**Assisted Practice: 2.6 HTTPS for Spring Boot**

This section will guide you to:

* Set up Eclipse to work with Spring Boot using the STS plugin
* Create a self-signed SSL certificate key for localhost
* Create a Controller that will run the site in SSL in the browser

**Development Environment**

* Eclipse IDE for Enterprise Java Developers v2019-03 (4.11.0)
* Apache Tomcat Server v9.0
* JRE: OpenJDK Runtime Environment 11.0.2
* Spring Boot STS 4
* All other software components are configured automatically by Spring Boot

This lab has nine subsections, namely:

* + 1. Installing the STS plugin in Eclipse
    2. Creating a Spring Boot Starter Project which is web enabled
    3. Using keytool utility to create a self-signed SSL certificate key
    4. Creating MainController for showing a page in the browser under SSL
    5. Configuring application.properties to run the site in SSL
    6. Building the project
    7. Publishing and starting the project
    8. Running the project
    9. Pushing the code to your GitHub repositories

**Step 2.6.1:** Installing the Spring Tool Suite plugin in Eclipse

* Spring Tool Suite is already installed as an Eclipse plugin in your practice lab. (Refer FSD: Lab Guide - Phase 3 to verify the installation.)

**Step 2.6.2:** Creating a Spring Boot Starter Project which is web enabled

* Open Eclipse
* Go to the **File** menu. Choose **New->Other**
* In the **Wizard** list select **Spring Boot->Spring Starter Project**
* In **Name** enter SpringSSL, **Type** as Maven, **Packaging** as Jar, **Group** as com.ecommerce, and **Package** as com.ecommerce
* Click on **Next**
* In the list of **Available** dependencies, scroll down to select **Web->Spring Web Starter**
* Click on **Next**
* Click on **Finish**
* This will create the project files in the Project Explorer

**Step 2.6.3:** Using keytool utility to create a self-signed SSL certificate key

* In the terminal window of your computer, type the following command:

**keytool -genkey -alias selfsigned\_localhost\_sslserver -keyalg RSA -keysize 2048 -validity 700 -keypass changeit -storepass changeit -keystore ssl-server.jks**

* For first and last name, enter **localhost**
* For organizational unit, enter **Website**
* For name of your organization, enter **Website**
* For city or locality, enter **New York**
* For State or Province, enter **NY**
* For two-letter country code, enter **US**
* For final confirmation, enter **yes**
* This will create a file called ssl-server.jks in your current working directory
* Copy this file into the **src->main->resources** folder

**Step 2.6.4:** Creating MainController for showing a page in the browser under SSL

* In the Project Explorer, expand **SpringSSL->src->main>java**
* Right click on **java** and click on **New->Other**
* In the **Wizard** list, choose **Class** and click on **Next**
* In **Package,** enter com.ecommerce.controllers and in **Name,** enter MainController and click on **Finish**
* Add the following code:

**package** com.ecommerce.controllers;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.RequestBody;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RequestMethod;

**import** org.springframework.web.bind.annotation.ResponseBody;

**@Controller**

**public** **class** MainController {

**@Autowired**

**private** ProductRepository repository;

**@RequestMapping("/")**

**@ResponseBody**

**public** **String** index() {

**return** “This is running under SSL”;

}

}

**Step 2.6.5:** Configuring application.properties to run the site in SSL

* In the Project Explorer, expand **SpringSSL->src->main>resources**
* Double click on **application.properties**
* Enter the following data:

server.port=8443

server.ssl.key-alias=selfsigned\_localhost\_sslserver

server.ssl.key-password=changeit

server.ssl.key-store=classpath:ssl-server.jks

server.ssl.key-store-provider=SUN

server.ssl.key-store-type=JKS

**Step 2.6.6:** Building the project

* From the **Project** menu at the top, click on **Build**
* If any compile errors are shown, fix them as required

**Step 2.6.7:** Publishing and starting the project

* In the Project Explorer, right click on **SpringSSL->Run As->Spring Boot App**
* Check in the Eclipse Console for the message **Started SpringBootStarterApplication**

**Step 2.6.8:** Running the project

* To run the project, open a web browser and type [**https://localhost:8443/**](https://localhost:8443/secured)to run the app under SSL

**Step 2.6.9:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**