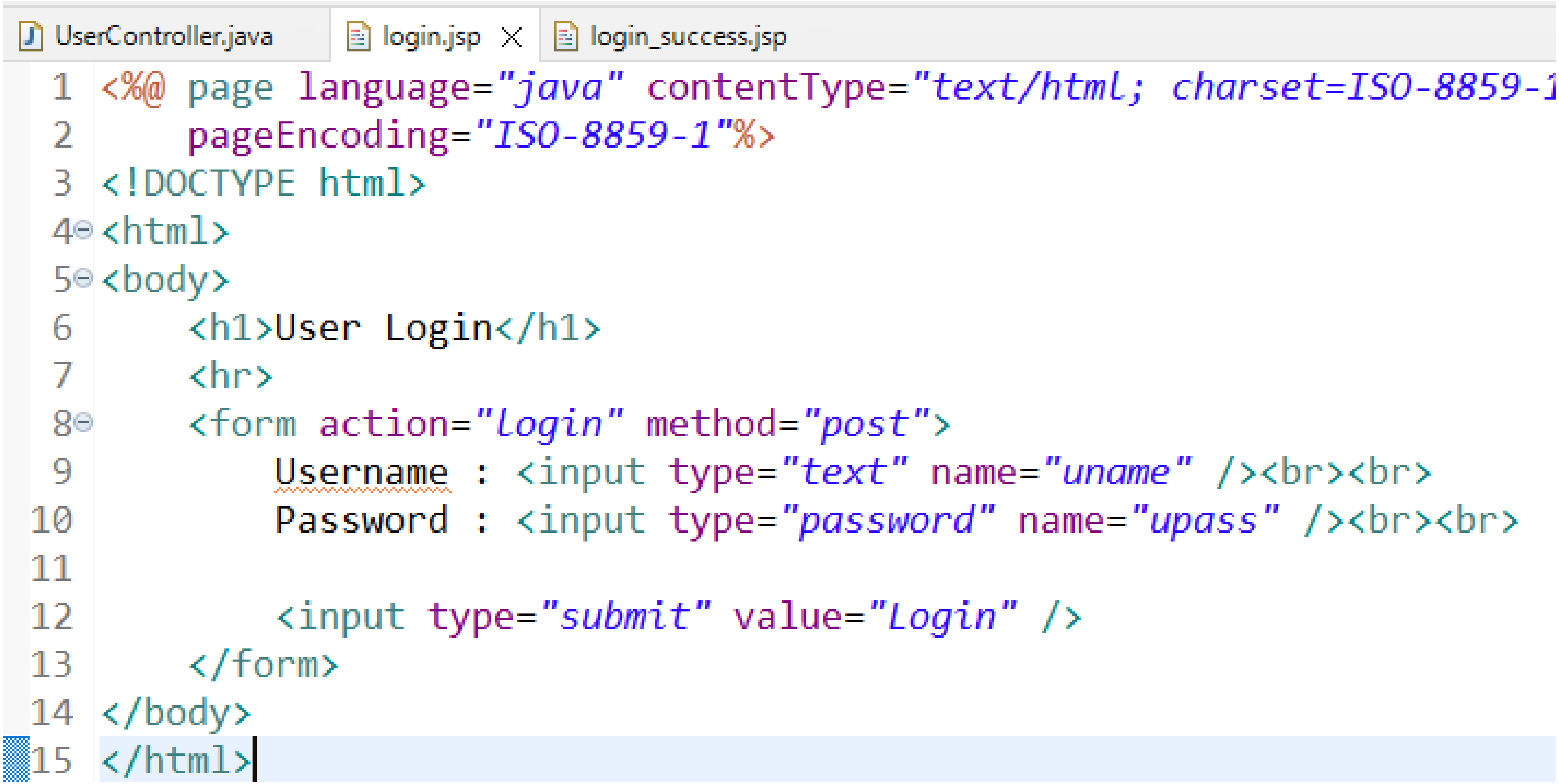


Gender : Female

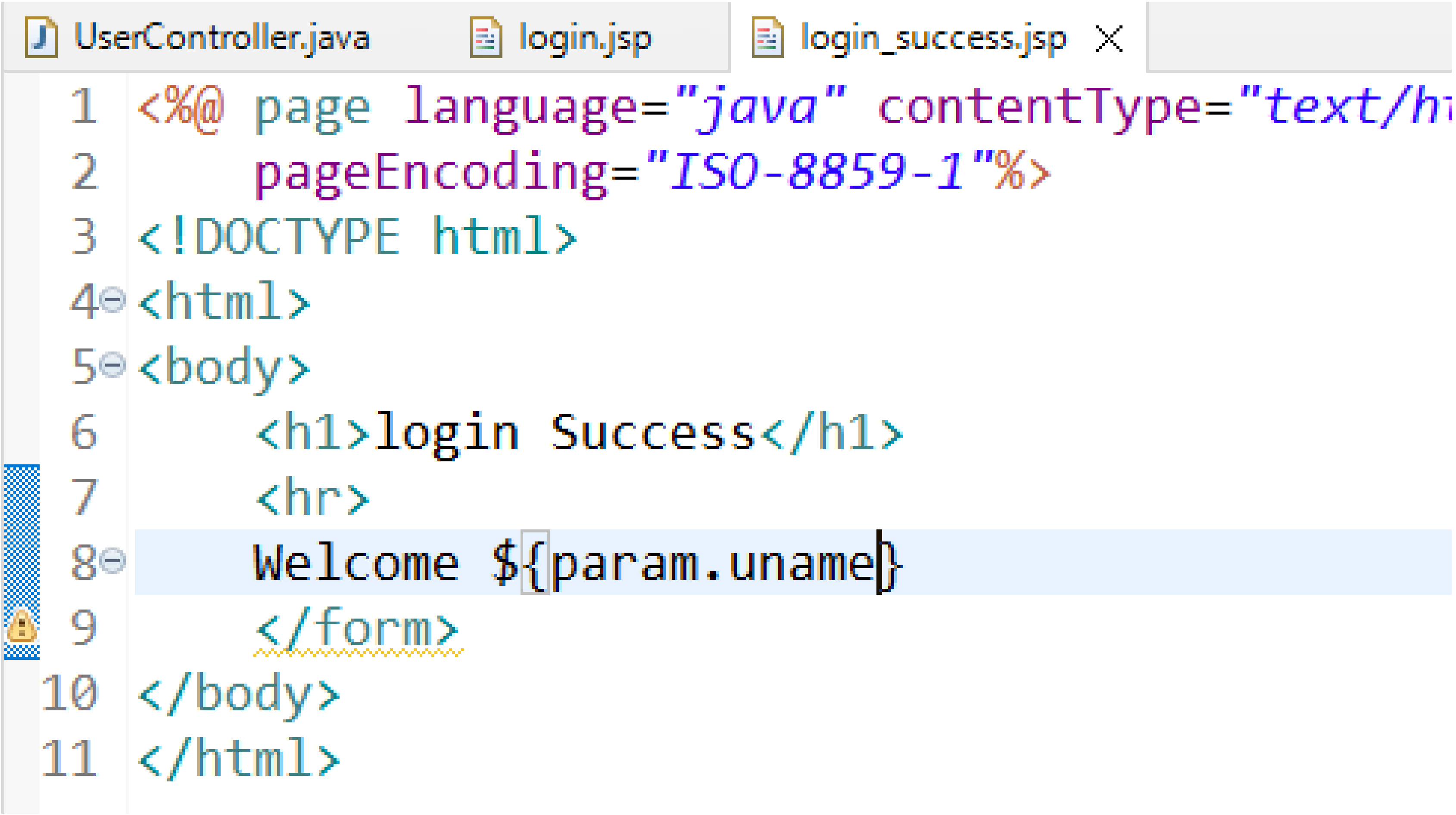
@GetMapping & @PostMapping



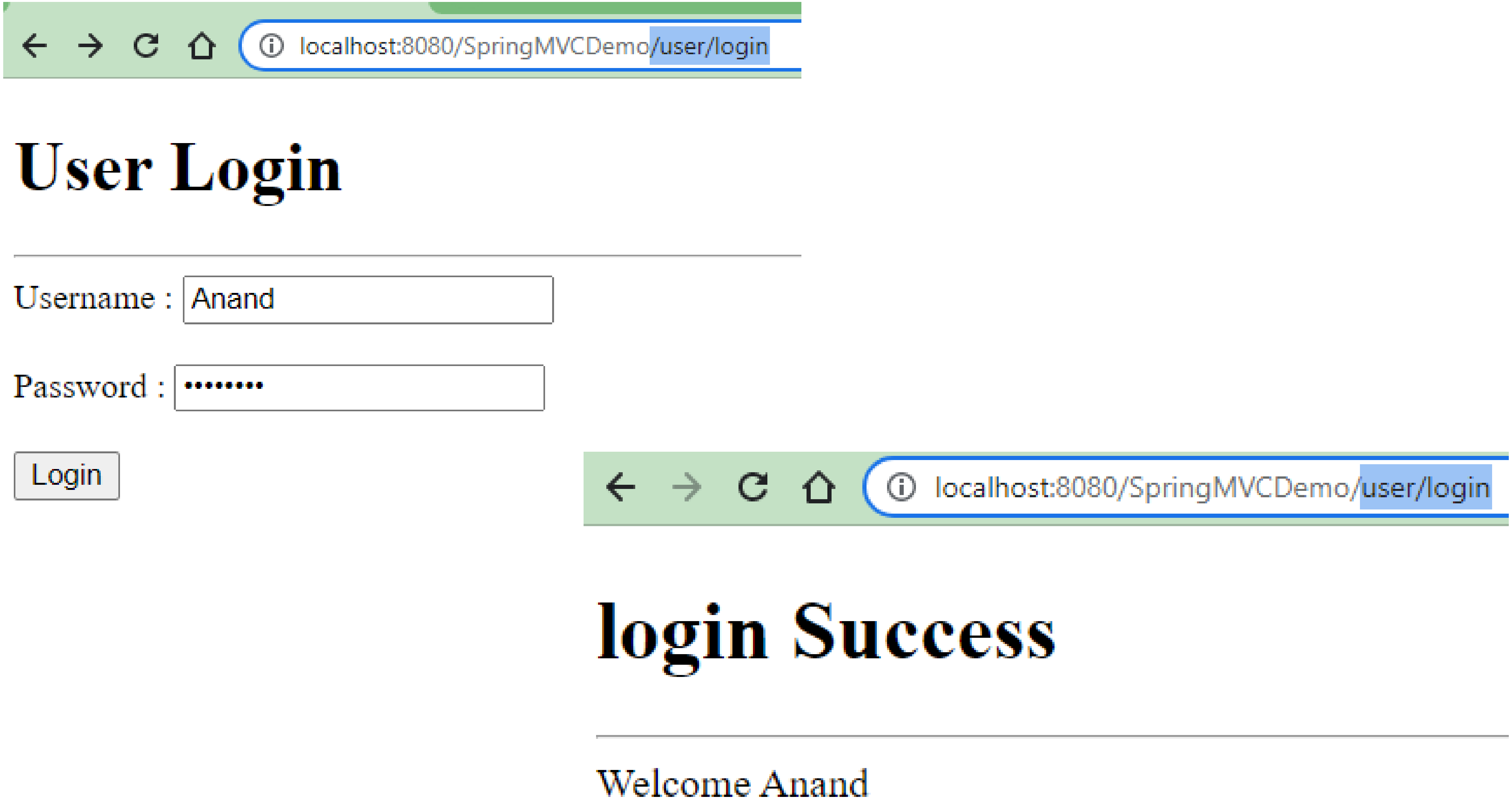
```java
package com.rit.mvc;
import org.springframework.stereotype.Controller;

@Controller
@RequestMapping("/user")
public class UserController {

    @RequestMapping(value="/login", method=RequestMethod.GET)
    public String userLogin() {
        return "login";
    }

    @RequestMapping(value="/login", method=RequestMethod.POST)
    public String userLoginPrc() {
        return "login_success";
    }
```



```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<body>
    <h1>User Login</h1>
    <hr>
    <form action="login" method="post">
        Username : <input type="text" name="uname" /><br><br>
        Password : <input type="password" name="upass" /><br><br>

        <input type="submit" value="Login" />
    </form>
</body>
</html>
```



```
<%@ page language="java" contentType="text/h
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<body>
    <h1>login Success</h1>
    <hr>
    Welcome ${param.uname}
    </form>
</body>
</html>
```



localhost:8080/SpringMVCDemo/user/login

User Login

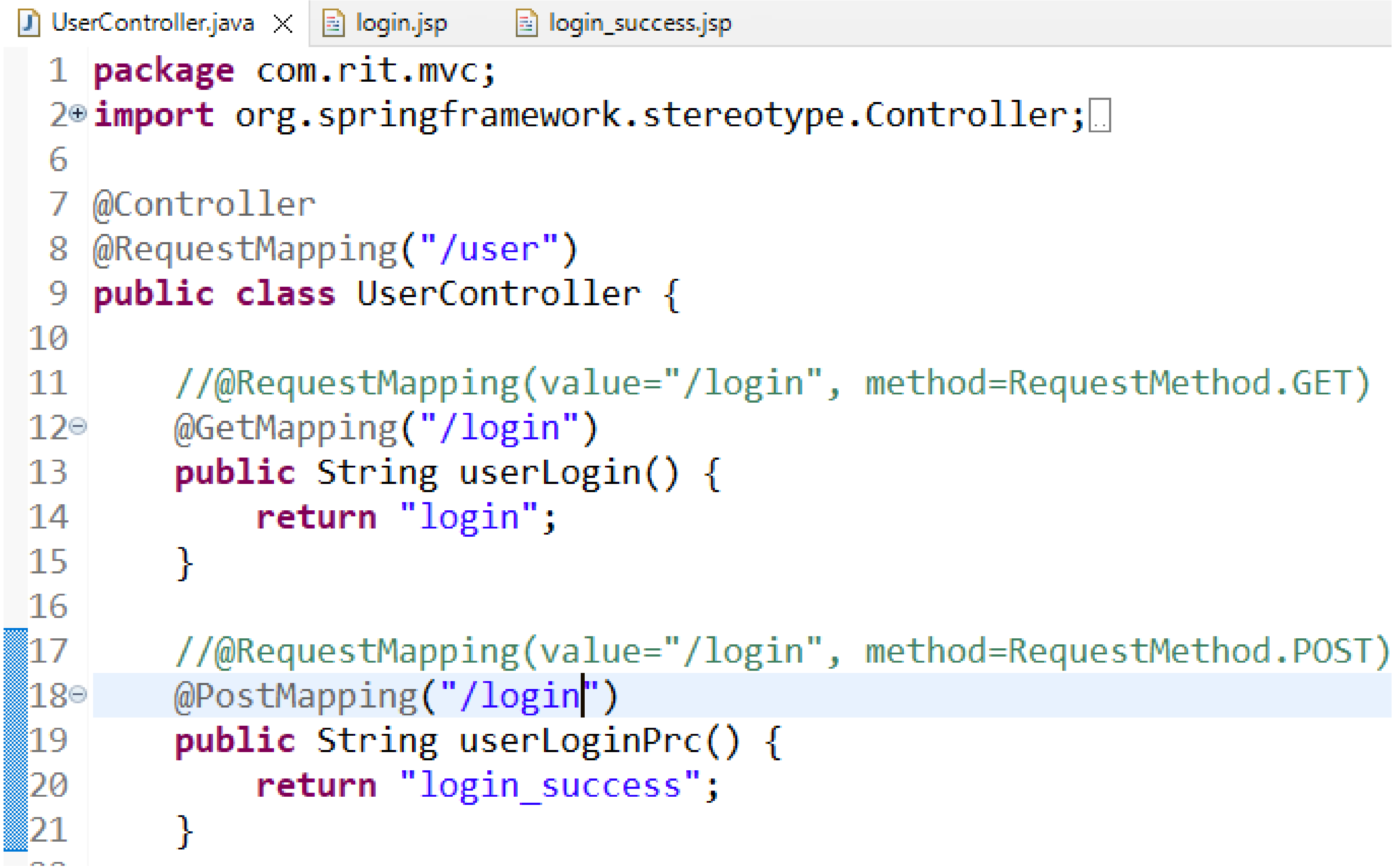
Username : Anand

Password : •••••••••

localhost:8080/SpringMVCDemo/user/login

login Success

Welcome Anand



```java
package com.rit.mvc;
import org.springframework.stereotype.Controller;

@Controller
@RequestMapping("/user")
public class UserController {

    //@RequestMapping(value="/login", method=RequestMethod.GET)
    @GetMapping("/login")
    public String userLogin() {
        return "login";
    }

    //@RequestMapping(value="/login", method=RequestMethod.POST)
    @PostMapping("/login")
    public String userLoginPrc() {
        return "login_success";
    }
```

Thank you