Creating a Spring MVC Project

Estimated time needed: 30 minutes

In this lab, you will use `Spring Initializr` to generate a Spring MVC project and a dynamic website. You will use HTML and CSS for the web page and Thymeleaf for templating.

Learning Objectives

After completing this lab, you will be able to:

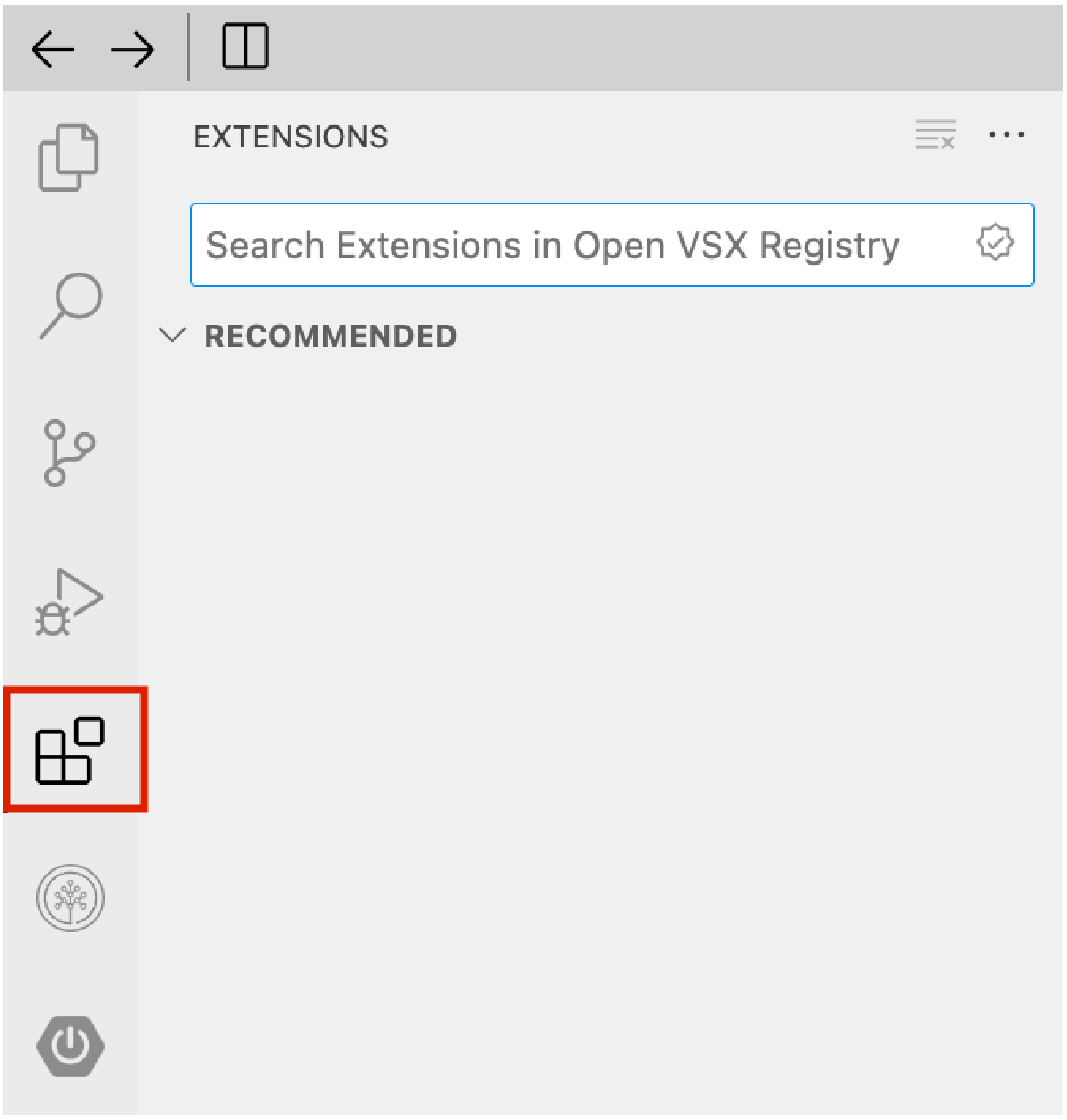
- Use Spring Initializr to generate a project
- Import web and Thymeleaf dependencies to use in an MVC project
- Set up a Spring MVC project structure
- Create webpages and stylesheets to use in a web application
- Create a Controller and configure the routes to webpages
- Run the application

Prerequisites (optional)

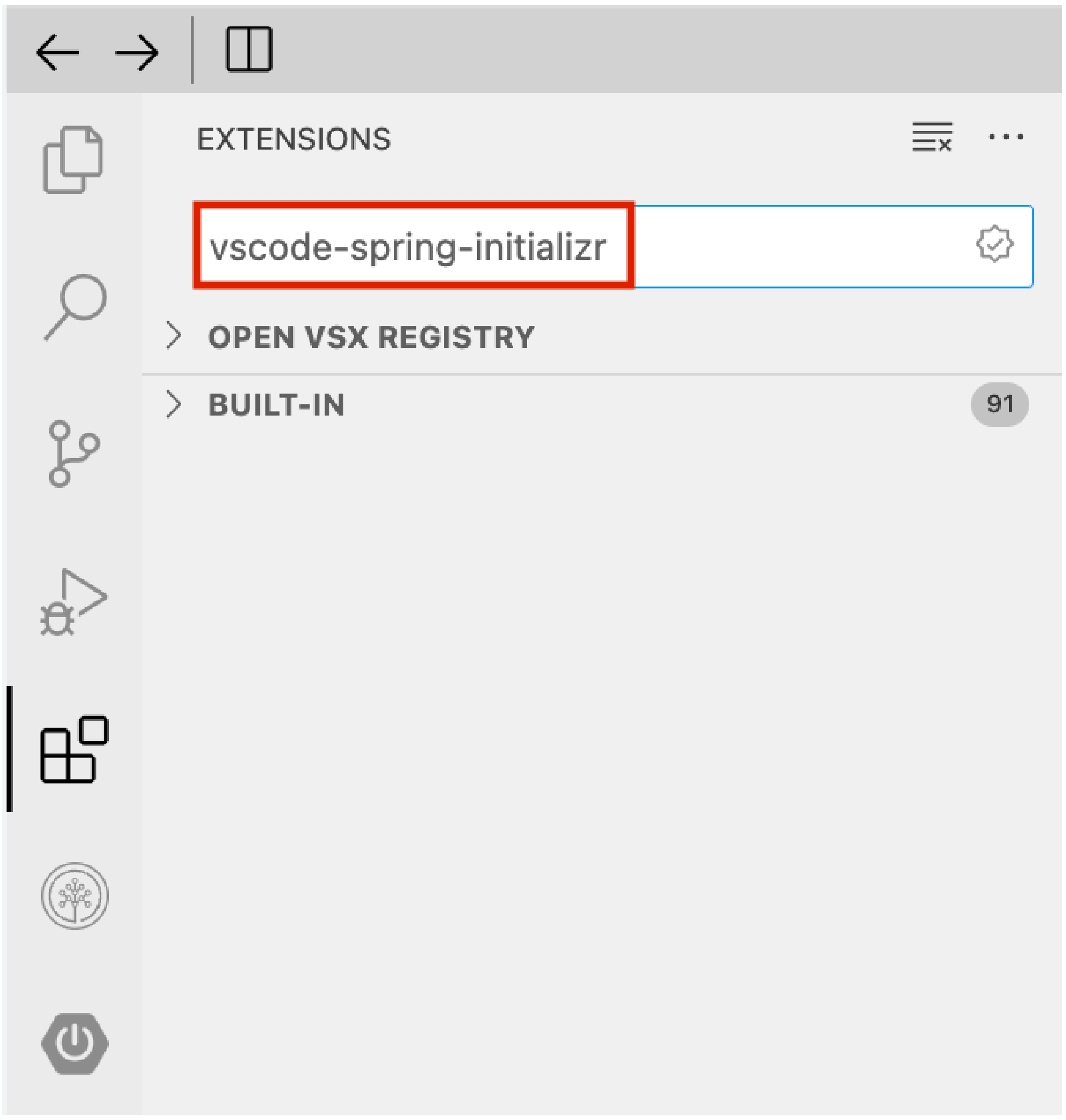
You should have an understanding of basic Java programming before you get started with this lab. Please ensure that you complete the labs sequentially as they are constructed progressively. Some background in HTML and CSS will be useful.

Set up Cloud IDE for Maven Project

1. Click the EXTENSIONS icon to start installing extensions.



2. In the EXTENSIONS search bar, type `vscode-spring-initializr`.

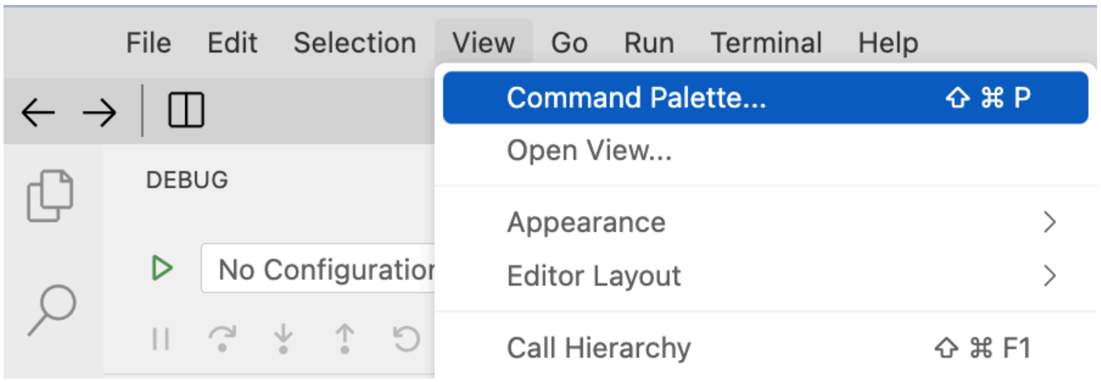


You will get a list of all the extensions you will need for Spring Initializr.

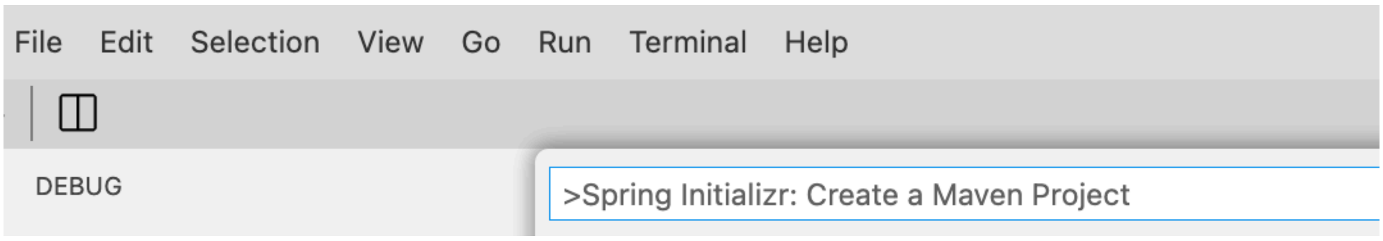
3. Install the `Spring Initializr Java Support` extensions.

Create Maven Project

1. Open Command Palette.

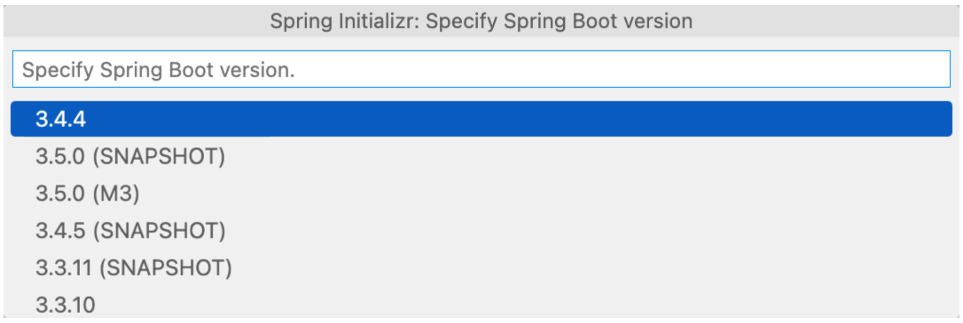


2. In the Command Palette text entry box, type `Spring Initializr: Create a Maven Project`.



>Spring Initializr: Create a Maven Project

3. Specify the Spring Boot version `3.4.4`.

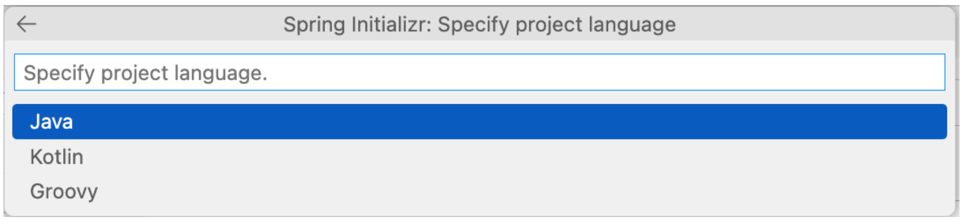


Spring Initializr: Specify Spring Boot version

Specify Spring Boot version.

3.4.4
3.5.0 (SNAPSHOT)
3.5.0 (M3)
3.4.5 (SNAPSHOT)
3.3.11 (SNAPSHOT)
3.3.10

4. Select the project language you want to use. In this case, select `Java`.



Spring Initializr: Specify project language

Specify project language.

Java
Kotlin
Groovy

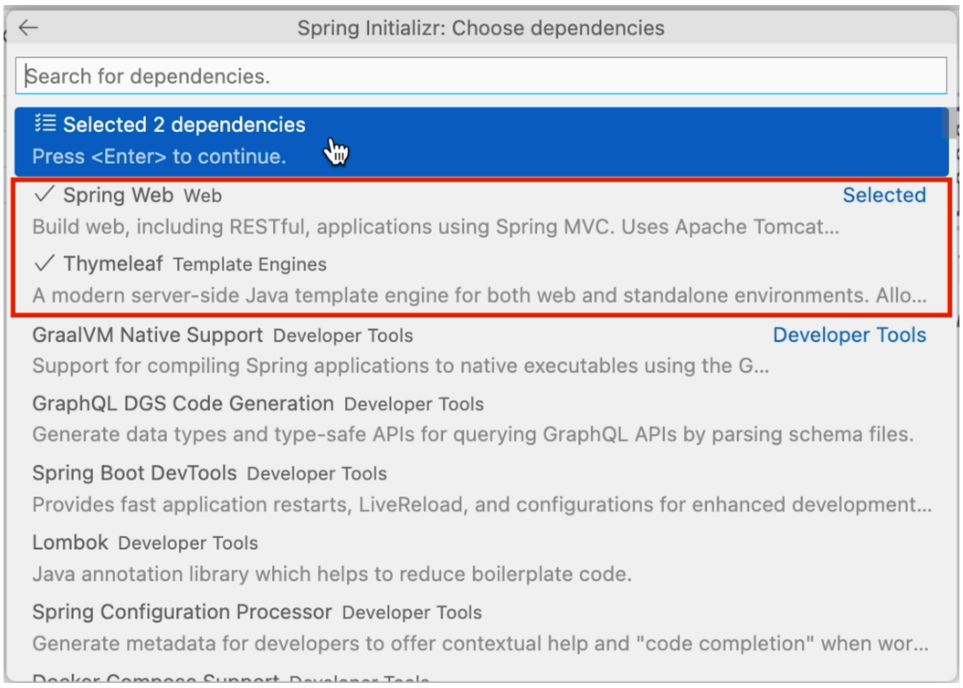
5. Specify `com.example` as the the Group ID.

6. Specify `mvcwebdemo`as the Artifact ID.

7. Specify `Jar` as the packaging you want to use for Java archives.

8. Specify the Java version you want to use, `21` in the case of the IDE.

9. You will be using `Spring Web` dependency for web applications and REST APIs and `Thymeleaf` for templating the project. Using the search box, search for the option `Spring Web web` and select it. Then search for `Thymeleaf` and select it. Press enter to continue.



Spring Initializr: Choose dependencies

Search for dependencies.

Selected 2 dependencies
Press <Enter> to continue.

✓ Spring Web Web Selected
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat...

✓ Thymeleaf Template Engines
A modern server-side Java template engine for both web and standalone environments. Allo...

GraalVM Native Support Developer Tools Developer Tools
Support for compiling Spring applications to native executables using the G...

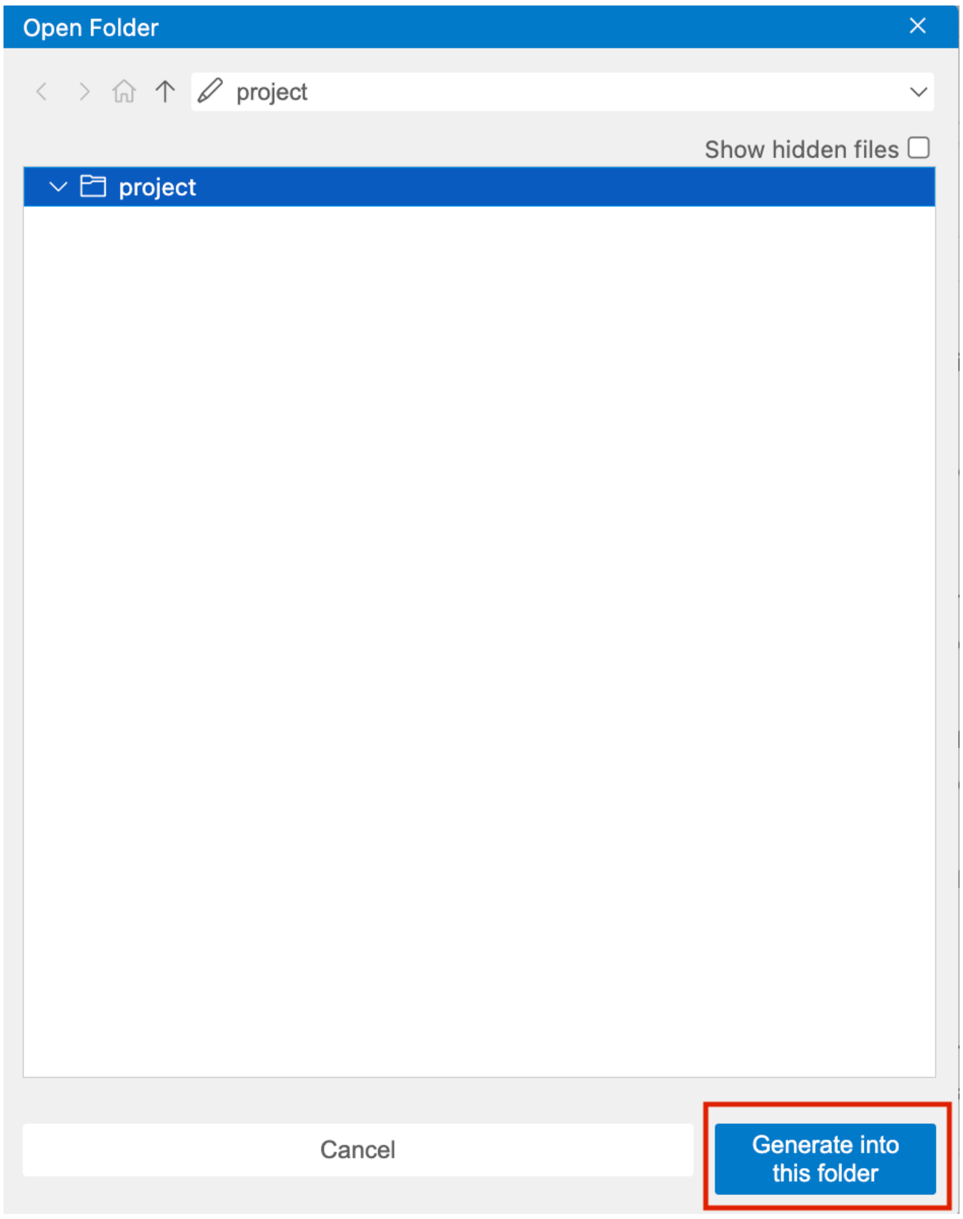
GraphQL DGS Code Generation Developer Tools
Generate data types and type-safe APIs for querying GraphQL APIs by parsing schema files.

Spring Boot DevTools Developer Tools
Provides fast application restarts, LiveReload, and configurations for enhanced development...

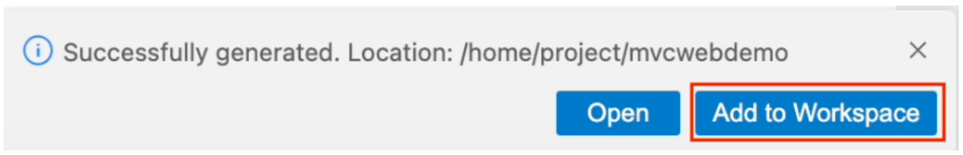
Lombok Developer Tools
Java annotation library which helps to reduce boilerplate code.

Spring Configuration Processor Developer Tools
Generate metadata for developers to offer contextual help and "code completion" when wor...

10. The project is generated by default in the `/home/project` folder. Click the `Generate into this folder` button to generate the project in the desired folder.



11. Add the project to the workspace.


Successfully generated. Location: /home/project/mvcwebdemo

Verify and set up project structure

1. Select the button below to open `pom.xml` to edit the project settings.

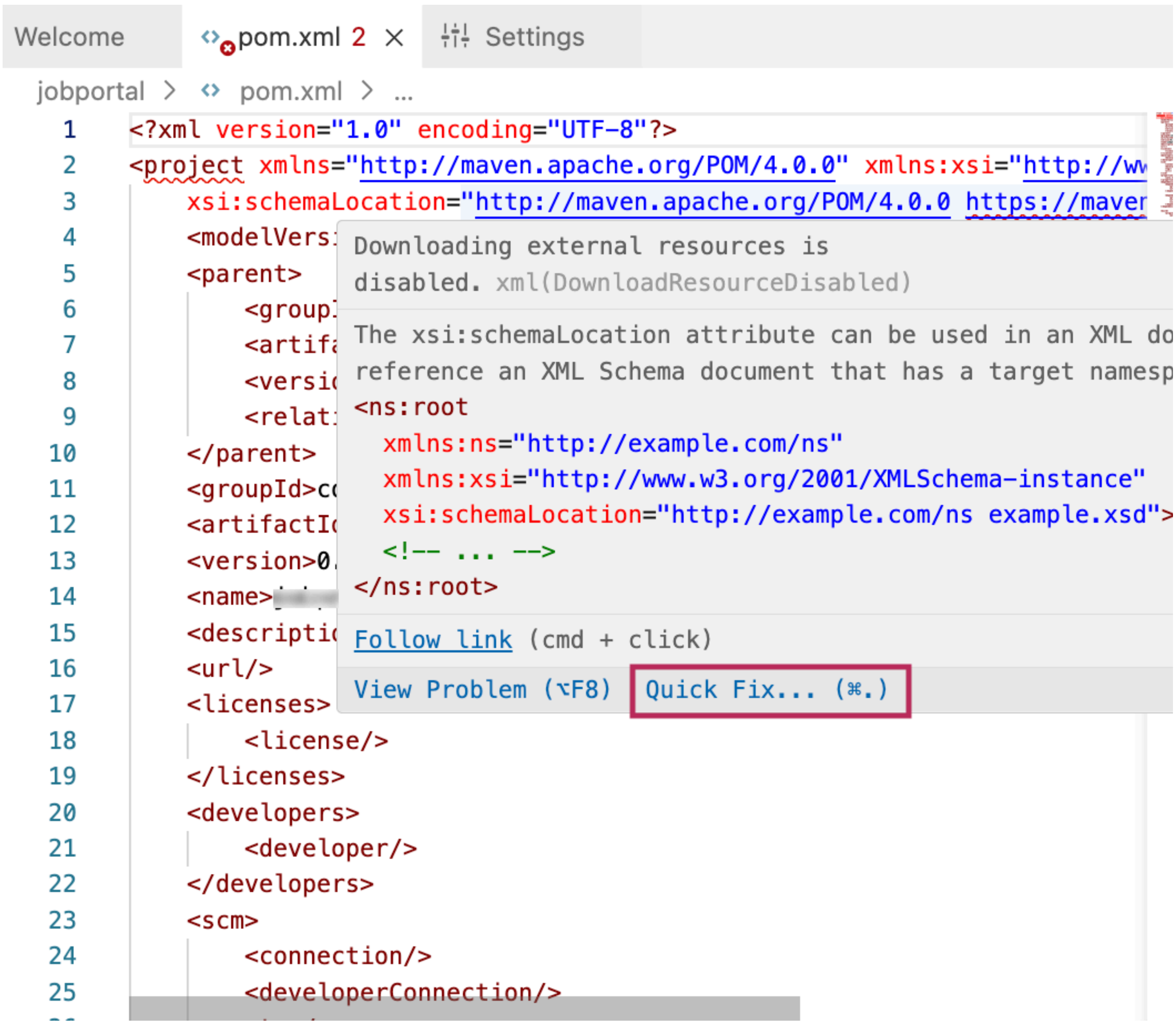
Open **pom.xml** in IDE

Ensure that the required dependencies are included.

- spring-boot-starter-web
- spring-boot-starter-thymeleaf

With these you get a framework for building web applications using the Model-View-Controller (MVC) pattern with an embedded Tomcat server. This allows you to run the web application without a separate application server. Thymeleaf is widely used in Spring-based applications, especially with Spring MVC and Spring Boot. It supports Spring Expression Language (SpEL) and can easily bind data from the backend (model) to the view (HTML).

2. Hover over the highlighted links on `pom.xml` and click. A suggestion comes up. Click on `Quick Fix`.

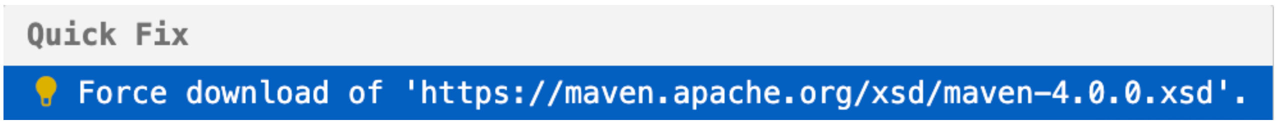


```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://ww
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven
    <modelVers
    <parent>
        <group
        <artifa
        <versio
        <relat
    </parent>
    <groupId>c
    <artifactI
    <version>0.
    <name>[redacted]
    <descripti
    <url/>
    <licenses>
        <license/>
    </licenses>
    <developers>
        <developer/>
    </developers>
    <scm>
        <connection/>
        <developerConnection/>
```

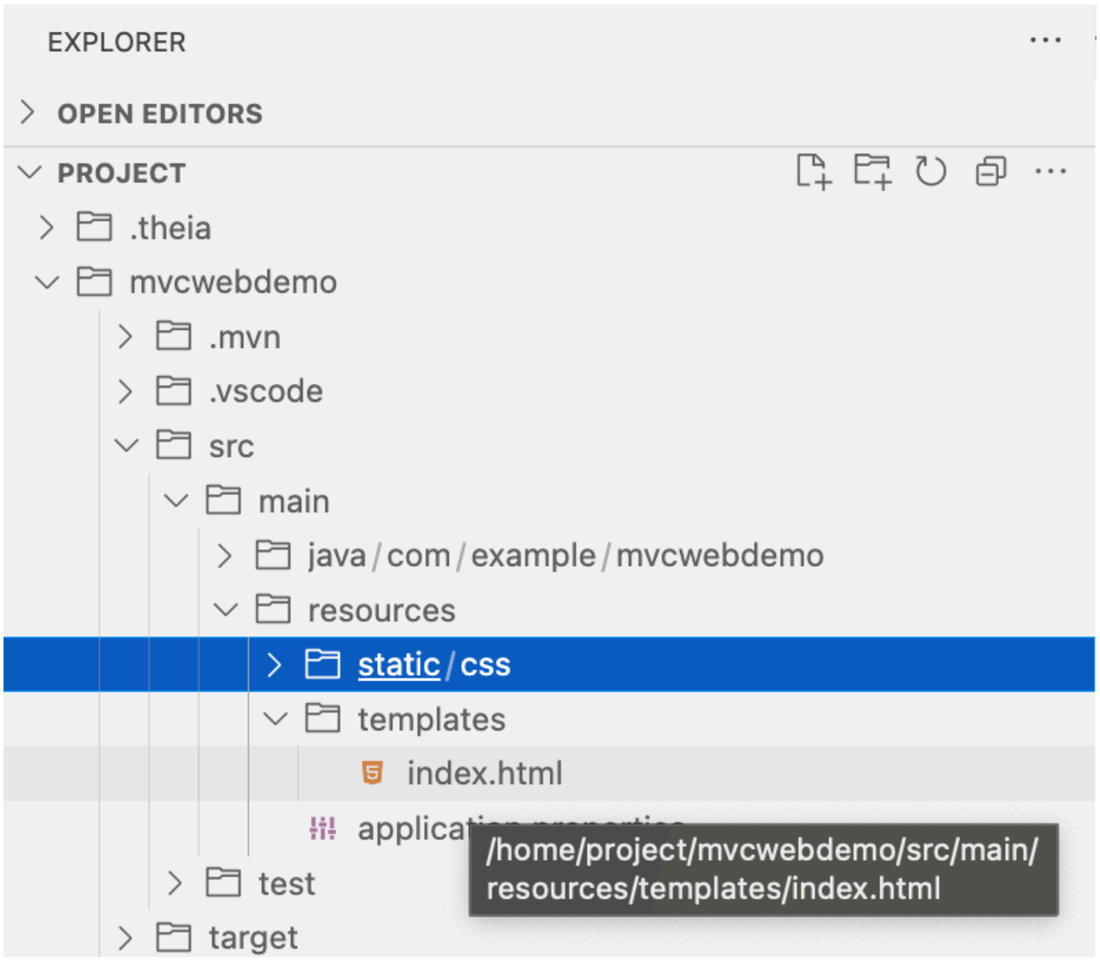
```
Downloading external resources is
disabled. xml(DownloadResourceDisabled)

The xsi:schemaLocation attribute can be used in an XML do
reference an XML Schema document that has a target namesp
<ns:root
  xmlns:ns="http://example.com/ns"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://example.com/ns example.xsd">
  <!-- ... -->
</ns:root>
```

3. Click to download the xsd file.

   The xsd (XML Schema Definition) file in a pom.xml serves as a schema validator that defines the structure, syntax, and data types allowed in the Maven Project Object Model (POM) file.

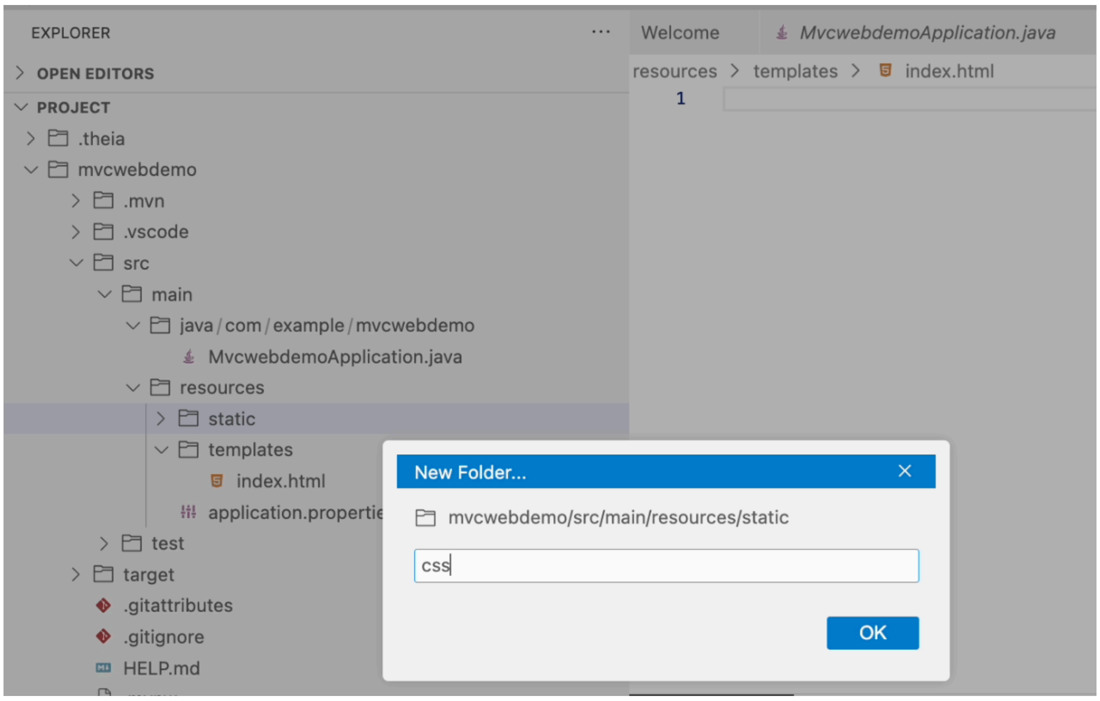

Quick Fix
Force download of 'https://maven.apache.org/xsd/maven-4.0.0.xsd'.

4. Create a new file named `index.html` in the `src/main/resources/templates` folder.



/home/project/mvcwebdemo/src/main/resources/templates/index.html

5. Create a folder named `css` in the `src/main/resources/static` folder.

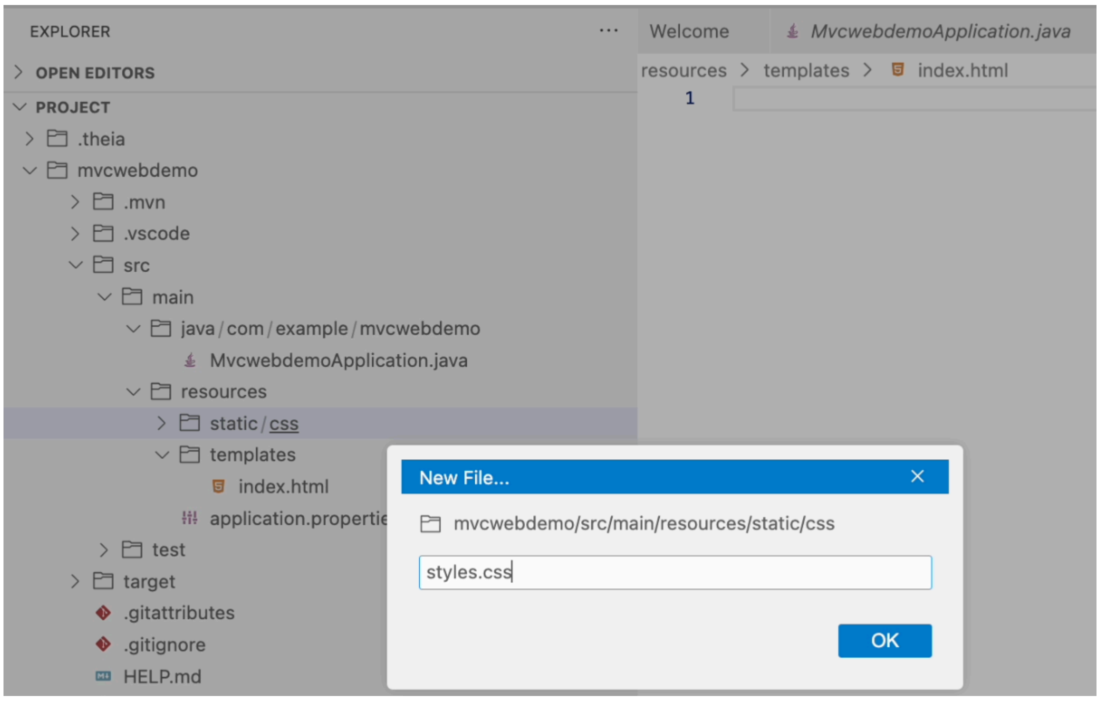


New Folder...

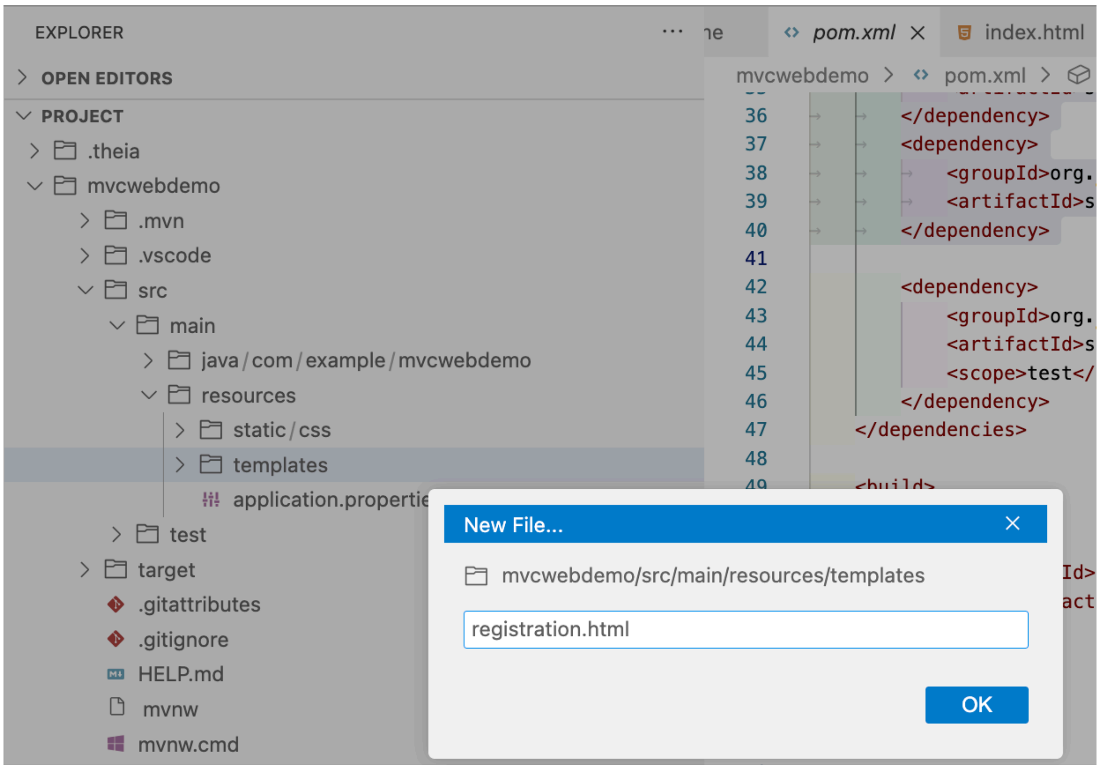
mvcwebdemo/src/main/resources/static

css

6. In the `css` folder that you just created, create `styles.css`. The `index.html` will use this for styling.



7. Create a new file named `registration.html` in the `src/main/resources/templates` folder.



```
        </dependency>
        <dependency>
            <groupId>org.
            <artifactId>s
        </dependency>

        <dependency>
            <groupId>org.
            <artifactId>s
            <scope>test</
        </dependency>
    </dependencies>
```

Add HTML page and CSS

1. Select the Open index.html in IDE button to open the `index.html` that you just created and add the following code in it.

Open **index.html** in IDE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Java Academy</title>
    <link rel="stylesheet" th:href="@{/css/styles.css}">
</head>
<body>
    <!-- Container for aside and main -->
    <div class="container">
        <!-- Aside Section (Navigation Bar) -->
        <aside>
            <nav>
                <ul>
                    <li><a href="#">Home</a></li>
                    <li><a href="#">About</a></li>
                    <li><a href="#">Services</a></li>
                    <li><a href="#">Contact</a></li>
                </ul>
            </nav>
        </aside>
        <!-- Main Content -->
        <main>
            <h1>Welcome to the Java Academy</h1>
            <p>You will learn foundations of Java, OOPs, Spring Boot and Spring MVC here. Below is a link to the registration page:</p>
            <p th:text="${message}">Let's get started</p>
            <a th:href="@{/registration}">Go to Registration Page</a>
        </main>
    </div>
    <!-- Footer -->
    <footer>
        <p>&copy; <span th:text="${#dates.format(#dates.createNow(), 'yyyy')}"></span> Java Academy. All rights reserved.</p>
    </footer>
</body>
</html>
```

`<link rel="stylesheet" th:href="@{/css/styles.css}">` - The th:href attribute is a Thymeleaf attribute used to dynamically resolve URLs. The @{…} syntax is used to create context-relative URLs. In this case, it resolves to /css/styles.css.

The link `Go to Registration Page` is in index.html and it points to http://<hostname>:8080/registration. You will create this mapping in the next step.

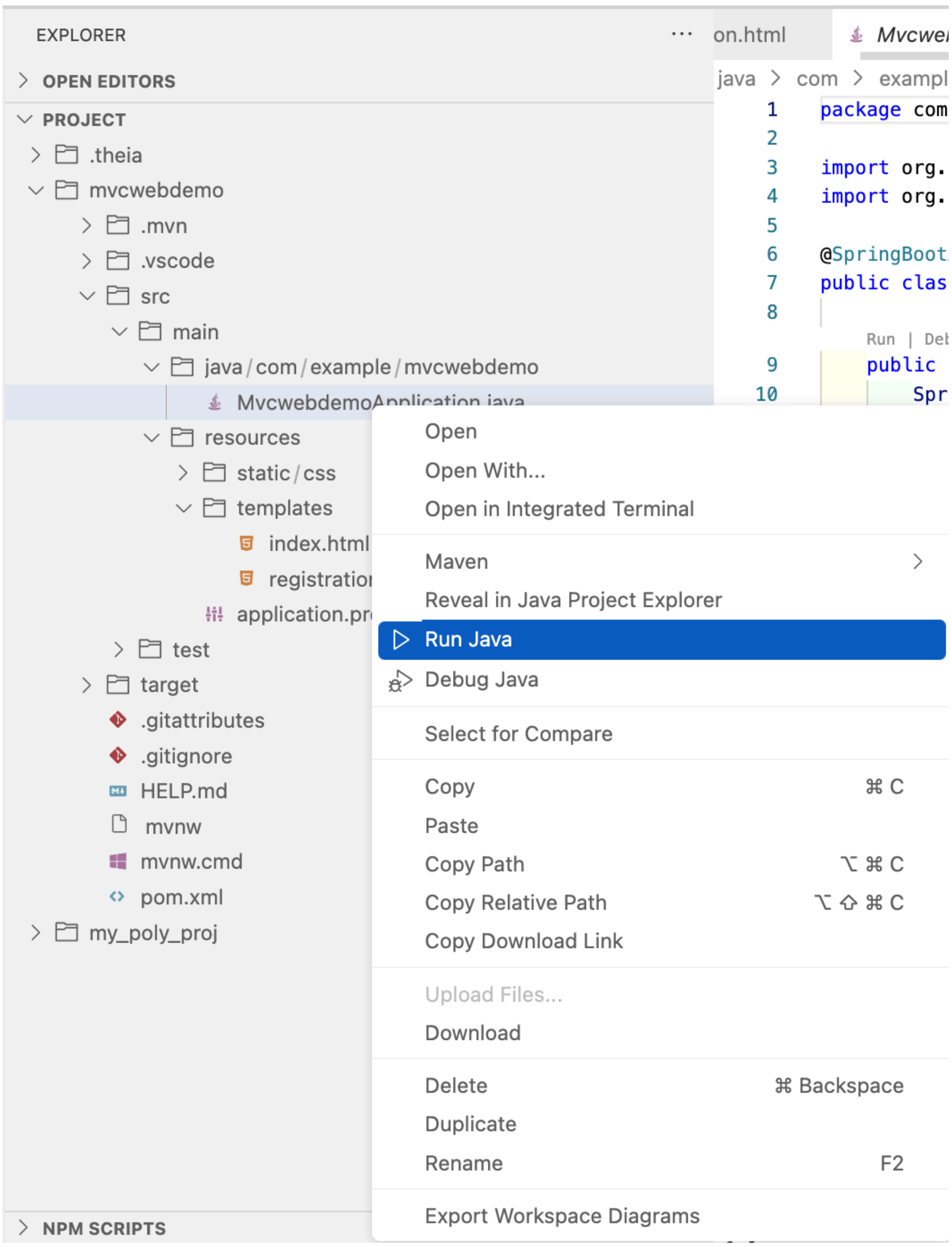
2. Select the Open styles.css in IDE button to open the `styles.css` that you just created and add the following code in it.

Open **styles.css** in IDE

```
/* Basic Reset */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
body {
    font-family: Arial, sans-serif;
    display: flex;
    flex-direction: column; /* Make body a column flex container */
    min-height: 100vh; /* Ensure body takes full viewport height */
}
/* Container for aside and main */
.container {
    display: flex;
    flex: 1; /* Take up remaining space */
}
/* Aside (Navigation Bar) */
aside {
    width: 200px;
    background-color: #333;
    color: white;
    padding: 20px;
}
aside nav ul {
    list-style-type: none;
}
aside nav ul li {
    margin-bottom: 15px;
}
aside a {
    color: white;
    font-size: 25px;
    text-decoration: none;
    display: block;
    transition: transform 0.3s ease-in-out; /* Add transition effect */
    margin: 5px;
}
aside a:hover {
    transform: scale(1.2); /* Scale up on hover */
    cursor: pointer; /* Add pointer cursor on hover */
}
/* Main Content */
main {
    flex: 1;
    padding: 20px;
    background-color: #f4f4f4;
}
main h1 {
    margin-bottom: 20px;
}
main a {
    display: inline-block;
    padding: 10px 20px;
    background-color: #28a745;
    color: white;
    text-decoration: none;
    border-radius: 5px;
    margin: 10px;
}
main a:hover {
    background-color: #218838;
}
/* Footer */
footer {
```

```
            background-color: #333;
            color: white;
            text-align: center;
            padding: 10px;
        }
        /* General Reset */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
/* Container for the Form */
.form-container {
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    width: 100%;
    max-width: 400px;
}
/* Heading Styling */
h1 {
    text-align: center;
    margin-bottom: 20px;
    color: #333;
}
/* Form Styling */
form {
    display: flex;
    flex-direction: column;
    margin-left: 20%;
    margin-right: 20%;
}
/* Label Styling */
label {
    margin-bottom: 5px;
    font-weight: bold;
    color: #555;
}
/* Input Field Styling */
input[type="text"],
input[type="email"] {
    padding: 10px;
    margin-bottom: 15px;
    border: 1px solid #ccc;
    border-radius: 4px;
    font-size: 16px;
}
/* Button Styling */
button[type="submit"] {
    padding: 10px;
    background-color: #28a745;
    color: white;
    border: none;
    border-radius: 4px;
    font-size: 16px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}
button[type="submit"]:hover {
    background-color: #218838;
}
/* Link Styling */
a {
    display: block;
    text-align: center;
    margin-top: 15px;
    color: #007bff;
    text-decoration: none;
    font-size: 14px;
}
a:hover {
    text-decoration: underline;
}
```

4. Right-click `MvcwebdemoApplication.java` and run it.



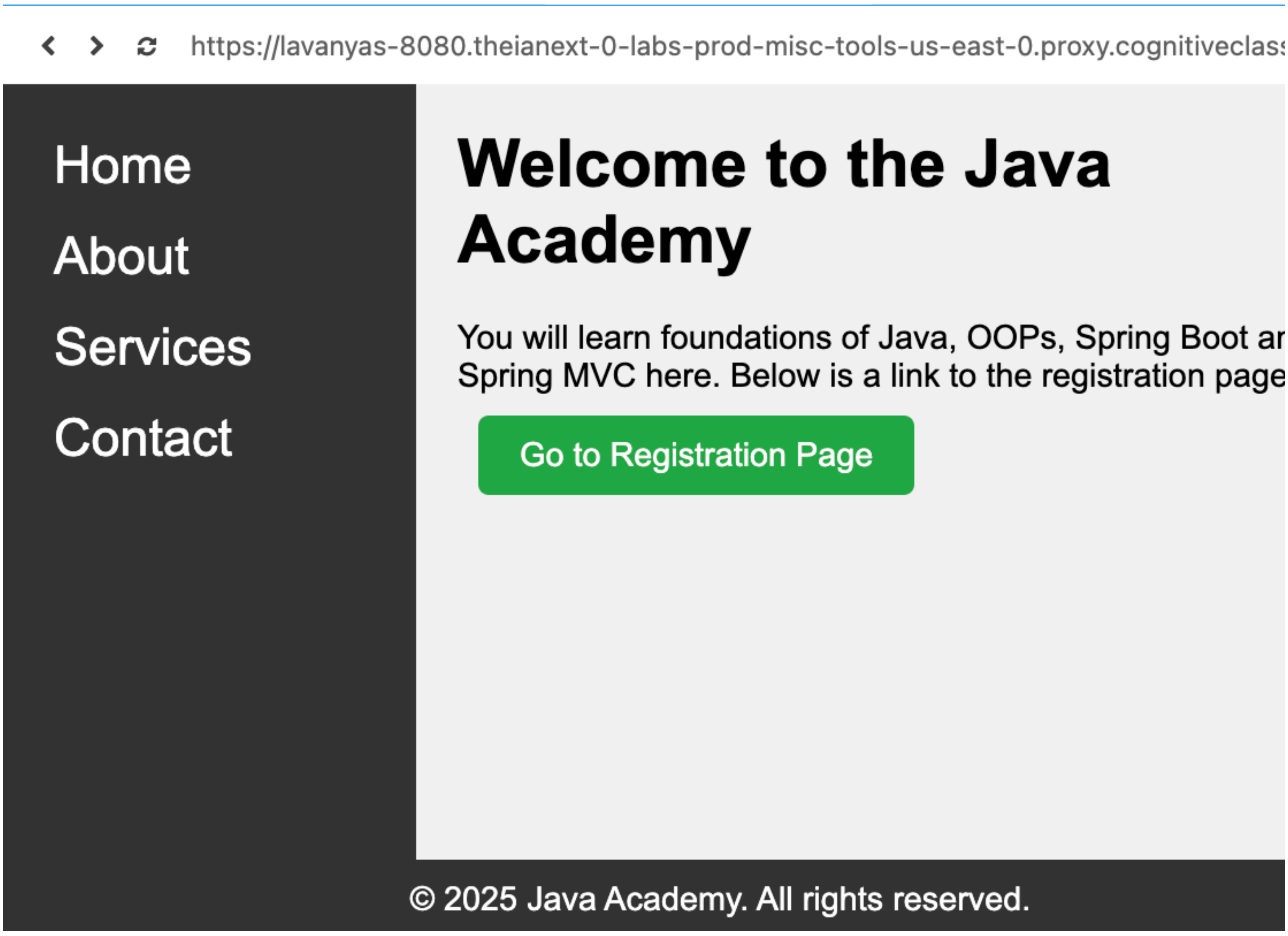
```
package com

import org.
import org.

@SpringBoot
public clas

Run | Deb
    public
        Spr
```

The server will start in port 8080.

5. Click the Website button to open the browser page to access the endpoint.

Website

You should see a page like this.


https://lavanyas-8080.theianext-0-labs-prod-misc-tools-us-east-0.proxy.cognitiveclas

Welcome to the Java Academy

You will learn foundations of Java, OOPs, Spring Boot ar Spring MVC here. Below is a link to the registration page

© 2025 Java Academy. All rights reserved.

Spring Boot treats index.html as the default welcome file. When you access the root URL (/), Spring Boot looks for index.html in the static resource directories and serves it automatically. As you have Thymeleaf configured, `Thymeleaf` is set up to render `index.html` from the `templates` folder. As you placed the index.html in the resource/template, Spring Boot will serve it when you access the root URL (http://localhost:8080/).

The `Registration.html` page and the endpoints are not created. So clicking the button will display the error page.

6. Press `Ctrl+C` to stop the server.

Add content in registration page and configure route

1. Select the Open registration.html in IDE button to open the `registration.html` that you created earlier and add the following code in it.

Open **registration.html** in IDE

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registration Page</title>
    <!-- Link to the external CSS file -->
    <link rel="stylesheet" th:href="@{/css/styles.css}">
</head>
<body>
    <h1>Registration Form</h1>
    <form action="/register" method="post">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required>
        <br>
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required>
        <br>
        <button type="submit">Register</button>
    </form>
    <!-- Link back to the home page -->
    <a th:href="@{/}">Back to Home</a>
</body>
</html>
```

2. You need to configure the route to register.html. For this, you need to create a Controller and add mapping to that. Create a folder named `Controller` under `com/example/mvcwebdemo/controller`.

3. Create a file named `MvcwebdemoController.java` under `com/example/mvcwebdemo/controller`.

4. Add the following code in `MvcwebdemoController.java`.

```
package com.example.mvcwebdemo.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
@Controller
public class MvcwebdemoController {
    // Serve the index.html page
    @GetMapping("/")
    public String home() {
        return "index"; // Refers to src/main/resources/templates/index.html
    }
    // Serve the registration.html page
    @GetMapping("/registration")
    public String registration() {
        return "registration";
    }
}
```

The registration mapping, refers to src/main/resources/templates/registration.html as Thymeleaf is configured to look for HTML pages in the templates folder.

Practice exercise

1. Add a `contactus.html` page to the templates.

2. Configure the endpoint for `contactus` in the `MvcwebdemoController.java`.

3. Make changes in `index.html` to go to the `contactus` page when the menu option is selected.

Conclusion

In this lab you have:

- Used Spring Initializr to generate a project
- Imported web and thymeleaf dependencies to use in an MVC project
- Set up a Spring MVC project structure
- Created webpages and stylesheets to use in the web application
- Created a Controller and configured the routes to the webpages
- Run the application, accessed the home page, and navigated to the registration page

Author(s)

Lavanya