**WEEK – 4**

**Spring REST using Spring Boot 3**

**1)** **Create a Spring Web Project using Maven:**

**CODE :**

**SpringLearnApplication.java :**

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

public static void main(String[] args) {

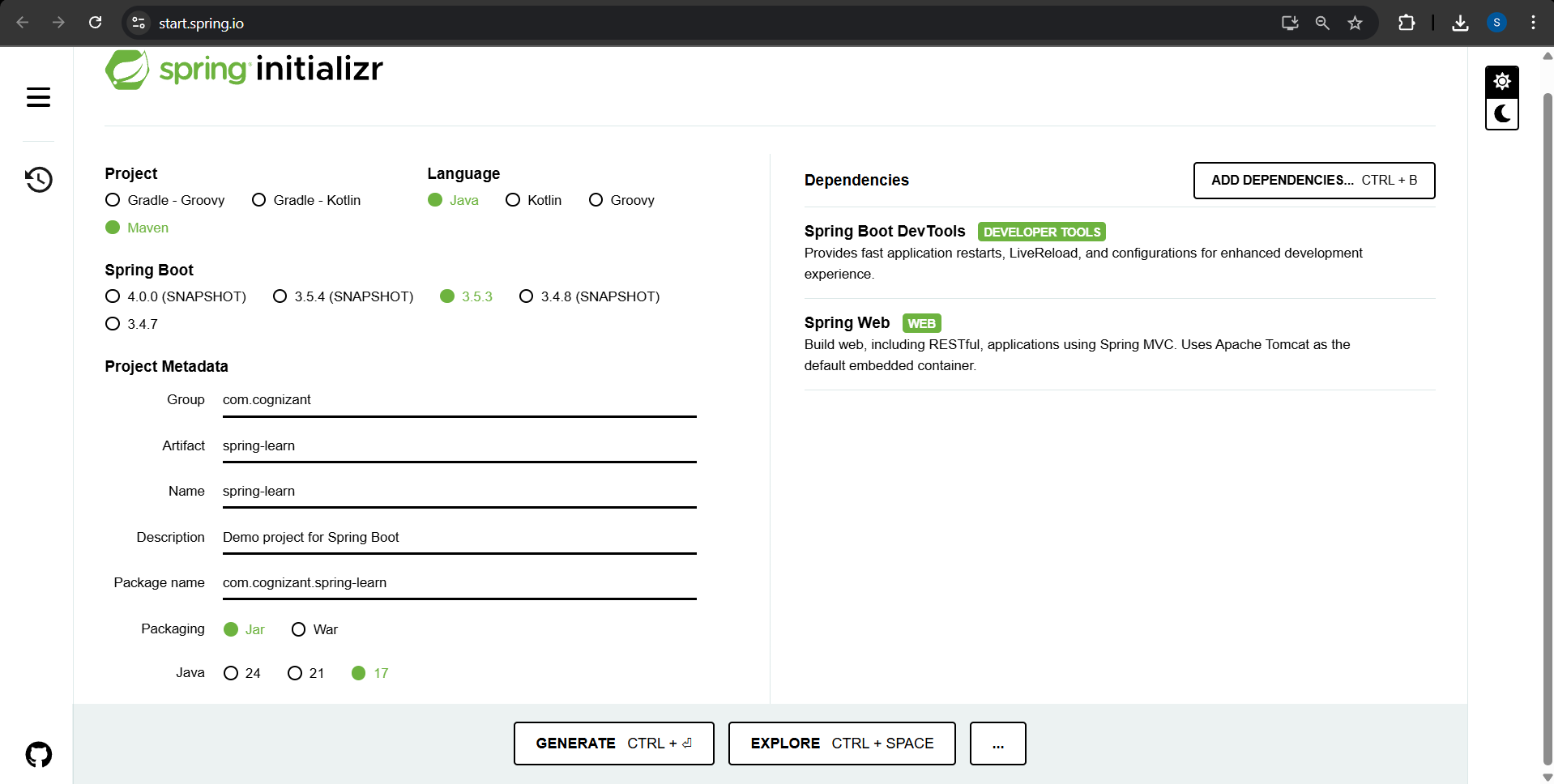
System.*out*.println("Application Started...");

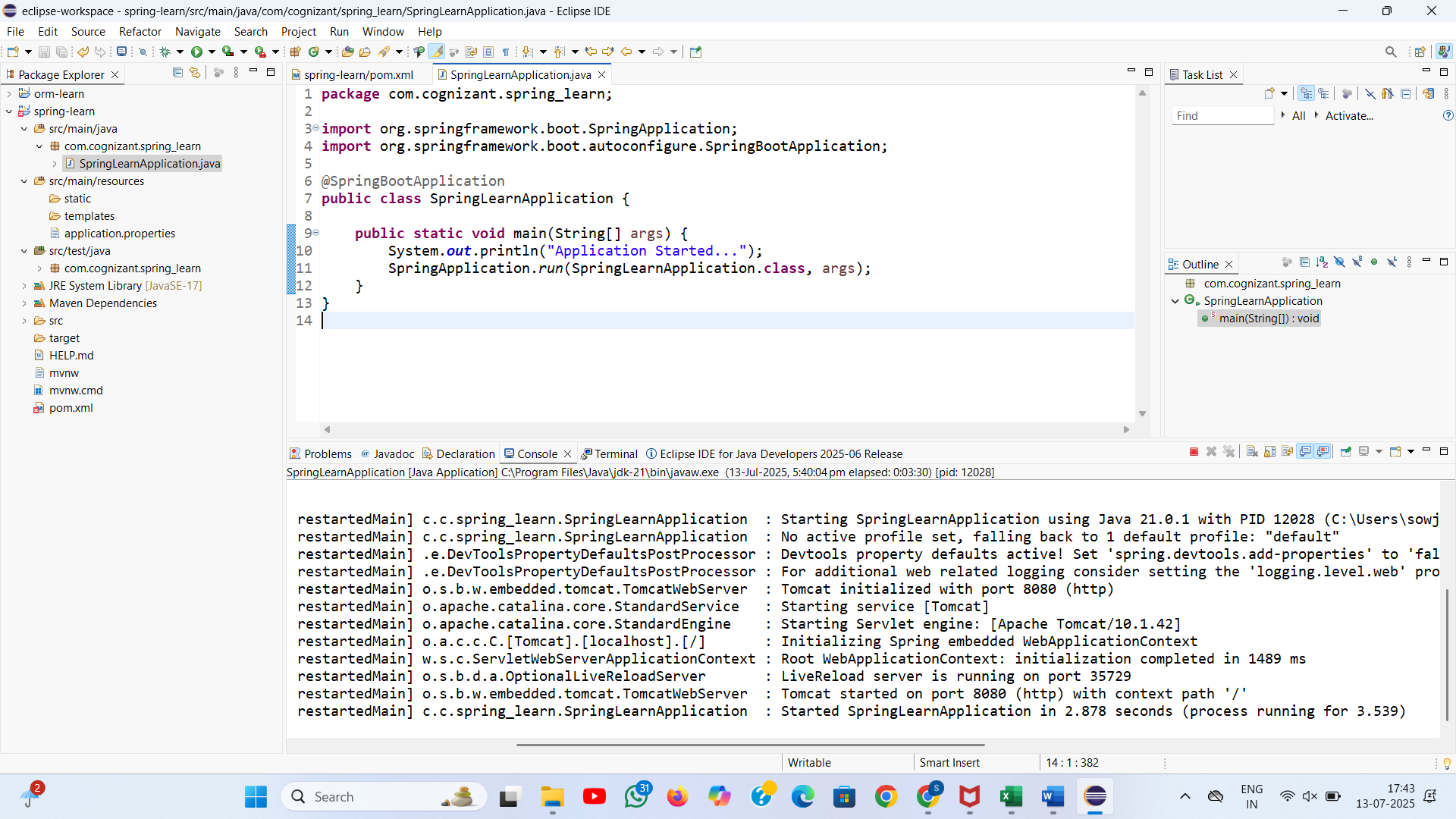
SpringApplication.*run*(SpringLearnApplication.class, args);

}

}

**OUTPUT:**





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

**CODE:**

**SpringLearnApplication Code:**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class SpringLearnApplication implements CommandLineRunner {

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

public static void main(String[] args) {

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

}

*@Override*

public void run(String... args) {

*LOGGER*.debug("START");

*displayCountry*();

*LOGGER*.debug("END");

}

public static void displayCountry() {

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

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

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

}

}

**Country.java :**

package com.cognizant.spring\_learn;

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("getCode() called");

return code;

}

public void setCode(String code) {

*LOGGER*.debug("setCode() called with: {}", code);

this.code = code;

}

public String getName() {

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

return name;

}

public void setName(String name) {

*LOGGER*.debug("setName() called with: {}", name);

this.name = name;

}

*@Override*

public String toString() {

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

}

}

**Application.properties :**

spring.application.name=spring-learn

logging.level.root=DEBUG

**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="country" class="com.cognizant.spring\_learn.Country">

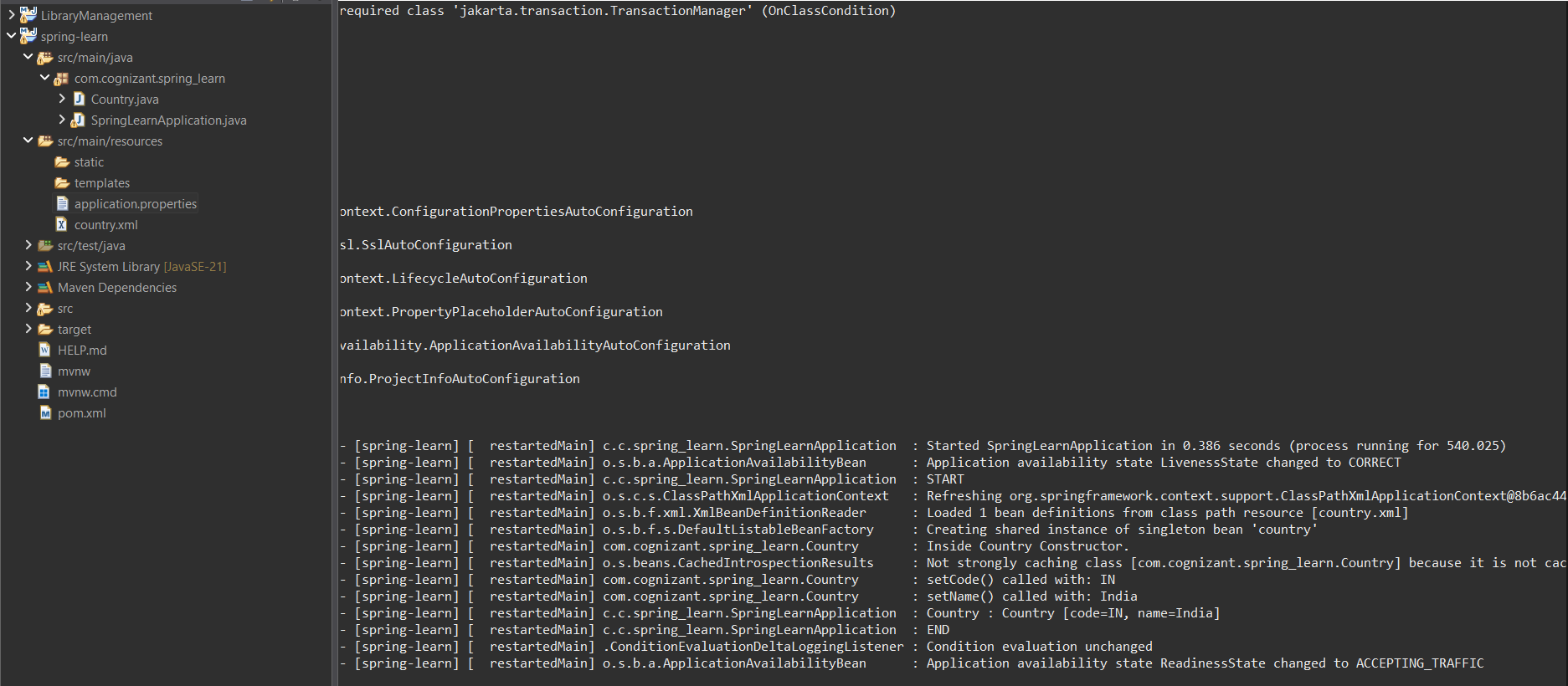
<property name="code" value="IN" />

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

</bean>

</beans>

**OUTPUT:**



**3)Hello World RESTful Web Service**

**STATEMENT:**

Method: GET  
URL: /hello  
Controller: com.cognizant.spring-learn.controller.HelloController  
Method Signature: public String sayHello()  
Method Implementation: return hard coded string "Hello World!!"  
Sample Request: http://localhost:8083/hello  
Sample Response: Hello World!!

**CODE:**

**HelloController.java:**

package com.cognizant.spring\_learn.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.debug("START: sayHello()");

String response = "Hello World!!";

LOGGER.debug("END: sayHello()");

return response;

}

}

**SME Explanation**

**Developer Tools (Chrome)**

1. Open Chrome → Visit http://localhost:8083/hello
2. Press F12 → Go to Network tab
3. Reload the page
4. Click on the /hello request
5. View these Response Headers:
   1. Content-Type: text/plain;charset=UTF-8
   2. Content-Length: 14
   3. Date: ...
   4. Connection: keep-alive

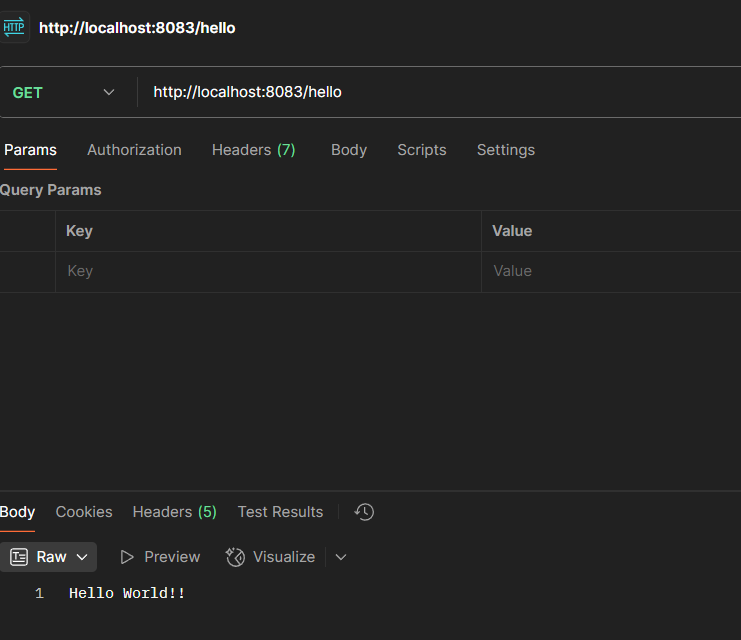
**Also view Request Headers:**

* GET /hello HTTP/1.1
* Host: localhost:8083
* User-Agent: Mozilla/...
* Accept: text/html,...

Postman → Headers Tab

1. After sending GET request to /hello, click on the Headers tab.
2. You’ll see similar headers:
   1. Response Headers:
      1. Content-Type: text/plain;charset=UTF-8
      2. Date, Connection, Content-Length, etc.
   2. Request Headers:
      1. User-Agent, Accept, Host

**OUTPUT:**





**4)** **REST - Country Web Service**

**STATEMENTS:**

**URL:** /country  
**Controller:** com.cognizant.spring-learn.controller.CountryController  
**Method Annotation:** @RequestMapping  
**Method Name:** getCountryIndia()  
**Method Implementation:** Load India bean from spring xml configuration and return  
**Sample Request**: http://localhost:8083/country  
**Sample Response**:

**Code:**

**CountryController.java**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.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.debug("START: getCountryIndia()");

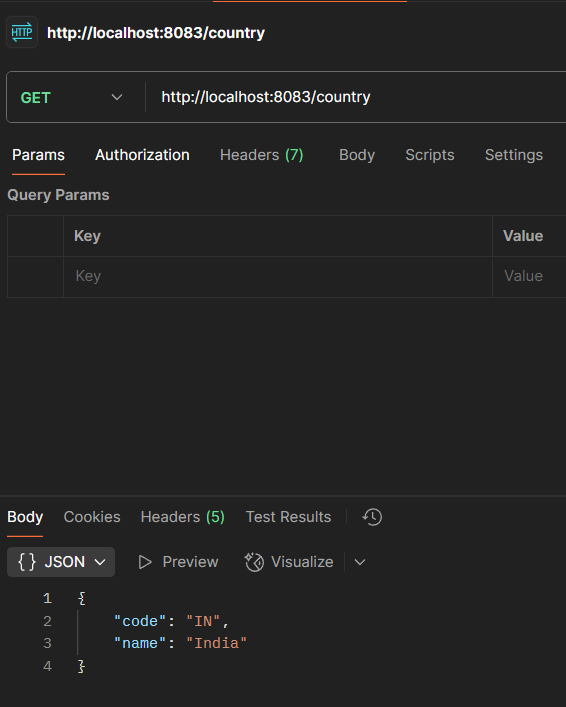
ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

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

LOGGER.debug("END: getCountryIndia()");

return country;

} }

**OUTPUT:**

**SME:**

**What happens in the controller method?**

@RequestMapping("/country")

public Country getCountryIndia() {

...

}

* This method handles HTTP GET requests to /country.
* It loads the Spring bean (Country) from country.xml using ApplicationContext.
* Returns a Java object, not a JSON string — Spring takes care of conversion.

**How is the bean converted to JSON?**

1. Spring Boot includes Jackson (a JSON processor) in its dependencies.
2. When a Java object is returned from a @RestController method, Spring uses Jackson to automatically:
   * Inspect the object's fields via getters
   * Convert it to a JSON string
   * Set Content-Type: application/json in the response headers

**In Chrome Developer Tools**

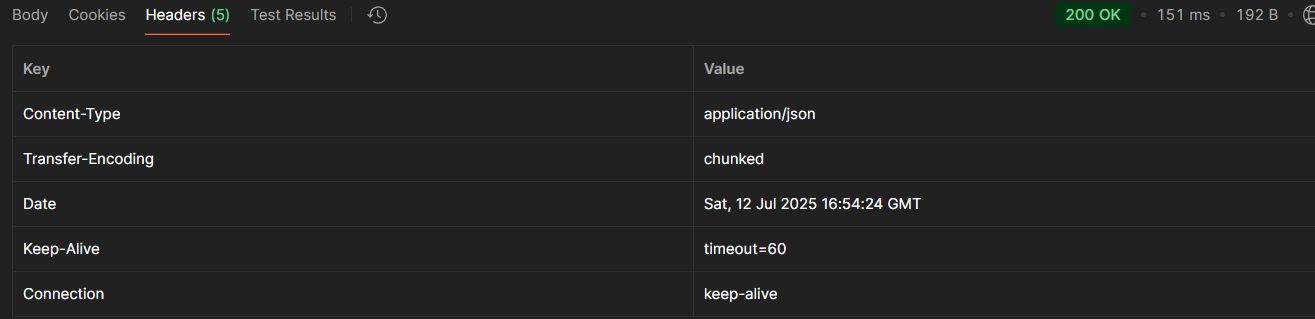
1. Visit http://localhost:8083/country
2. Press F12 → Go to Network tab → Reload
3. Click the /country request

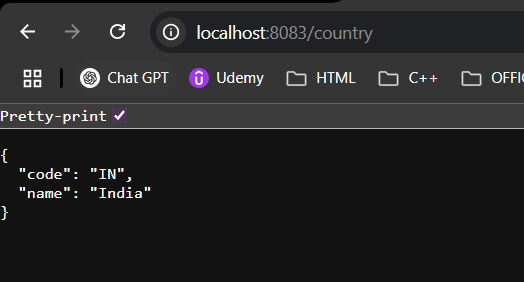
Request Headers:

* GET /country
* Host: localhost:8083
* Accept: application/json

**In Postman → Headers Tab**

1. Send a GET request to http://localhost:8083/country
2. Click the Headers tab





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

**CountryController.java:**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.model.Country;

import com.cognizant.spring\_learn.service.CountryService;

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

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

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

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

@RestController

public class CountryController {

@Autowired

private CountryService countryService;

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

public Country getCountry(@PathVariable String code) {

return countryService.getCountry(code);

}

}

**Country.java**

package com.cognizant.spring\_learn.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;

}

}

**CountryLookupApplication**

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class CountryLookupApplication {

public static void main(String[] args) {

SpringApplication.run(CountryLookupApplication.class, args);

}

}

**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="country1" class="com.cognizant.spring\_learn.model.Country">

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

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

</bean>

<bean id="country2" class="com.cognizant.spring\_learn.model.Country">

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

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

</bean>

<bean id="country3" class="com.cognizant.spring\_learn.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="country1"/>

<ref bean="country2"/>

<ref bean="country3"/>

</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.spring-learn</groupId>

<artifactId>country-lookup-service</artifactId>

<version>0.0.1-SNAPSHOT</version>

<description>Country Lookup REST Service</description>

<packaging>jar</packaging>

<parent>

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

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

<version>3.2.0</version>

<relativePath/>

</parent>

<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</groupId>

<artifactId>spring-context</artifactId>

</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>

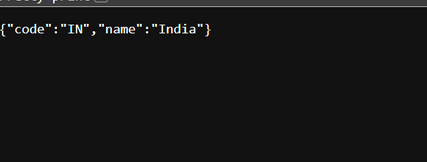
</plugin>

</plugins>

</build>

</project>

**Output :**

****

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

**Util:**package com.example.jwt;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class JwtAuthExampleApplication {

public static void main(String[] args) {

SpringApplication.run(JwtAuthExampleApplication.class, args);

}

}

**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>jwt-auth-example</artifactId>

<version>1.0-SNAPSHOT</version>

<packaging>jar</packaging>

<name>jwt-auth-example</name>

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

<parent>

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

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

<version>2.7.15</version>

<relativePath/>

</parent>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-api</artifactId>

<version>0.11.5</version>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-impl</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-jackson</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

