**REST - Country Web Service**

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

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

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

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

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

return country;

}

}

**SpringLearnApplication:**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

LOGGER.info("START");

displayCountry();

LOGGER.info("END");

}

public static void displayCountry() {

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

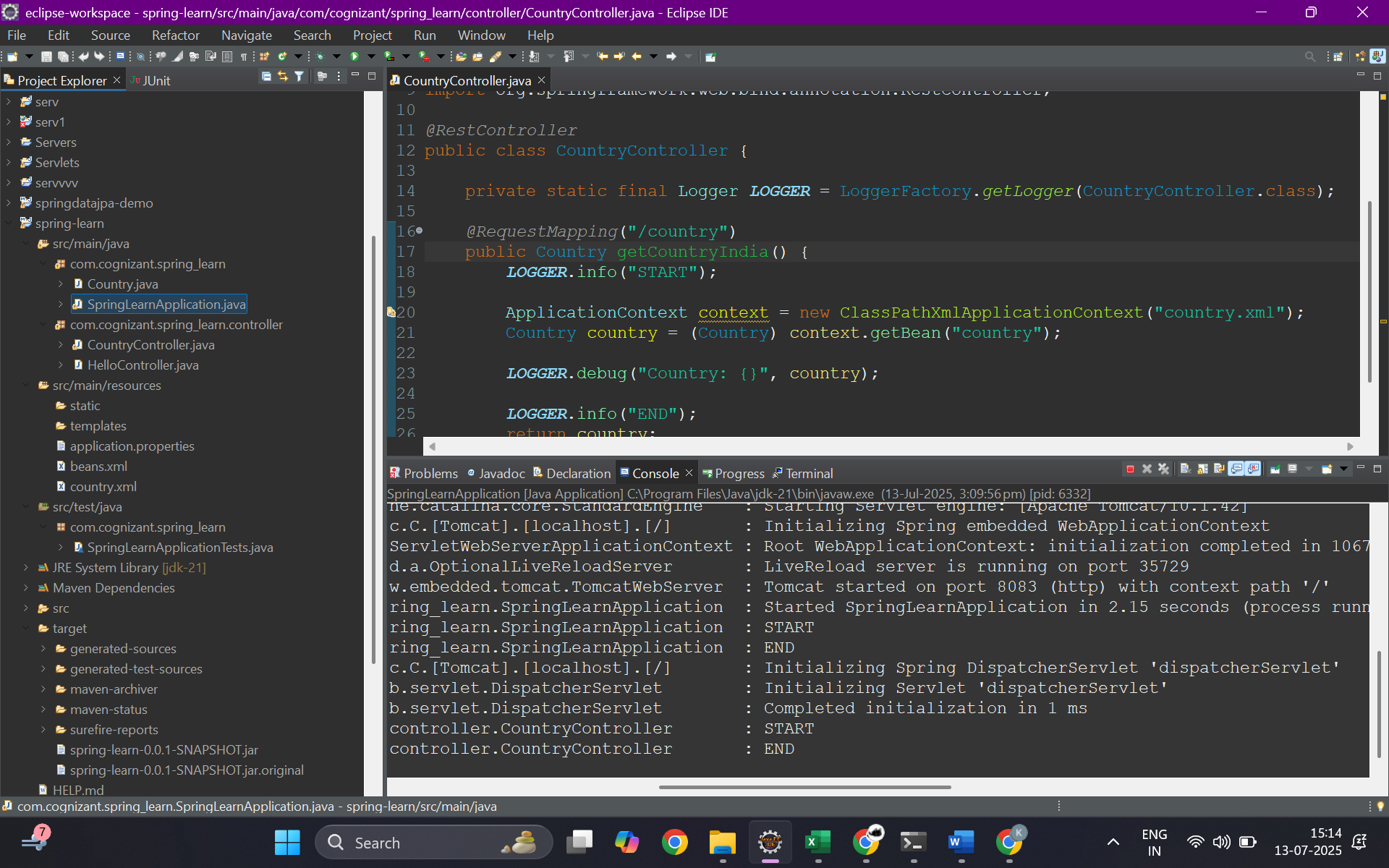
Country country = (Country) context.getBean("country", Country.class);

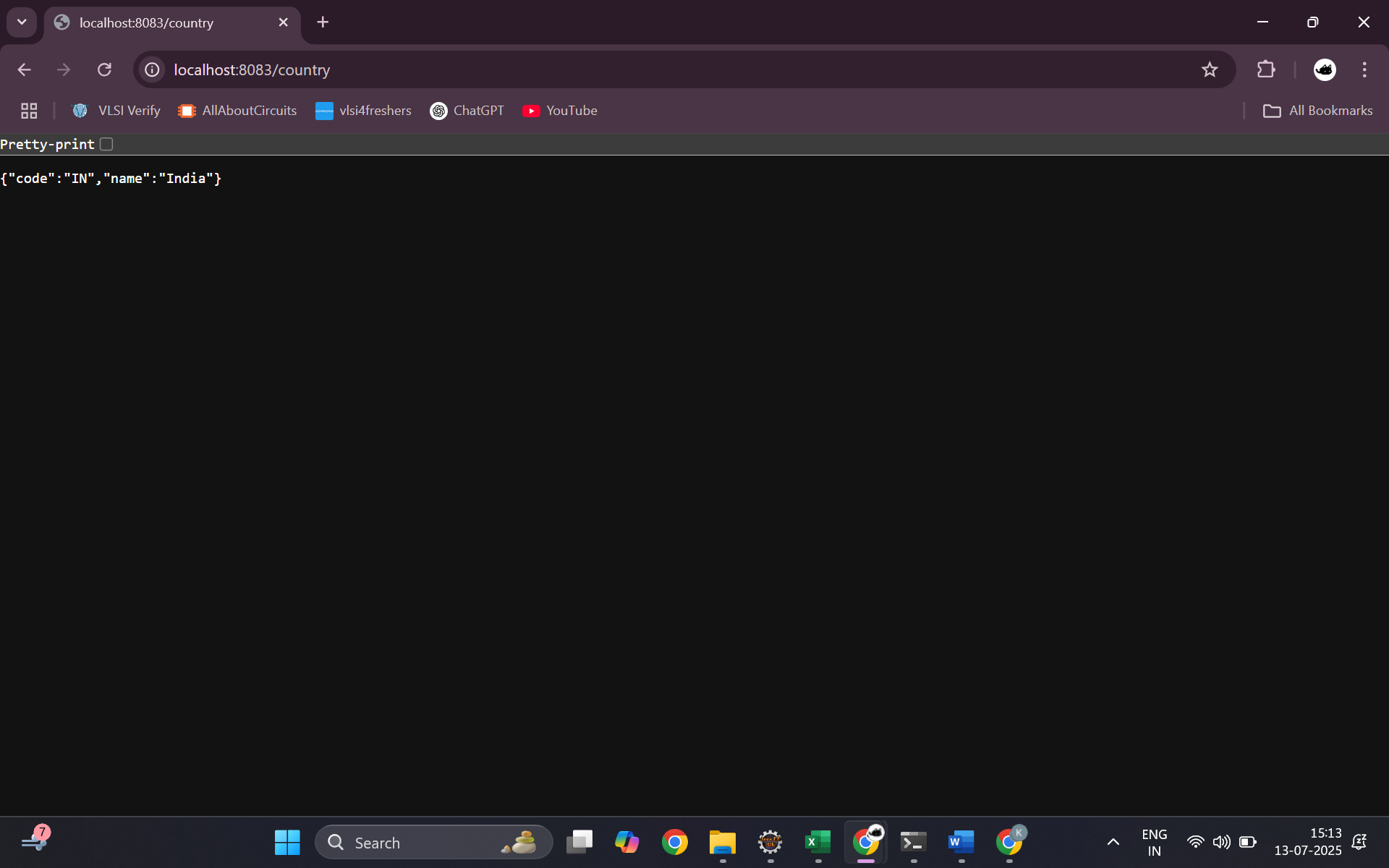
LOGGER.debug("Country: {}", country.toString());

}

}

**Output:**





**SME:**

In the CountryController, the method getCountryIndia() is annotated with @RequestMapping("/country"), which means it handles HTTP GET requests to /country. When this endpoint is hit, the controller uses Spring's ClassPathXmlApplicationContext to load the country.xml file and retrieve the bean with ID country, which holds details of the India object. This bean is returned directly by the method. Spring Boot, through the use of the @RestController annotation and the built-in **Jackson library**, automatically converts the returned Country object into a JSON response. This is part of Spring Boot's message conversion mechanism, where Java objects are serialized into JSON using MappingJackson2HttpMessageConverter. When the request is tested using the browser or Postman, the **network tab in the browser's developer tools** or the **Headers tab in Postman** will show HTTP header details such as Content-Type: application/json, response status 200 OK, and other headers like Date, Content-Length, and Server. These headers confirm that a RESTful response was served in JSON format and can also include CORS headers depending on configuration. The entire flow demonstrates how a Spring Boot application retrieves a configured bean and serves it as a RESTful JSON response using standard HTTP protocol.