**Week – 04**

**Spring REST using Spring Boot 3 - spring-rest-handson**

**1. Create a Spring Web Project using Maven**

**Explanation:**

* **Project Setup:** Used Spring Initializer to generate a Spring Boot project with `Spring Web` and `DevTools` dependencies.
* **Build with Maven:** Ran `mvn clean package` to build the project; `pom.xml` handles dependencies and plugins.
* **Import in Eclipse:** Imported as "Existing Maven Project" into Eclipse; IDE auto-resolves dependencies.
* **Main Class:** `SpringLearnApplication.java` is the entry point with `@SpringBootApplication`. Added print/log to verify launch.
* **Project Structure:**
  + `src/main/java`: app code
  + `resources`: configs
  + `test/java`: tests
  + `pom.xml`: project config

**Code:**

**Dependencies in pom.xml:**

<?xml version="1.0" encoding="UTF-8"?>

<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

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

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

<version>3.5.3</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learndemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>spring-learndemo</name>

<description>Spring Boot Web App</description>

<url/>

<licenses>

<license/>

</licenses>

<developers>

<developer/>

</developers>

<scm>

<connection/>

<developerConnection/>

<tag/>

<url/>

</scm>

<properties>

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

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**SpringLearndemoApplication.java**

**(src/main/java -> com.cognizant.spring\_learndemo -> SpringLearndemoApplication.java)**

package com.cognizant.spring\_learndemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class SpringLearndemoApplication {

public static void main(String[] args) {

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

System.***out***.println("SpringLearnApplication started");

}

}

**application.properties**

**(src/main/resources -> application.properties)**

spring.application.name=spring-learndemo

server.port=8081

**SpringLearndemoApplicationTests**

**(src/test/java -> com.cognizant.spring\_learndemo -> SpringLearndemoApplicationTests.java)**

package com.cognizant.spring\_learndemo;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

*@SpringBootTest*

class SpringLearndemoApplicationTests {

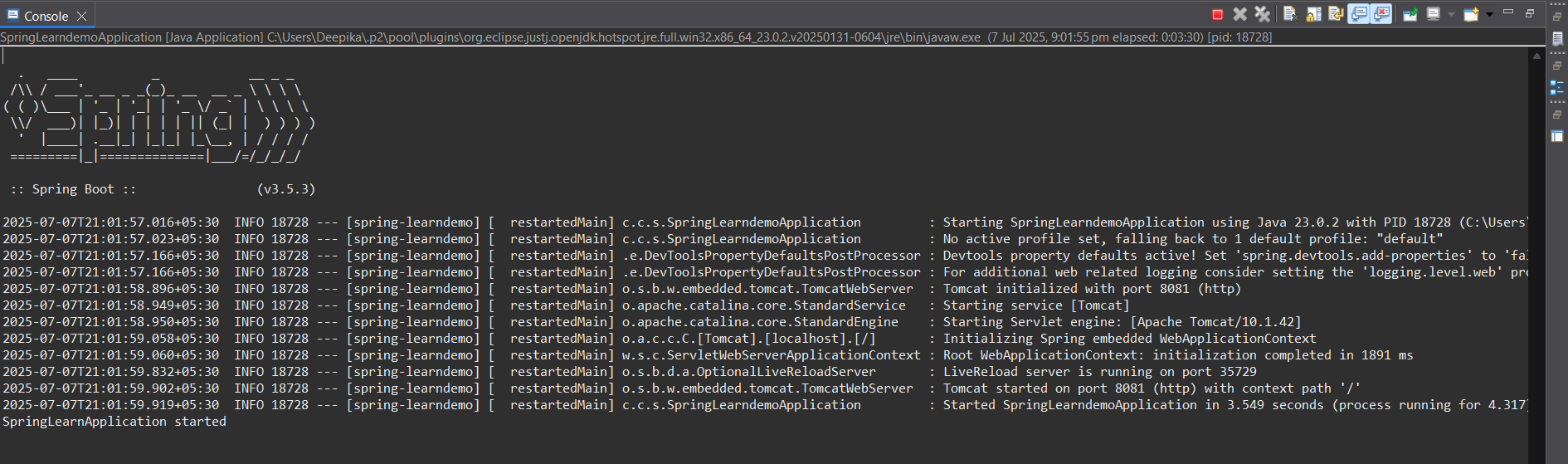
*@Test*

void contextLoads() {

}

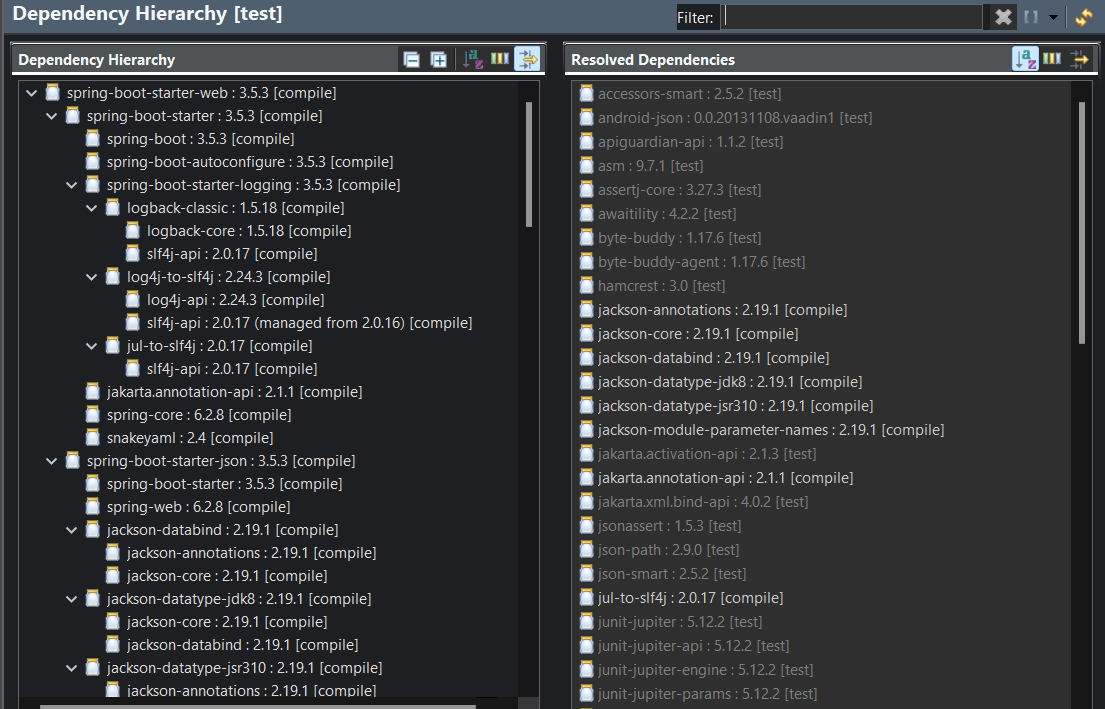
}

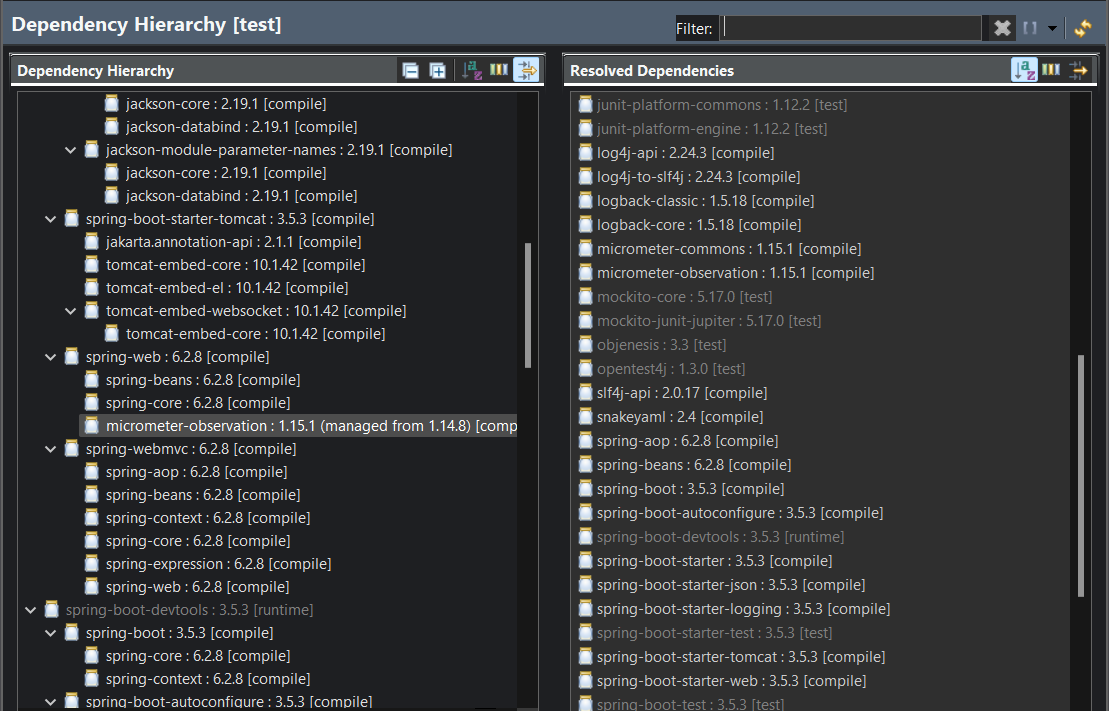
**Output:**

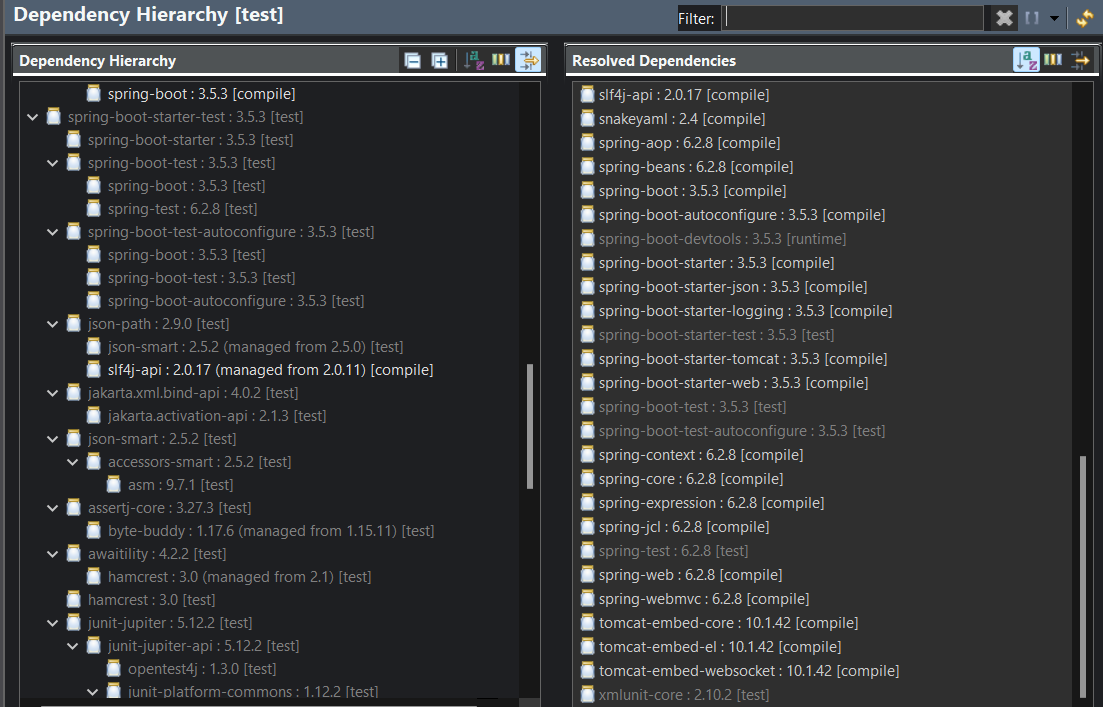


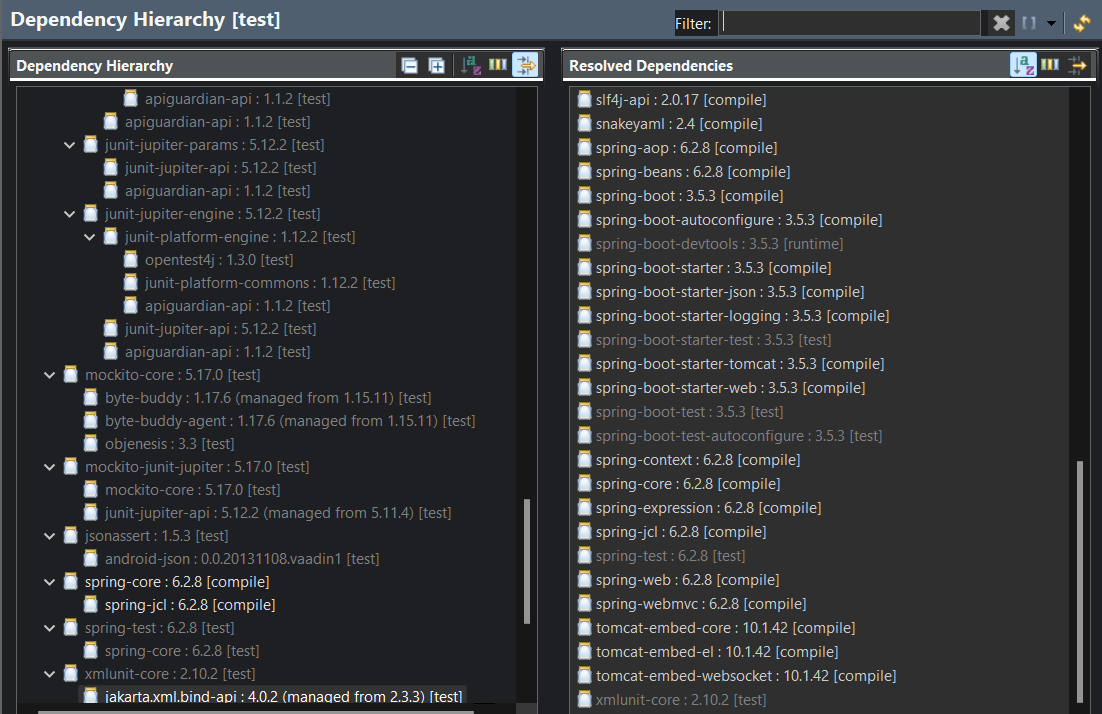
**SME Walkthrough:**

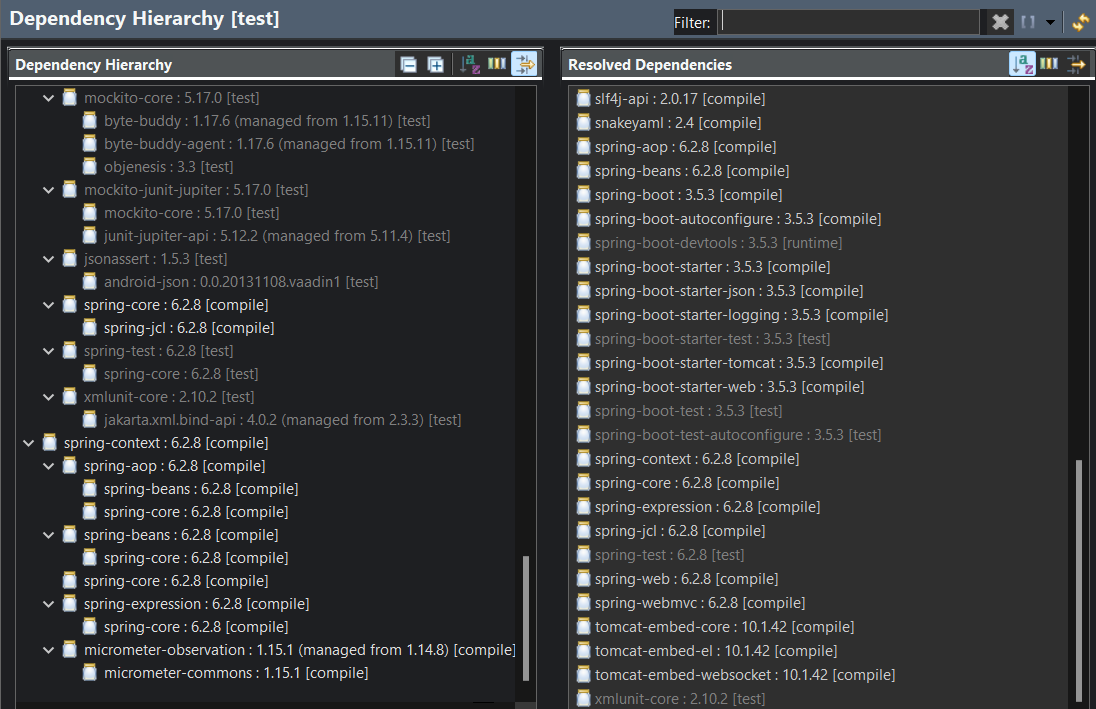
* src/main/java - Holds all your **application logic**.
* src/main/resources - For **application.properties**, application.yml, and static content.
* src/test/java - Contains **unit and integration test code**.
* SpringLearnApplication.java - This is the **entry point** of the Spring Boot app:
* Annotated with @SpringBootApplication which:
  + Enables component scanning
  + Configures Spring Boot auto-configuration
  + Sets up Spring application context
* pom.xml - Defines project configuration and dependencies.











**2. Spring Core – Load Country from Spring Configuration XML**

**Explanation:**

* Created a Spring XML (country.xml) to define a Country bean with code and name properties.
* Developed a Country class with constructor, getters, setters, toString(), and debug logs.
* Used ClassPathXmlApplicationContext in SpringLearnApplication to load and access the bean.
* Configured logback.xml to show only the log message (suppressing timestamps and thread info).
* Verified that Spring successfully created the bean and injected values, as shown in the debug logs.

**Code:**

**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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant.springlearn.loadcountry.demo</groupId>

<artifactId>spring-learn\_loadcountry\_demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- Spring Core and Context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.33</version>

</dependency>

<!-- SLF4J API -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.36</version>

</dependency>

<!-- Logback for logging -->

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>1.2.11</version>

</dependency>

</dependencies>

<build>

<resources>

<resource>

<directory>src/main/resources</directory>

</resource>

</resources>

</build>

</project>

**Country,java:**

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

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

private String code;

private String name;

public Country() {

***LOGGER***.debug("Inside Country Constructor.");

}

public String getCode() {

***LOGGER***.debug("Inside getCode");

return code;

}

public void setCode(String code) {

***LOGGER***.debug("Inside setCode");

this.code = code;

}

public String getName() {

***LOGGER***.debug("Inside getName");

return name;

}

public void setName(String name) {

***LOGGER***.debug("Inside setName");

this.name = name;

}

*@Override*

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

**SpringLearnApplication.java:**

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SpringLearnApplication {

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

public static void main(String[] args) {

*displayCountry*();

}

public static void displayCountry() {

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

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

***LOGGER***.debug("Country : {}", country.toString());

}

}

**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.cognizant.springlearn.Country">

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

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

</bean>

</beans>

**logback.xml:**

<configuration>

<appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">

<encoder>

<pattern>%msg%n</pattern>

</encoder>

</appender>

<root level="DEBUG">

<appender-ref ref="STDOUT"/>

</root>

</configuration>

**Output:**

**SME walkthrough:**

* **<bean> tag -** Defines a Spring-managed object.
* **id attribute** - Unique identifier of the bean.
* **class attribute** - Fully qualified class name of the bean.
* **<property> tag** - Sets the value of a bean property via setter injection.
* **name attribute** - The property name to inject (should match setter).
* **value attribute** - The value to inject into the property.
* **ApplicationContext -** Spring container that loads bean definitions and manages lifecycle.
* **ClassPathXmlApplicationContext** - Loads the Spring XML file from the classpath.
* **context.getBean("country", Country.class)**
* Loads bean with id country
* Converts it to Country class instance
* Spring internally:
  + Instantiates Country
  + Calls setter methods with values from XML

**3. Hello World RESTful Web Service**

**Explanation:**

* Initialized a Spring Boot project with group ID com.cognizant.springlearn.hellodemo and artifact ID helloworlddemo.
* Structured packages and placed the main class and controller under appropriate folders.
* Created a REST controller with a /hello GET endpoint returning "Hello World!!".
* Added SLF4J logs to indicate method start and end.
* Configured port 8083 and tested the service via browser and Postman.

**Code:**

**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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant.springlearn.hellodemo</groupId>

<artifactId>helloworlddemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>helloworlddemo</name>

<parent>

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

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

<version>2.7.18</version>

<relativePath/>

</parent>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

</dependency>

<dependency>

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

<artifactId>jackson-databind</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**SpringLearnApplication.java**

package com.cognizant.springlearn.hellodemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class SpringLearnApplication {

public static void main(String[] args) {

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

}

}

**HelloController.java**

package com.cognizant.springlearn.hellodemo.controller;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

*@RestController*

public class HelloController {

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

*@GetMapping*("/hello")

public String sayHello() {

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

String response = "Hello World!!";

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

return response;

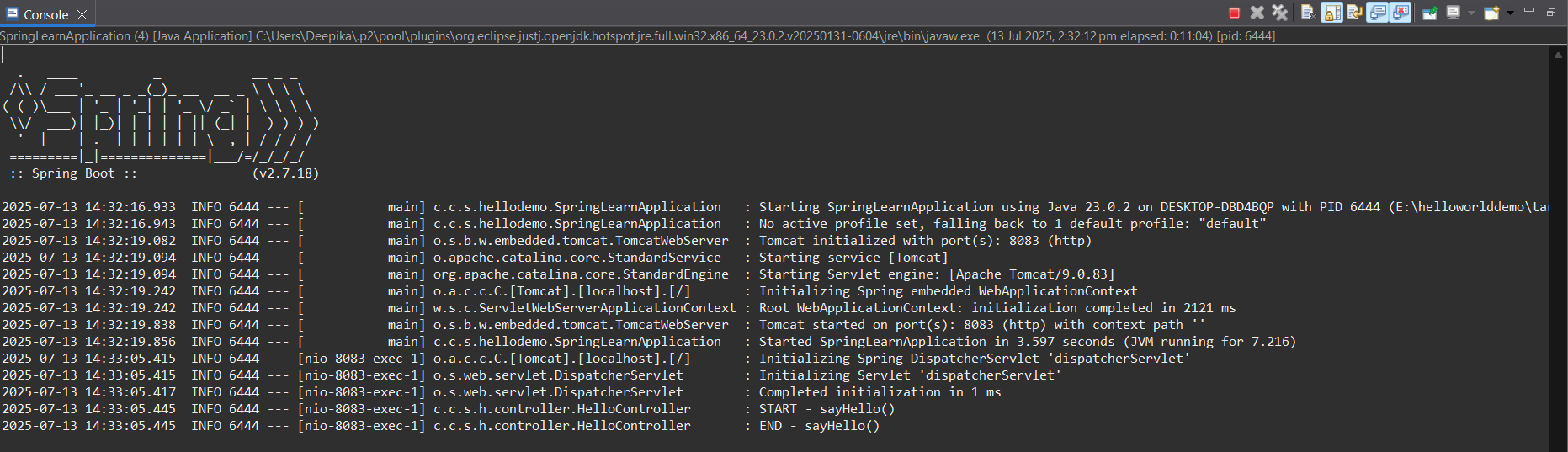
}

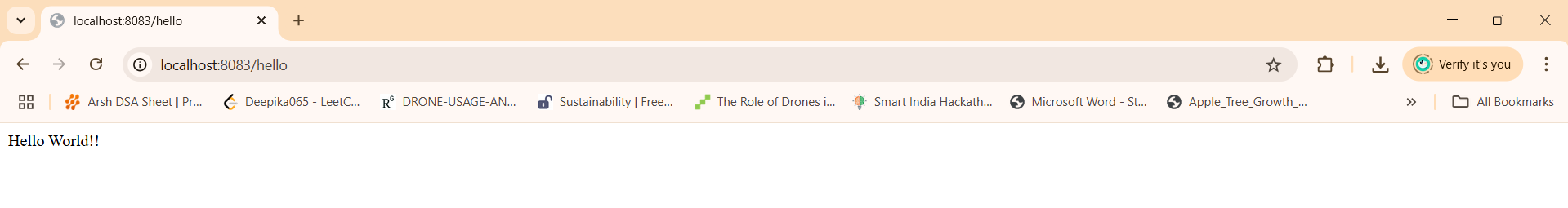
}

**Application.properties**

server.port=8083

**Output:**

****

****

**4. REST - Country Web Service**

**Explanation:**

* **Created a Country POJO -** Defined a simple Java class with code and name fields, along with getters and setters.
* **Configured the Bean in country.xml -** Created an XML file in src/main/resources to define the Country bean with ID in and assigned values for India.
* **Built a REST Controller -** Developed CountryController with a /country endpoint that loads the Spring context, fetches the Country bean, and returns it.
* **Enabled Automatic JSON Conversion -** Used @RestController so that Spring Boot automatically converts the returned bean into a JSON response using Jackson.
* **Tested the Endpoint -** Verified the output at http://localhost:8083/country via browser and Postman, and examined response headers in both tools.

**Code:**

**SpringLearnApplication.java**

package com.cognizant.springlearn.hellodemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class SpringLearnApplication {

public static void main(String[] args) {

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

}

}

**CountryController.java**

package com.cognizant.springlearn.hellodemo.controller;

import com.cognizant.springlearn.hellodemo.model.Country;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

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

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

*@RestController*

public class CountryController {

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

*@RequestMapping*("/country")

public Country getCountryIndia() {

***LOGGER***.info("START - getCountryIndia()");

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

Country india = (Country) context.getBean("in");

***LOGGER***.info("END - getCountryIndia()");

return india;

}

}

**HelloController.java**

package com.cognizant.springlearn.hellodemo.controller;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

*@RestController*

public class HelloController {

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

*@GetMapping*("/hello")

public String sayHello() {

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

String response = "Hello World!!";

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

return response;

}

}

**Country.java**

package com.cognizant.springlearn.hellodemo.model;

public class Country {

private String code;

private String name;

// Getters and Setters

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;

}

}

**application.properties**

server.port=8083

**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.cognizant.springlearn.hellodemo.model.Country">

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

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

</bean>

</beans>

**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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant.springlearn.hellodemo</groupId>

<artifactId>helloworlddemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>helloworlddemo</name>

<parent>

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

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

<version>2.7.18</version>

<relativePath/>

</parent>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

</dependency>

<dependency>

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

<artifactId>jackson-databind</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

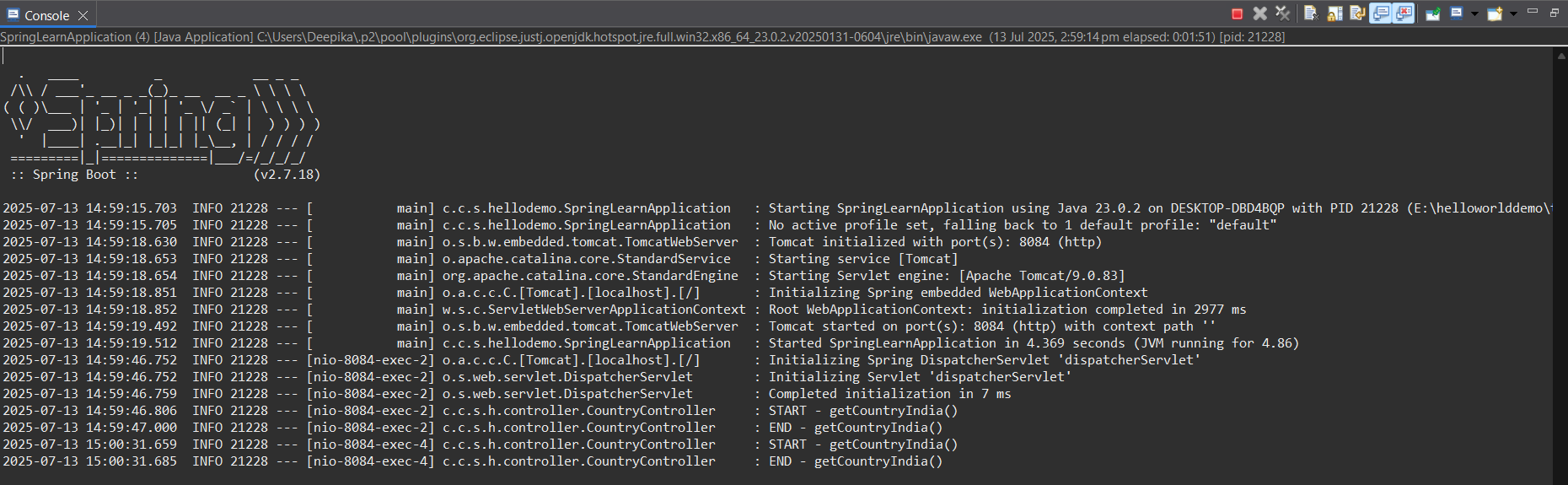
</plugin>

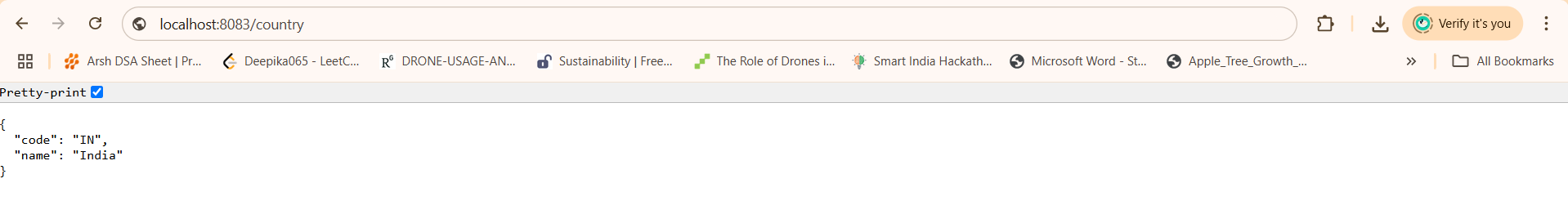
</plugins>

</build>

</project>

**Output:**

****

****

**SME Walkthrough:**

* **What happens in the controller? -** It loads country.xml, retrieves the in bean, and returns it.
* **How is the bean converted to JSON?** - Spring Boot auto-converts the object to JSON using Jackson via @RestController.
* **Network tab in DevTools -** Inspect /country request → Headers → View details like Content-Type, Status.
* **Postman Headers tab -** Send GET request to /country → Check Headers for Content-Type: application/json.

**5. REST - Get country based on country code**

**Explanation:**

* Added multiple Country beans and a list in country.xml.
* Created CountryService to fetch a country by code using a case-insensitive match.
* Used Java 8 Streams to search through the list of countries.
* Added a new REST endpoint /countries/{code} in CountryController.
* Tested using browser and Postman with various country codes.

**Code:**

**SpringLearnApplication.java**

package com.cognizant.springlearn.hellodemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class SpringLearnApplication {

public static void main(String[] args) {

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

}

}

**CountryController.java**

package com.cognizant.springlearn.hellodemo.controller;

import com.cognizant.springlearn.hellodemo.model.Country;

import com.cognizant.springlearn.hellodemo.service.CountryService;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

*@RestController*

public class CountryController {

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

*@Autowired*

private CountryService countryService;

*@RequestMapping*("/country")

public Country getCountryIndia() {

***LOGGER***.info("START - getCountryIndia()");

Country country = countryService.getCountry("IN");

***LOGGER***.info("END - getCountryIndia()");

return country;

}

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

public Country getCountry(*@PathVariable* String code) {

***LOGGER***.info("START - getCountry()");

Country country = countryService.getCountry(code);

***LOGGER***.info("END - getCountry()");

return country;

}

}

**Country.java**

package com.cognizant.springlearn.hellodemo.model;

public class Country {

private String code;

private String name;

// Getters and Setters

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;

}

}

**CountryService.java**

package com.cognizant.springlearn.hellodemo.service;

import com.cognizant.springlearn.hellodemo.model.Country;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

*@Service*

public class CountryService {

public Country getCountry(String code) {

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

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

return countries.stream()

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

.findFirst()

.orElse(null);

}

}

**application.properties**

server.port=8089

**country.xml**

<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.cognizant.springlearn.hellodemo.model.Country">

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

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

</bean>

<bean id="us" class="com.cognizant.springlearn.hellodemo.model.Country">

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

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

</bean>

<bean id="jp" class="com.cognizant.springlearn.hellodemo.model.Country">

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

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

</bean>

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

<constructor-arg>

<list>

<ref bean="in"/>

<ref bean="us"/>

<ref bean="jp"/>

</list>

</constructor-arg>

</bean>

</beans>

**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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant.springlearn.hellodemo</groupId>

<artifactId>helloworlddemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>helloworlddemo</name>

<parent>

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

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

<version>2.7.18</version>

<relativePath/>

</parent>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

</dependency>

<dependency>

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

<artifactId>jackson-databind</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

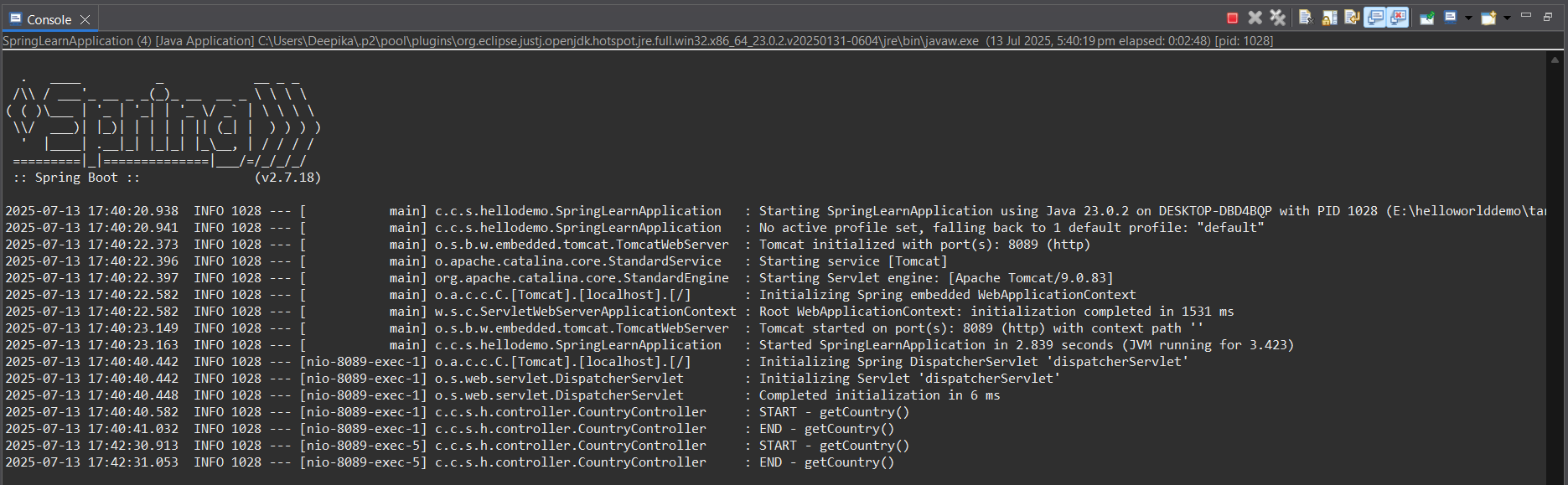
</plugin>

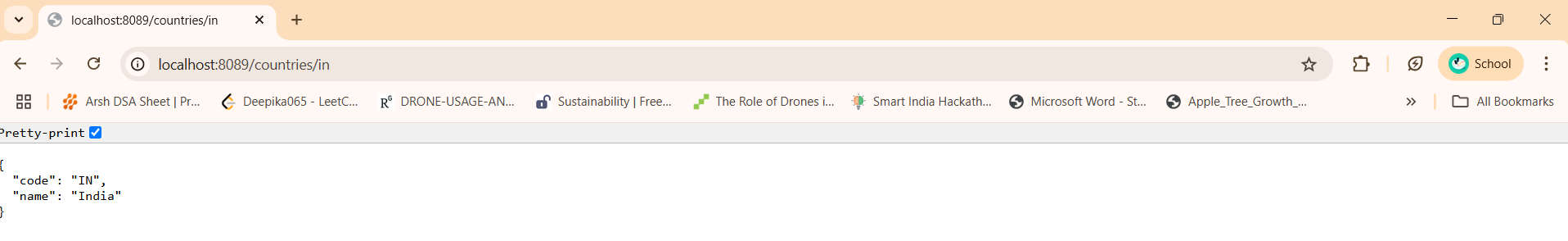
</plugins>

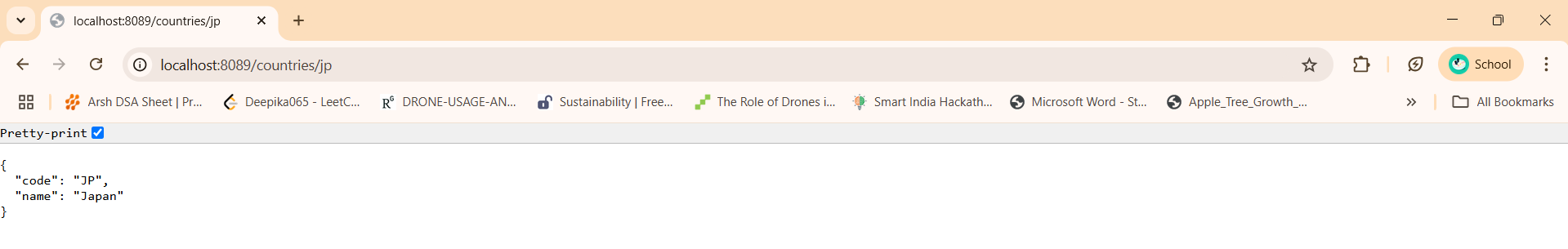
</build>

</project>

**Output**

****

****

****

**6. Create authentication service that returns JWT**

**Explanation:**

* **Created a REST Endpoint for Authentication -** Developed an AuthController with a GET API /authenticate that accepts username and password as query parameters.
* **Validated User Credentials -** Inside the controller, checked if the provided credentials matched a dummy user (user / pwd) for demonstration purposes.
* **Generated JWT Token -** Used a utility class JwtUtil to generate a JSON Web Token (JWT) when valid credentials were passed.
* **Returned Token in JSON Format -** Sent the generated JWT back in the response body as {"token":"<token>"}, using ResponseEntity.
* **Tested via Browser & Postman -** Successfully accessed the endpoint using a browser URL or Postman, like: http://localhost:8098/authenticate?username=user&password=pwd

**Code:**

**SpringLearnApplication.java**

package com.cognizant.springlearn.hellodemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class SpringLearnApplication {

public static void main(String[] args) {

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

}

}

**AuthController.java**

package com.cognizant.springlearn.hellodemo.controller;

import com.cognizant.springlearn.hellodemo.util.JwtUtil;

import org.springframework.http.ResponseEntity;

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

*@RestController*

public class AuthController {

*@GetMapping*("/authenticate")

public ResponseEntity<?> authenticate(

*@RequestParam* String username,

*@RequestParam* String password) {

// Dummy user check (replace with real authentication if needed)

if ("user".equals(username) && "pwd".equals(password)) {

String token = JwtUtil.*generateToken*(username);

return ResponseEntity.*ok*().body("{\"token\":\"" + token + "\"}");

} else {

return ResponseEntity.*status*(401).body("Invalid Credentials");

}

}

}

**SecurityConfig.java**

package com.cognizant.springlearn.hellodemo.config;

import org.springframework.context.annotation.Configuration;

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

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

import org.springframework.security.config.annotation.web.configuration.~~WebSecurityConfigurerAdapter~~;

*@Configuration*

*@EnableWebSecurity*

public class SecurityConfig extends ~~WebSecurityConfigurerAdapter~~ {

*@Override*

protected void configure(HttpSecurity http) throws Exception {

http

.csrf().disable()

.authorizeRequests()

.antMatchers("/authenticate").permitAll()

.anyRequest().authenticated()

.and()

.httpBasic();

}

}

**JwtUtil.java**

package com.cognizant.springlearn.hellodemo.util;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import io.jsonwebtoken.security.Keys;

import java.security.Key;

import java.util.Date;

public class JwtUtil {

private static final Key ***SECRET\_KEY*** = Keys.*secretKeyFor*(*SignatureAlgorithm*.***HS256***);

public static String generateToken(String username) {

long currentTimeMillis = System.*currentTimeMillis*();

return Jwts.*builder*()

.setSubject(username)

.setIssuedAt(new Date(currentTimeMillis))

.setExpiration(new Date(currentTimeMillis + 1000 \* 60 \* 60)) // 1 hour

.signWith(***SECRET\_KEY***)

.compact();

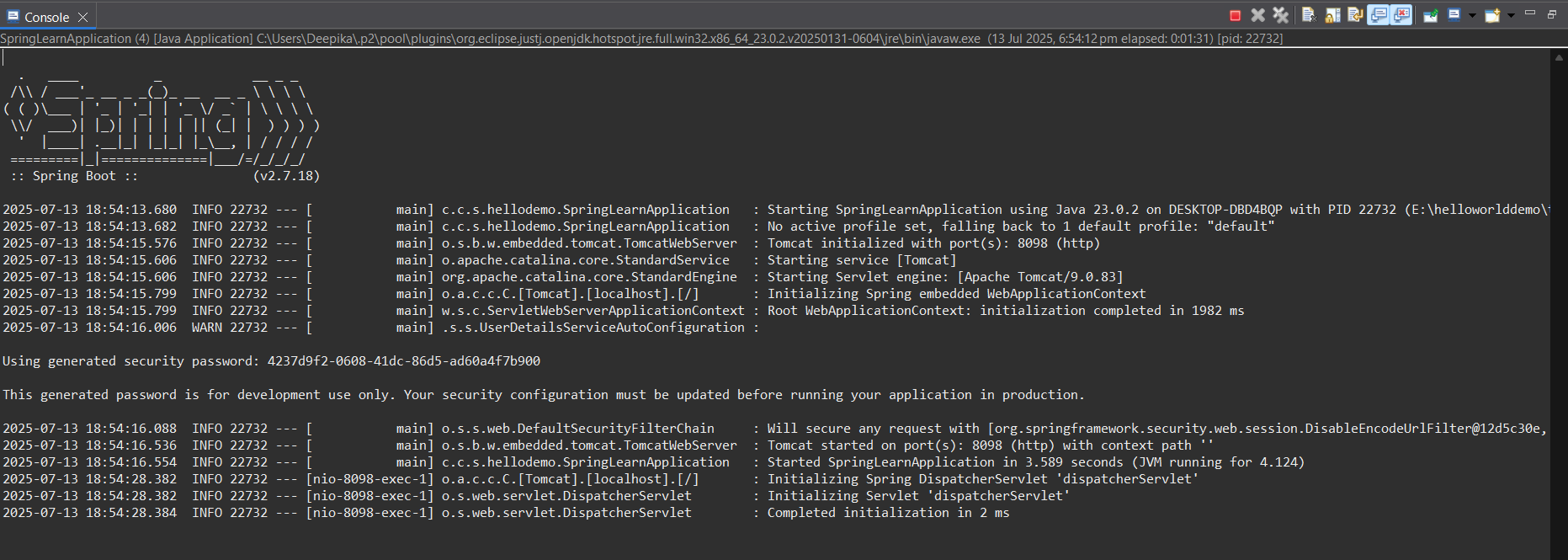
}

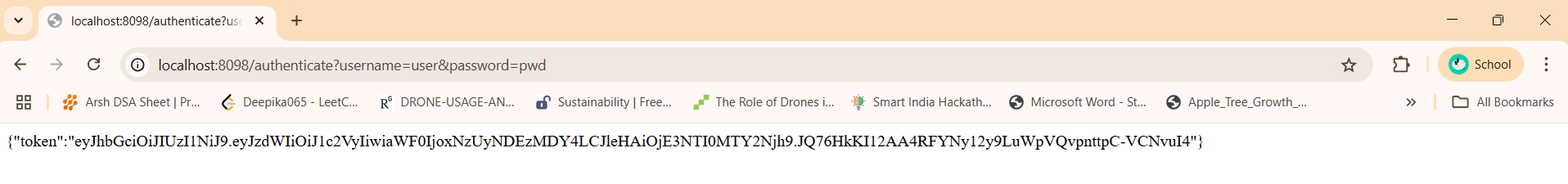
}

**application.properties**

server.port=8098

**Output**

****

****