**Hands on 1**

**Create a Spring Web Project using Maven**   
pom.xml:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>spring-web-demo</artifactId>

<version>1.0-SNAPSHOT</version>

<packaging>war</packaging>

<name>Spring Web Demo</name>

<dependencies>

<!-- Spring Web MVC -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.32</version>

</dependency>

<!-- JSTL -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<!-- Servlet API -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>4.0.1</version>

<scope>provided</scope>

</dependency>

</dependencies>

<build>

<finalName>spring-web-demo</finalName>

</build>

</project>

HomeController.java:

**package** com.example.controller;

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

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

**import** org.springframework.ui.Model;

@Controller

**public** **class** HomeController {

@RequestMapping("/")

**public** String home(Model model) {

model.addAttribute("message", "Welcome!");

**return** "Home";

}

}

Dispatcher-serverlet.xml:

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:mvc="http://www.springframework.org/schema/mvc"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

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">

<context:component-scan base-package="com.example.controller" />

<mvc:annotation-driven />

<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<property name="prefix" value="/WEB-INF/views/" />

<property name="suffix" value=".jsp" />

</bean>

</beans>

Home.jsp:

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<html>

<head><title>Home Page</title></head>

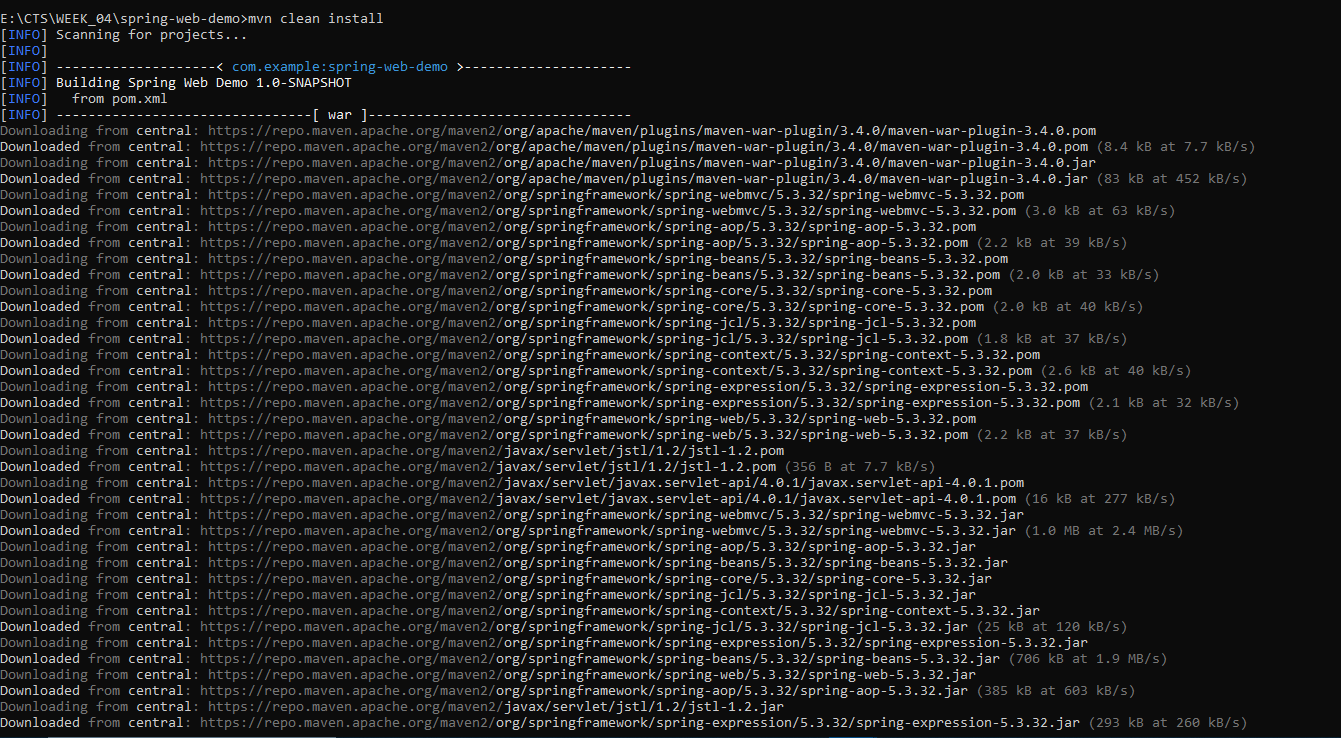
<body>

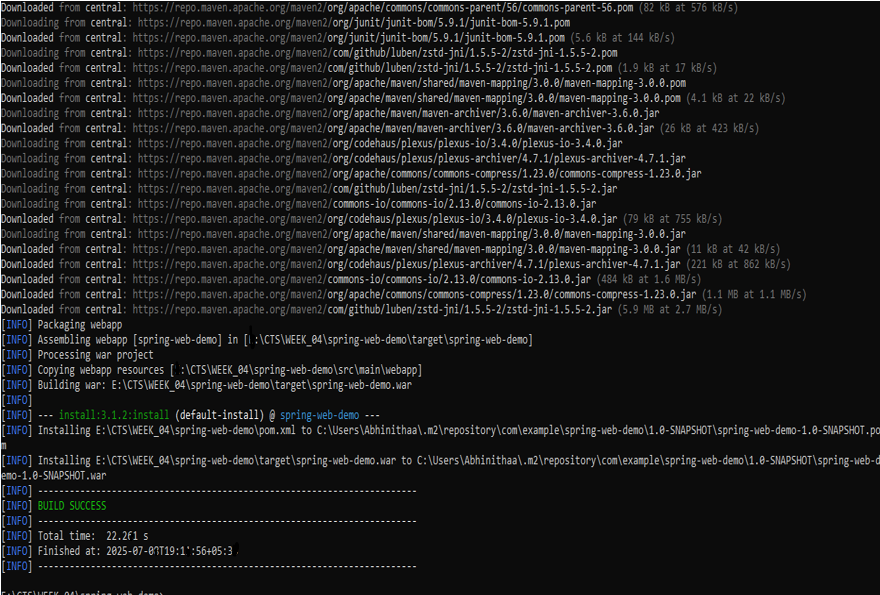
<h2>${message}</h2>

</body>

</html>

CommandPrompt Screenshot:

****



**Hands on 2**

**Spring Core – Load SimpleDateFormat from Spring Configuration XML**

Pom.xml:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>spring-country-xml</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<!-- Spring Core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

</dependencies>

</project>

Country.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"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="country" class="com.example.Country">

<property name="name" value="India"/>

<property name="capital" value="New Delhi"/>

</bean>

</beans>

Application.java:

**package** com.example;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** Application {

**public** **static** **void** main(String[] args) {

ApplicationContext context = **new** ClassPathXmlApplicationContext("country.xml");

Country country = context.getBean("country", Country.**class**);

System.***out***.println("Loaded Country: " + country);

}

}

Country.java:

**package** com.example;

**public** **class** Country {

**private** String name;

**private** String capital;

**public** String getName() {

**return** name;

}

**public** String getCapital() {

**return** capital;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **void** setCapital(String capital) {

**this**.capital = capital;

}

@Override

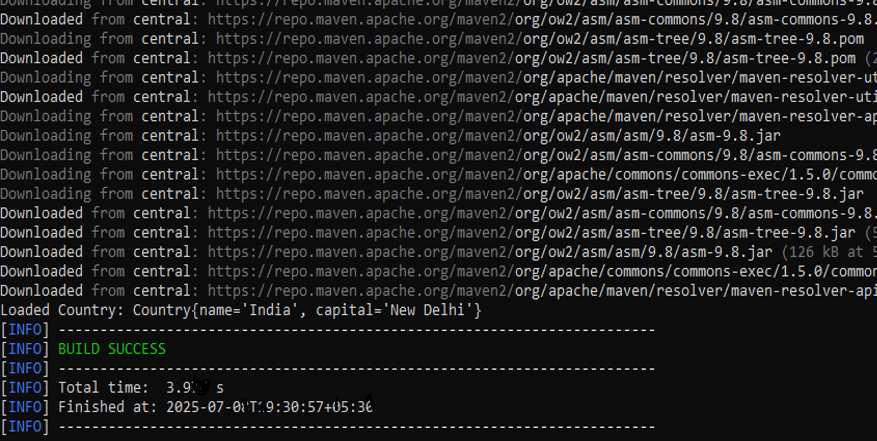
**public** String toString() {

**return** "Country{name='" + name + "', capital='" + capital + "'}";

}

}

CommandPrompt Screenshot:



**Hello World RESTful Web Service**   
  
Write a REST service in the spring learn application created earlier, that returns the text "Hello World!!" using Spring Web Framework. Refer details below:

Pom.xml:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>spring-rest-demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-rest-demo</name>

<properties>

<java.version>17</java.version>

<spring.boot.version>3.2.4</spring.boot.version>

</properties>

<dependencies>

<!-- ✅ Spring Boot Web Starter -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>${spring.boot.version}</version>

</dependency>

<!-- ✅ Spring Boot Starter Test (optional) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<version>${spring.boot.version}</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- ✅ Spring Boot Maven Plugin -->

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<version>${spring.boot.version}</version>

</plugin>

</plugins>

</build>

</project>

Application.properties:

server.port=8083

HelloController.java:

**package** com.example.controller;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

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

@RestController

**public** **class** HelloController {

**private** **static** **final** Logger ***LOGGER*** = LoggerFactory.*getLogger*(HelloController.**class**);

@GetMapping("/hello")

**public** String sayHello() {

***LOGGER***.info("START sayHello()");

String message = "Hello World!!";

***LOGGER***.info("END sayHello()");

**return** message;

}

}

Application.java:

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**public** **class** Application {

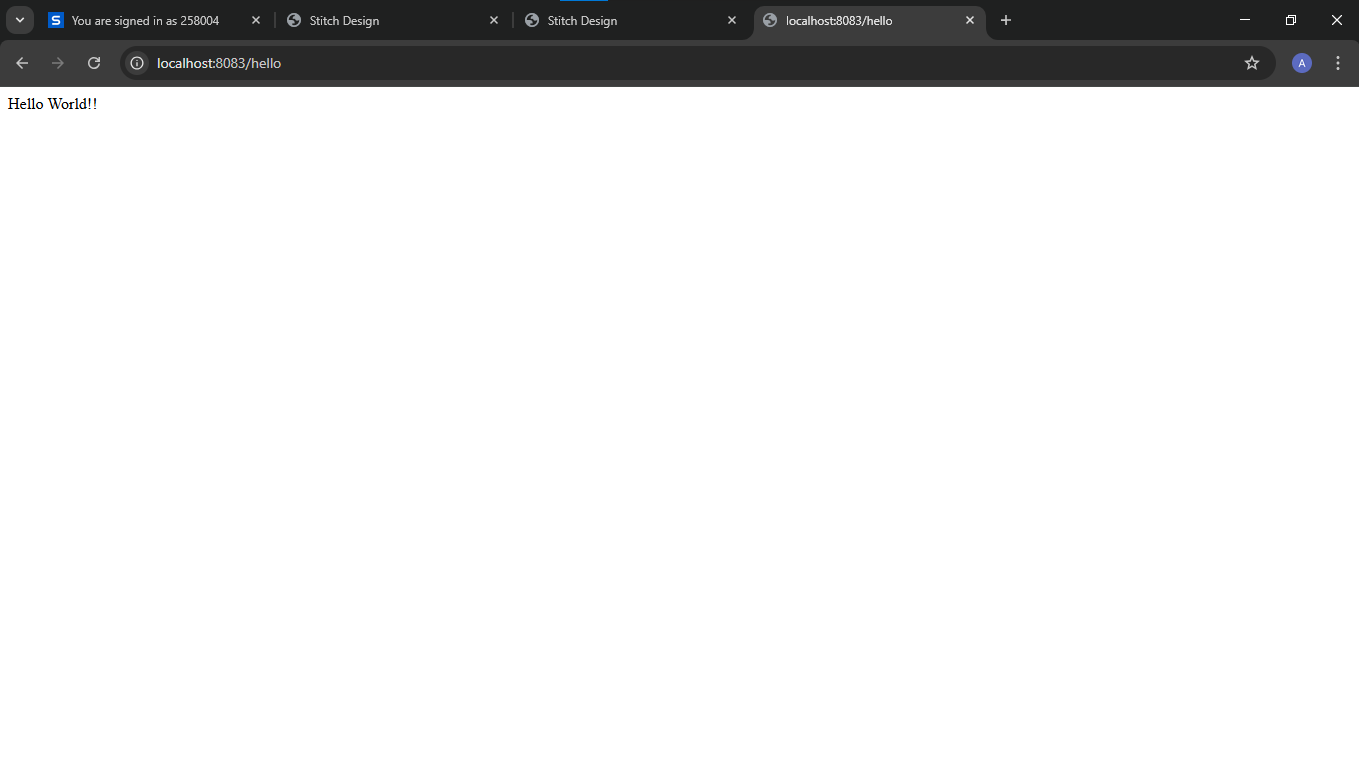
**public** **static** **void** main(String[] args) {

SpringApplication.*run*(Application.**class**, args);

}

}

Web Screenshot:

****

**REST - Country Web Service**   
  
Write a REST service that returns India country details in the earlier created spring learn application.

Pom.xml:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>spring-rest-demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-rest-demo</name>

<properties>

<java.version>17</java.version>

<spring.boot.version>3.2.4</spring.boot.version>

</properties>

<dependencies>

<!-- ✅ Spring Boot Web Starter -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>${spring.boot.version}</version>

</dependency>

<!-- ✅ Spring Boot Starter Test (optional) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<version>${spring.boot.version}</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- ✅ Spring Boot Maven Plugin -->

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<version>${spring.boot.version}</version>

</plugin>

</plugins>

</build>

</project>

Country.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"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="in" class="com.example.model.Country">

<property name="code" value="IN" />

<property name="name" value="India" />

</bean>

</beans>

countryController.java

**package** com.example.controller;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

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

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

**import** com.example.model.Country;

@RestController

**public** **class** countryController {

@GetMapping("/country")

**public** Country getCountryIndia() {

ApplicationContext context = **new** ClassPathXmlApplicationContext("country.xml");

**return** context.getBean("in", Country.**class**);

}

}

Country.java:

**package** com.example.model;

**public** **class** Country {

**private** String code;

**private** String name;

**public** String getCode() {

**return** code;

}

**public** String getName() {

**return** name;

}

**public** **void** setCode(String code) {

**this**.code = code;

}

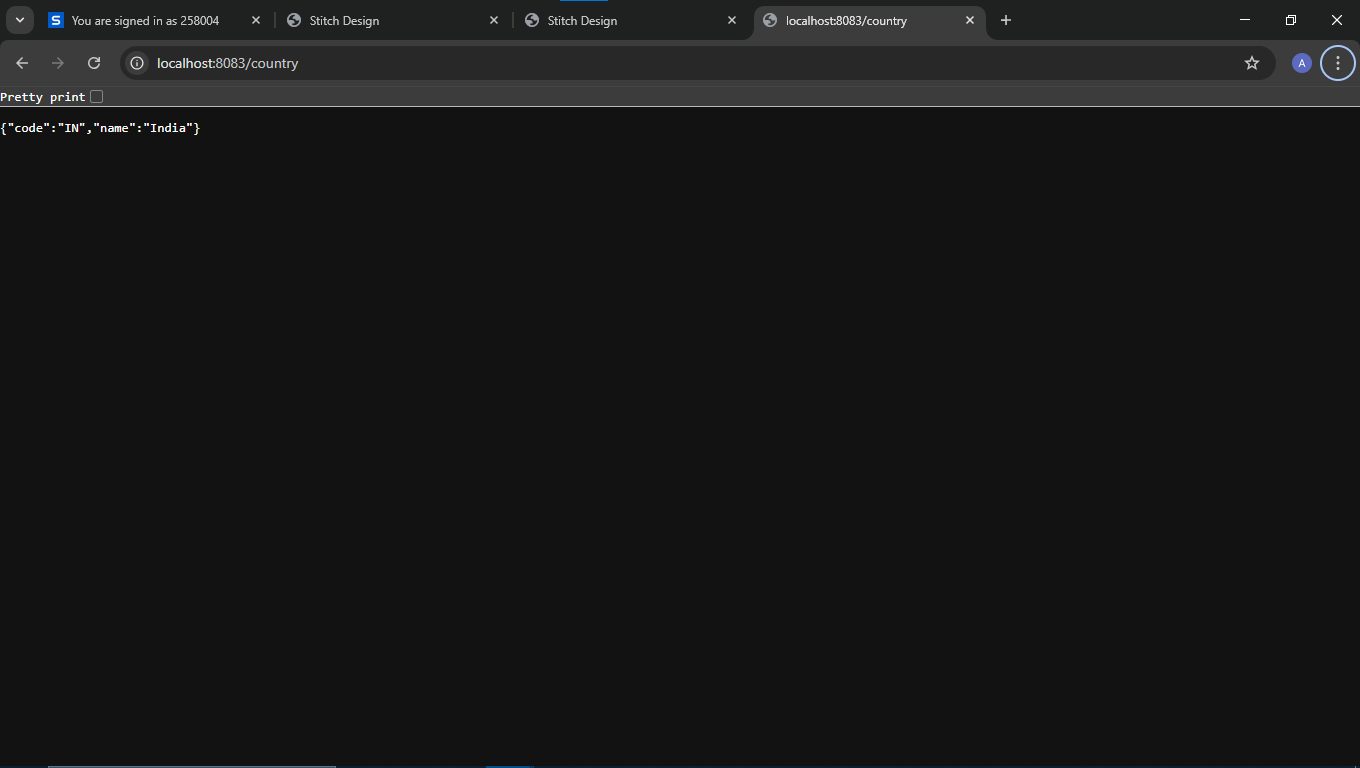
**public** **void** setName(String name) {

**this**.name = name;

}

}

Web Screenshot:

****

**REST - Get country based on country code**   
  
Write a REST service that returns a specific country based on country code. The country code should be case insensitive.

Pom.xml:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>spring-rest-demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-rest-demo</name>

<properties>

<java.version>17</java.version>

<spring.boot.version>3.2.4</spring.boot.version>

</properties>

<dependencies>

<!-- ✅ Spring Boot Web Starter -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>${spring.boot.version}</version>

</dependency>

<!-- ✅ Spring Boot Starter Test (optional) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<version>${spring.boot.version}</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- ✅ Spring Boot Maven Plugin -->

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<version>${spring.boot.version}</version>

</plugin>

</plugins>

</build>

</project>

CountryController.java:

**package** com.example.controller;

**import** com.example.model.Country;

**import** com.example.service.CountryService;

**import** com.example.exception.CountryNotFoundException;

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

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

**import** java.util.\*;

@RestController

**public** **class** CountryController {

@Autowired

**private** CountryService countryService;

@GetMapping("/country")

**public** Country getCountryIndia() {

**return** **new** Country("IN", "India");

}

@GetMapping("/countries")

**public** List<Country> getAllCountries() {

**return** countryService.getAllCountries();

}

@GetMapping("/countries/{code}")

**public** Country getCountryByCode(@PathVariable String code) **throws** CountryNotFoundException {

**return** countryService.getCountry(code);

}

}

CountryService.java:

**package** com.example.service;

**import** java.util.\*;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**import** org.springframework.stereotype.Service;

**import** com.example.exception.CountryNotFoundException;

**import** com.example.model.Country;

@Service

**public** **class** CountryService {

**public** Country getCountry(String code) **throws** CountryNotFoundException {

ApplicationContext context = **new** ClassPathXmlApplicationContext("country.xml");

List<Country> countries = (List<Country>) context.getBean("countryList");

**return** countries.stream()

.filter(c -> c.getCode().equalsIgnoreCase(code))

.findFirst()

.orElseThrow(() -> **new** CountryNotFoundException("Country not found: " + code));

}

**public** List<Country> getAllCountries() {

ApplicationContext context = **new** ClassPathXmlApplicationContext("country.xml");

**return** (List<Country>) context.getBean("countryList");

}

}

Country.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"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

https://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="in" class="com.example.model.Country">

<property name="code" value="IN" />

<property name="name" value="India" />

</bean>

<bean id="countryList" class="java.util.ArrayList">

<constructor-arg>

<list>

<ref bean="in"/>

<bean class="com.example.model.Country">

<property name="code" value="US"/>

<property name="name" value="United States"/>

</bean>

<bean class="com.example.model.Country">

<property name="code" value="JP"/>

<property name="name" value="Japan"/>

</bean>

<bean class="com.example.model.Country">

<property name="code" value="DE"/>

<property name="name" value="Germany"/>

</bean>

</list>

</constructor-arg>

</bean>

</beans>

Country.java:

**package** com.example.model;

**public** **class** Country {

**private** String code;

**private** String name;

**public** Country() {}

**public** Country(String code, String name) {

**this**.code = code;

**this**.name = name;

}

**public** String getCode() {

**return** code;

}

**public** **void** setCode(String code) {

**this**.code = code;

}

**public** String getName() {

**return** name;

}

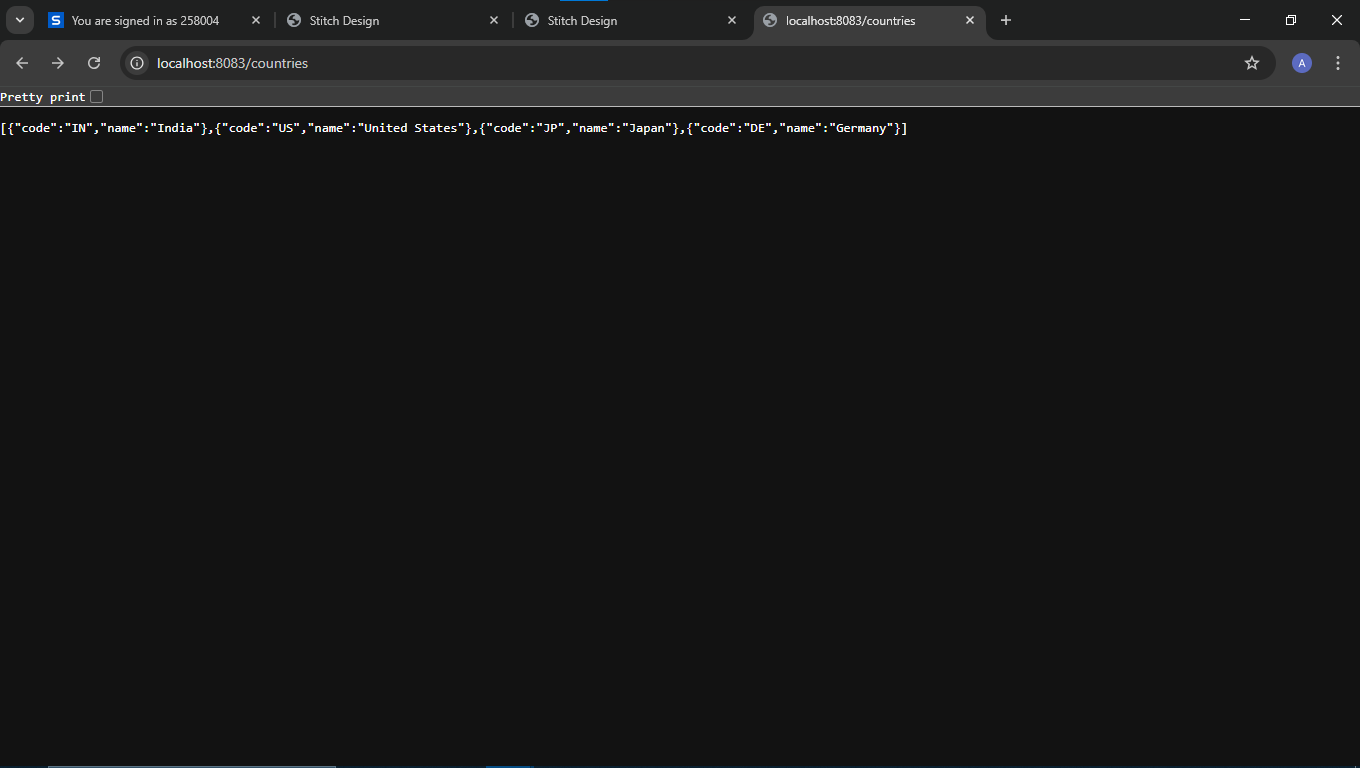
**public** **void** setName(String name) {

**this**.name = name;

}

}

Web screenshot:

****

**Create authentication service that returns JWT**   
  
As part of first step of JWT process, the user credentials needs to be sent to authentication service request that generates and returns the JWT.  
  
Ideally when the below curl command is executed that calls the new authentication service, the token should be responded. Kindly note that the credentials are passed using -u option.

Pom.xml:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>spring-jwt-demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-jwt-demo</name>

<description>Spring Boot JWT Authentication Demo</description>

<properties>

<java.version>17</java.version>

<spring.boot.version>3.2.4</spring.boot.version>

</properties>

<dependencies>

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.3.1</version>

</dependency>

<dependency>

<groupId>org.glassfish.jaxb</groupId>

<artifactId>jaxb-runtime</artifactId>

<version>2.3.1</version>

</dependency>

<!-- Spring Boot Web -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>${spring.boot.version}</version>

</dependency>

<!-- Spring Boot Security -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

<version>${spring.boot.version}</version>

</dependency>

<!-- JWT Token Library -->

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.0</version>

</dependency>

<!-- Jakarta Servlet API (required for Spring Boot 3+ filters) -->

<dependency>

<groupId>jakarta.servlet</groupId>

<artifactId>jakarta.servlet-api</artifactId>

<version>6.0.0</version>

<scope>provided</scope>

</dependency>

<!-- Optional: Lombok (if you want cleaner model code) -->

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.30</version>

<scope>provided</scope>

</dependency>

<!-- Spring Boot Test -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<version>${spring.boot.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>2.17.1</version> <!-- Latest as of mid-2025 -->

</dependency>

</dependencies>

<build>

<plugins>

<!-- Spring Boot Maven Plugin -->

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<version>${spring.boot.version}</version>

</plugin>

</plugins>

</build>

</project>

Application.properties:

server.port=8090

SecurityConfig.java:

**package** com.example.security;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.security.authentication.AuthenticationManager;

**import** org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

**import** org.springframework.security.config.annotation.web.builders.HttpSecurity;

**import** org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

**import** org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

**import** org.springframework.security.crypto.password.PasswordEncoder;

**import** org.springframework.security.web.SecurityFilterChain;

**import** org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfiguration;

**import** com.example.security.JwtAuthorizationFilter;

@Configuration

@EnableWebSecurity

**public** **class** SecurityConfig {

@Bean

**public** AuthenticationManager authenticationManager(AuthenticationConfiguration authConfig) **throws** Exception {

**return** authConfig.getAuthenticationManager();

}

@Bean

**public** SecurityFilterChain filterChain(HttpSecurity http) **throws** Exception {

http.csrf().disable()

.httpBasic()

.and()

.authorizeHttpRequests()

.requestMatchers("/authenticate").hasAnyRole("USER", "ADMIN")

.anyRequest().authenticated()

.and()

.addFilter(**new** JwtAuthorizationFilter(authenticationManager(http.getSharedObject(AuthenticationConfiguration.**class**))));

**return** http.build();

}

@Bean

**public** PasswordEncoder passwordEncoder() {

**return** **new** BCryptPasswordEncoder();

}

@Bean

**public** org.springframework.security.core.userdetails.UserDetailsService users() {

**return** username -> {

**if** (username.equals("user")) {

**return** org.springframework.security.core.userdetails.User.withUsername("user")

.password(passwordEncoder().encode("pwd"))

.roles("USER").build();

} **else** **if** (username.equals("admin")) {

**return** org.springframework.security.core.userdetails.User.withUsername("admin")

.password(passwordEncoder().encode("pwd"))

.roles("ADMIN").build();

}

**throw** **new** RuntimeException("User not found");

};

}

}

AuthenticationController.java:

**package** com.example.controller;

**import** io.jsonwebtoken.Jwts;

**import** io.jsonwebtoken.SignatureAlgorithm;

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

**import** java.util.\*;

**import** java.util.Base64;

@RestController

**public** **class** AunthenticationController {

@GetMapping("/authenticate")

**public** Map<String, String> authenticate(@RequestHeader("Authorization") String authHeader) {

Map<String, String> map = **new** HashMap<>();

String user = getUser(authHeader);

String token = generateJwt(user);

map.put("token", token);

**return** map;

}

**private** String getUser(String authHeader) {

String encoded = authHeader.substring("Basic ".length());

**byte**[] decodedBytes = Base64.*getDecoder*().decode(encoded);

String decoded = **new** String(decodedBytes);

**return** decoded.split(":")[0];

}

**private** String generateJwt(String user) {

**return** Jwts.builder()

.setSubject(user)

.setIssuedAt(**new** Date())

.setExpiration(**new** Date(System.*currentTimeMillis*() + 1200000))

.signWith(SignatureAlgorithm.HS256, "secretkey")

.compact();

}

}

Application.java:

**package** com.example;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**public** **class** Application {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(Application.**class**, args);

}

}

Web Screenshot:

