WEEK 4

SPRING REST using Spring Boot 3

**Hands on 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) {

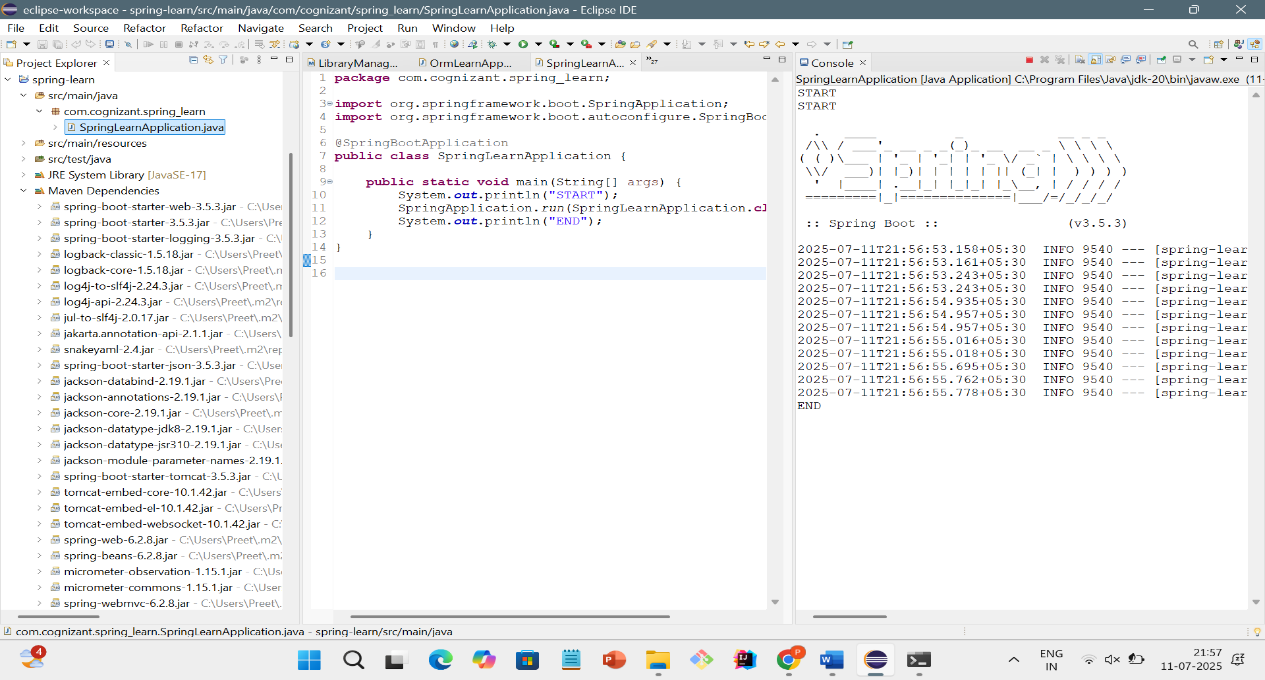
System.***out***.println("START");

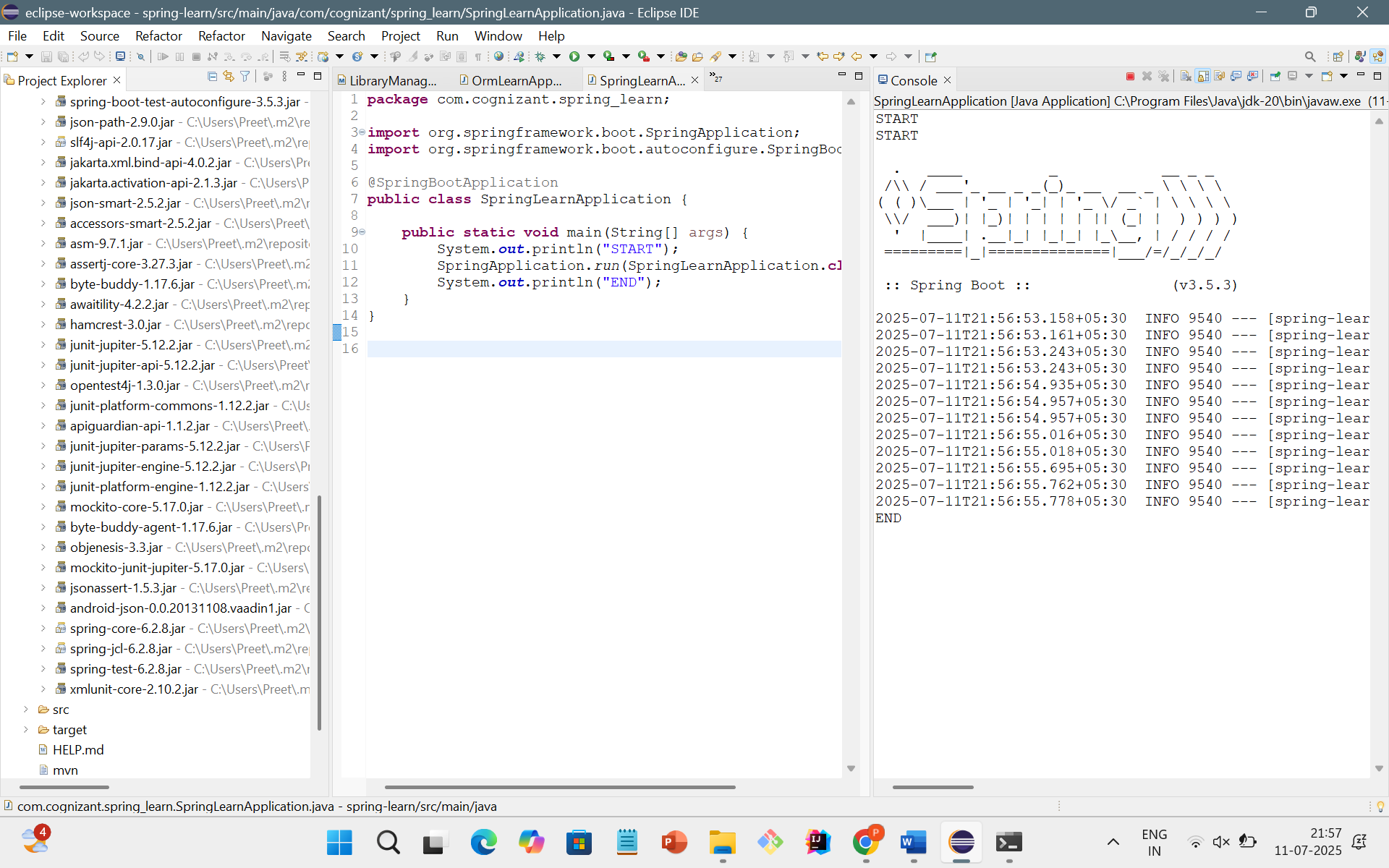
SpringApplication.*run*(SpringLearnApplication.**class**, args);

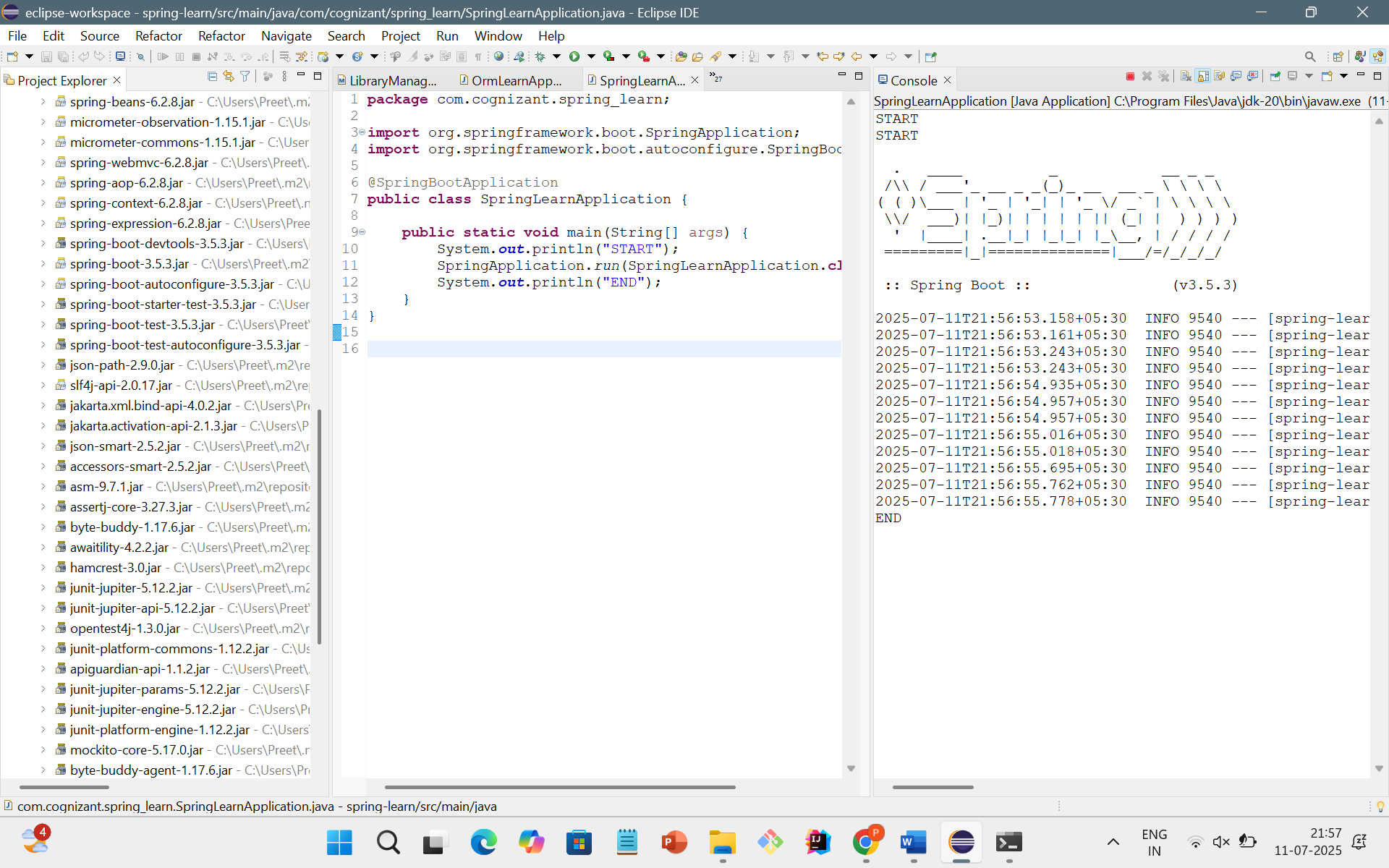
System.***out***.println("END");

}

}

OUTPUT: 





**Hands on 4**

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

CODE:

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>

Country.java

**package** com.cognizant.spring\_learn;

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

**private** String code;

**private** String name;

**public** Country() {

System.***out***.println("Inside Country Constructor.");

}

**public** String getCode() {

System.***out***.println("Inside getCode()");

**return** code;

}

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

System.***out***.println("Inside setCode()");

**this**.code = code;

}

**public** String getName() {

System.***out***.println("Inside getName()");

**return** name;

}

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

System.***out***.println("Inside setName()");

**this**.name = name;

}

@Override

**public** String toString() {

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

}

}

SpringlearnApplication.java

**package** com.cognizant.spring\_learn;

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

**public** **class** SpringLearnApplication {

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

System.***out***.println("START");

*displayCountry*();

System.***out***.println("END");

}

**public** **static** **void** displayCountry() {

**try** (ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext("country.xml")) {

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

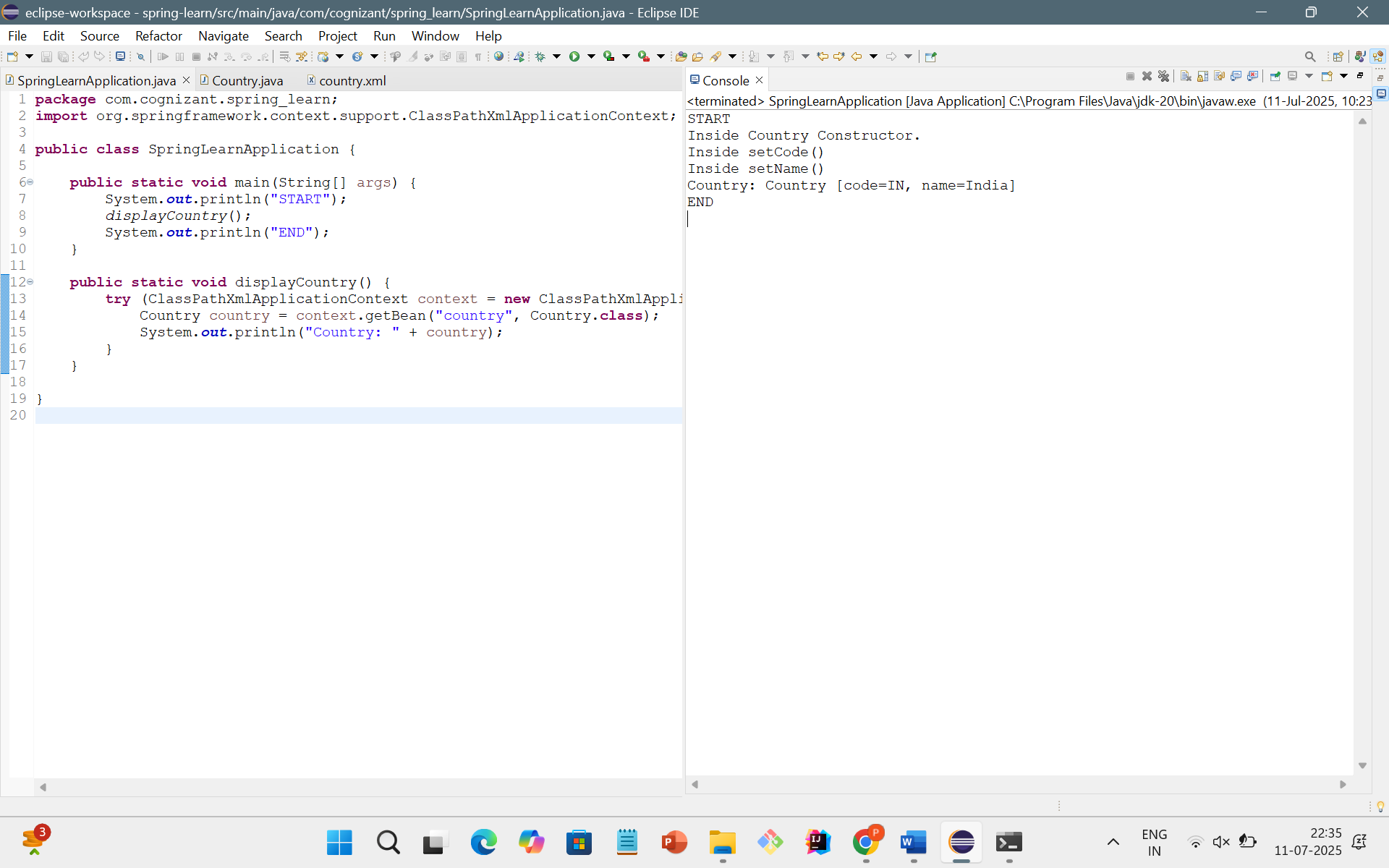
System.***out***.println("Country: " + country);

}

}

}

OUTPUT:



**Hello World RESTful Web Service**

CODE:

HelloController.java

**package** com.cognizant.spring\_learn.controller;

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

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

@RestController

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

@GetMapping("/hello")

**public** String sayHello() {

System.***out***.println("START");

String message = "Hello World!!";

System.***out***.println("END");

**return** message;

}

}

SpringLearnApplication.java

**package** com.cognizant.spring\_learn;

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

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

@SpringBootApplication

**public** **class** SpringLearnApplication {

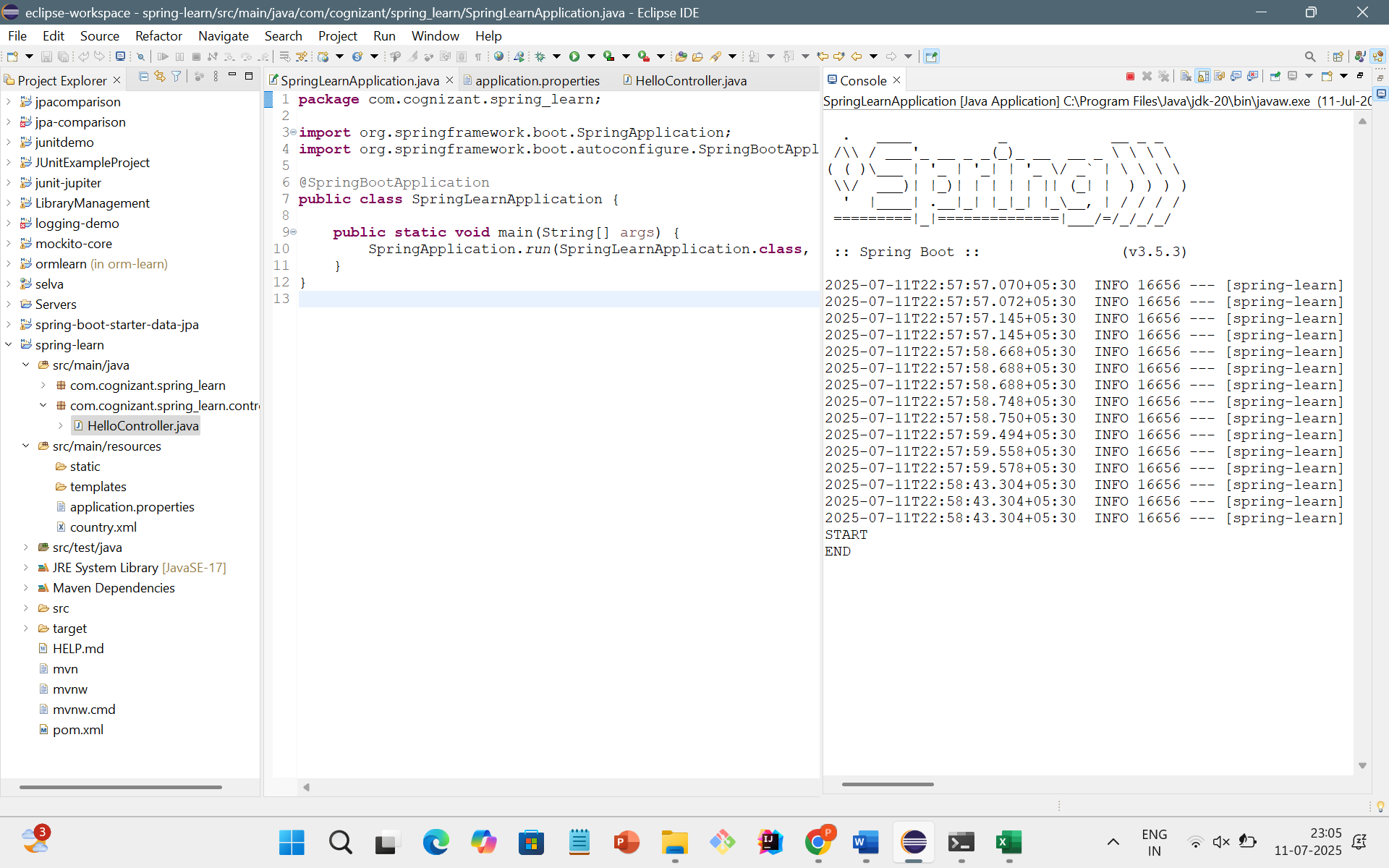
**public** **static** **void** main(String[] args) {

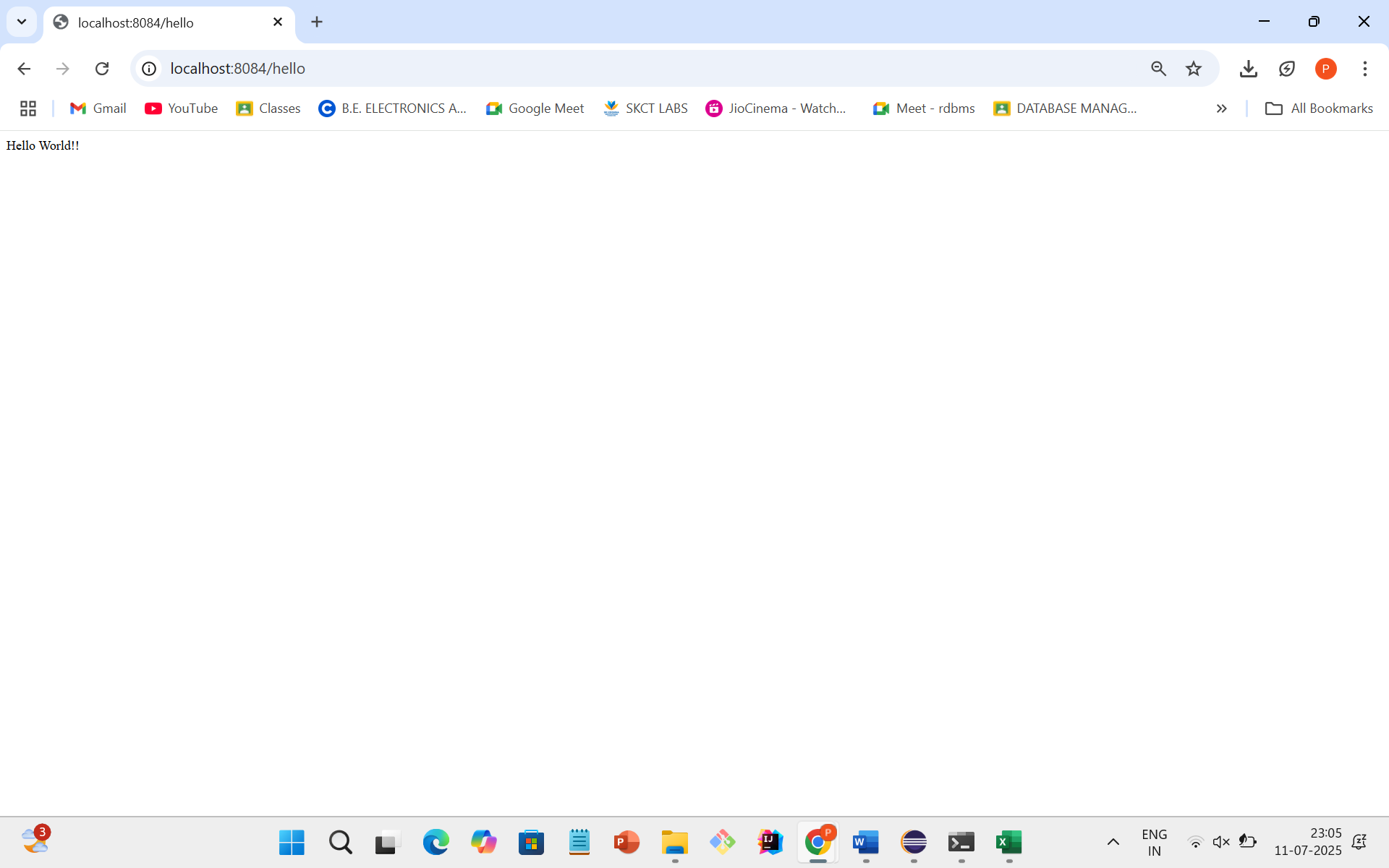
SpringApplication.*run*(SpringLearnApplication.**class**, args);

}

}

OUTPUT:





**REST - Country Web Service**

CODE:

Country.java

**package** com.cognizant.spring\_learn;

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

**private** String code;

**private** String name;

**public** Country() {

System.***out***.println("Inside Country Constructor.");

}

**public** String getCode() {

System.***out***.println("Inside getCode()");

**return** code;

}

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

System.***out***.println("Inside setCode()");

**this**.code = code;

}

**public** String getName() {

System.***out***.println("Inside getName()");

**return** name;

}

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

System.***out***.println("Inside setName()");

**this**.name = name;

}

@Override

**public** String toString() {

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

}

}

CountryController.java

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.Country;

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 {

@RequestMapping("/country")

public Country getCountryIndia() {

System.out.println("START");

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

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

System.out.println("END");

return country;

}

}

SpringLearnApplication.java

**package** com.cognizant.spring\_learn;

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

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

@SpringBootApplication

**public** **class** SpringLearnApplication {

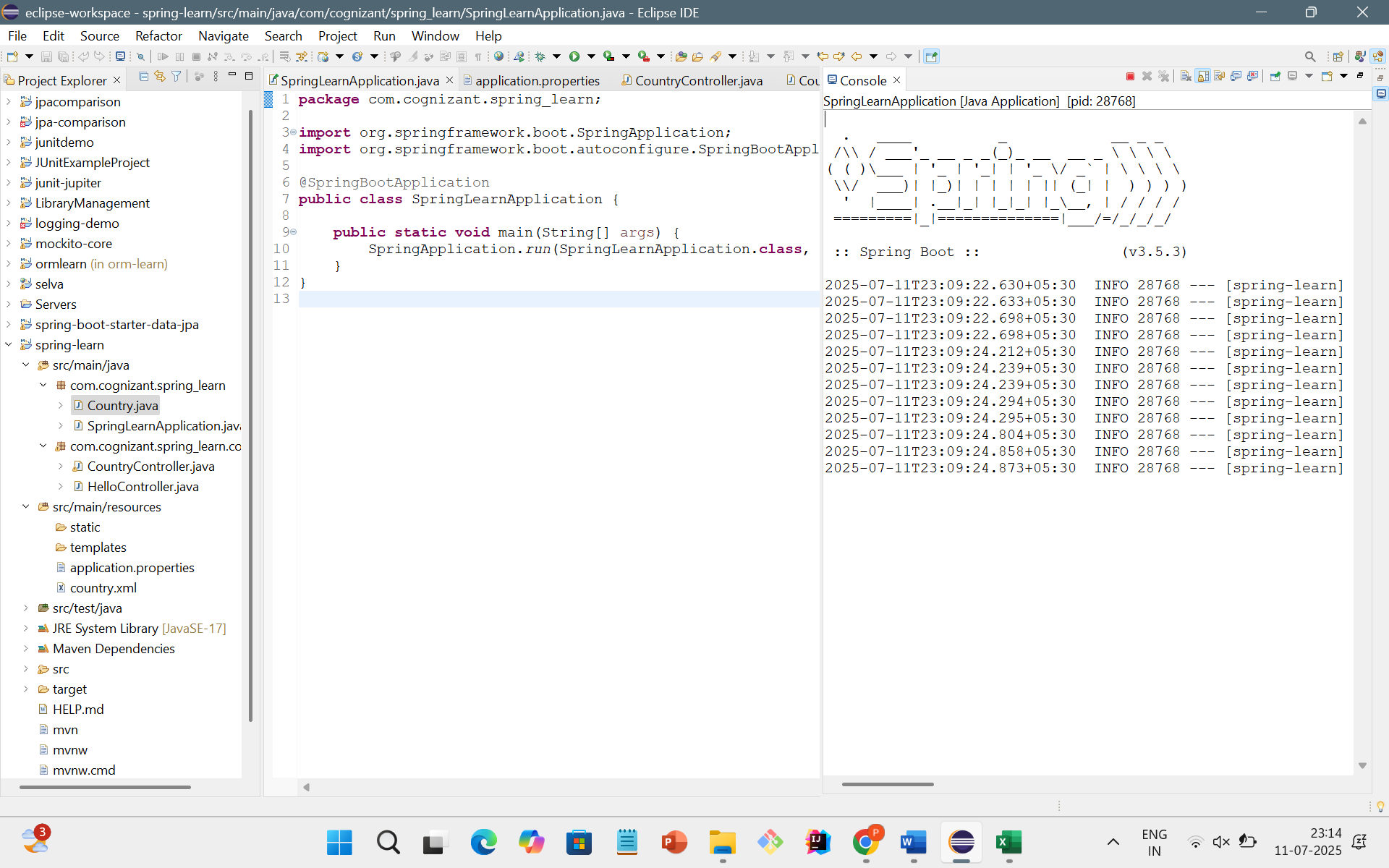
**public** **static** **void** main(String[] args) {

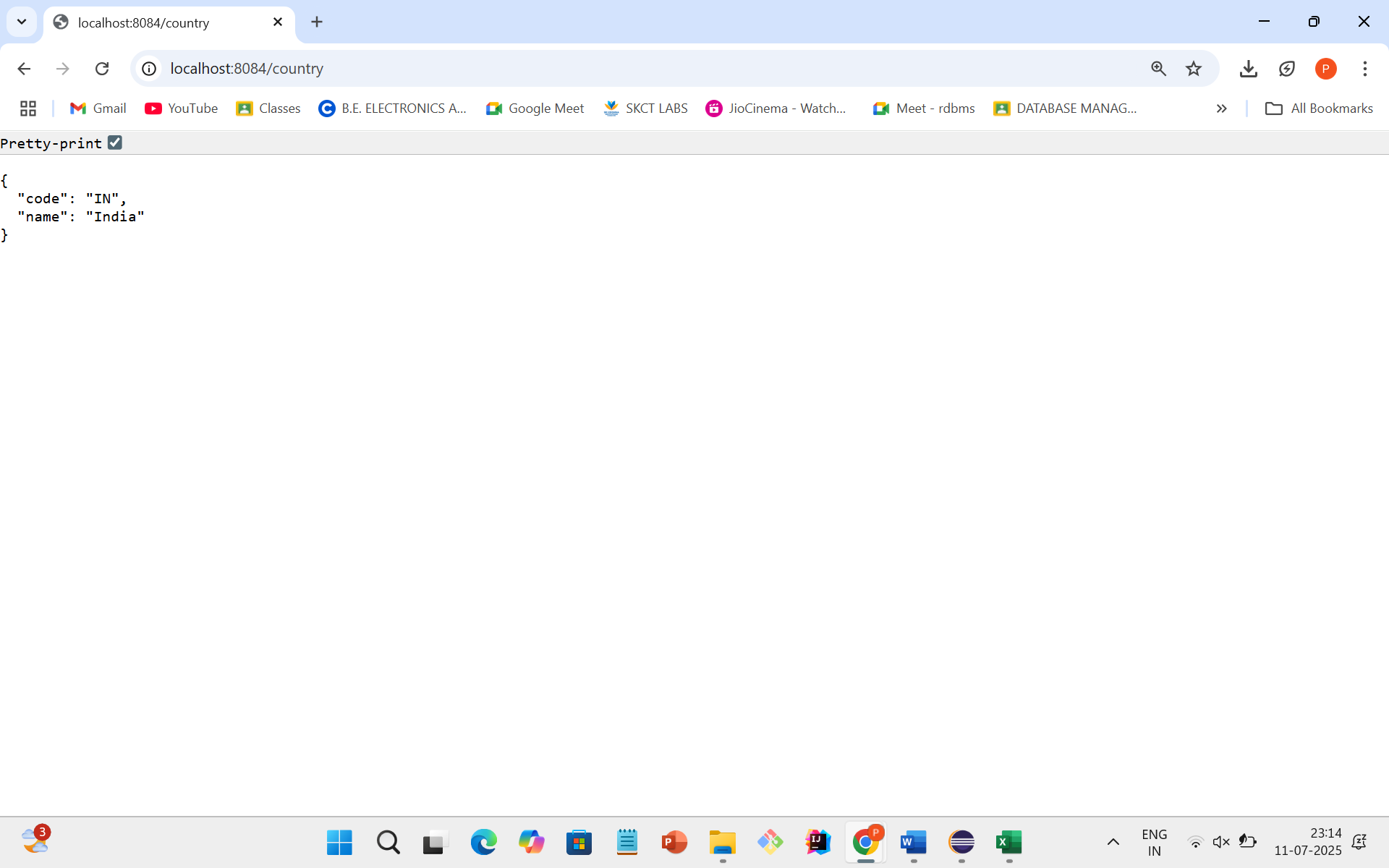
SpringApplication.*run*(SpringLearnApplication.**class**, args);

}

}

OUTPUT:





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

CODE:

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=*"countryList"* class=*"java.util.ArrayList"*>

<constructor-arg>

<list>

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

</list>

</constructor-arg>

</bean>

</beans>

Country.java

**package** com.cognizant.spring\_learn;

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

**private** String code;

**private** String name;

**public** Country() {

System.***out***.println("Inside Country Constructor.");

}

**public** String getCode() {

System.***out***.println("Inside getCode()");

**return** code;

}

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

System.***out***.println("Inside setCode()");

**this**.code = code;

}

**public** String getName() {

System.***out***.println("Inside getName()");

**return** name;

}

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

System.***out***.println("Inside setName()");

**this**.name = name;

}

@Override

**public** String toString() {

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

}

}

CountryController.java

**package** com.cognizant.spring\_learn.controller;

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

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

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

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

@RestController

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

@Autowired

**private** CountryService countryService;

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

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

System.***out***.println("START");

Country country = countryService.getCountry(code);

System.***out***.println("END");

**return** country;

}

}

SpringLearnApplication.java

**package** com.cognizant.spring\_learn;

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

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

@SpringBootApplication

**public** **class** SpringLearnApplication {

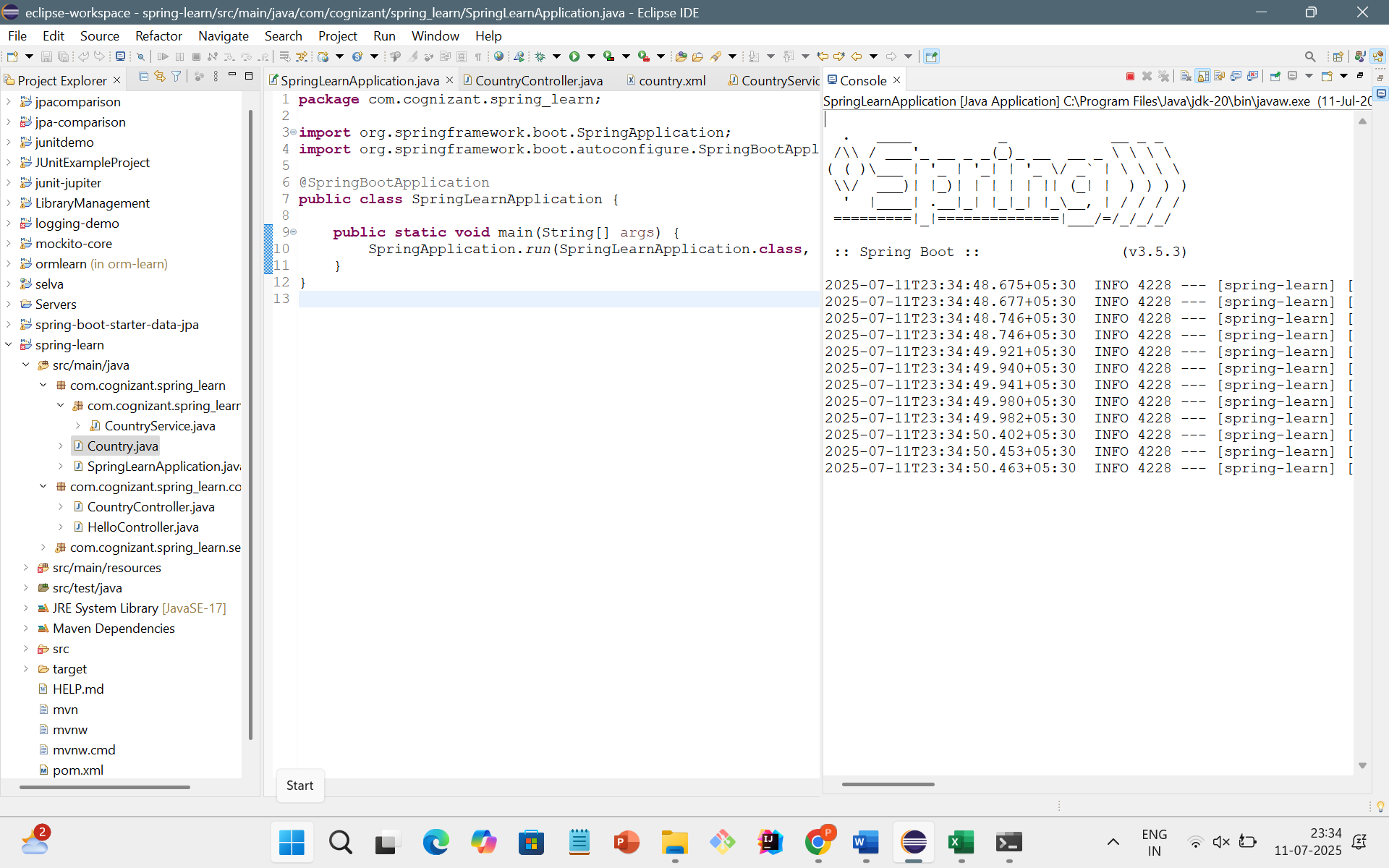
**public** **static** **void** main(String[] args) {

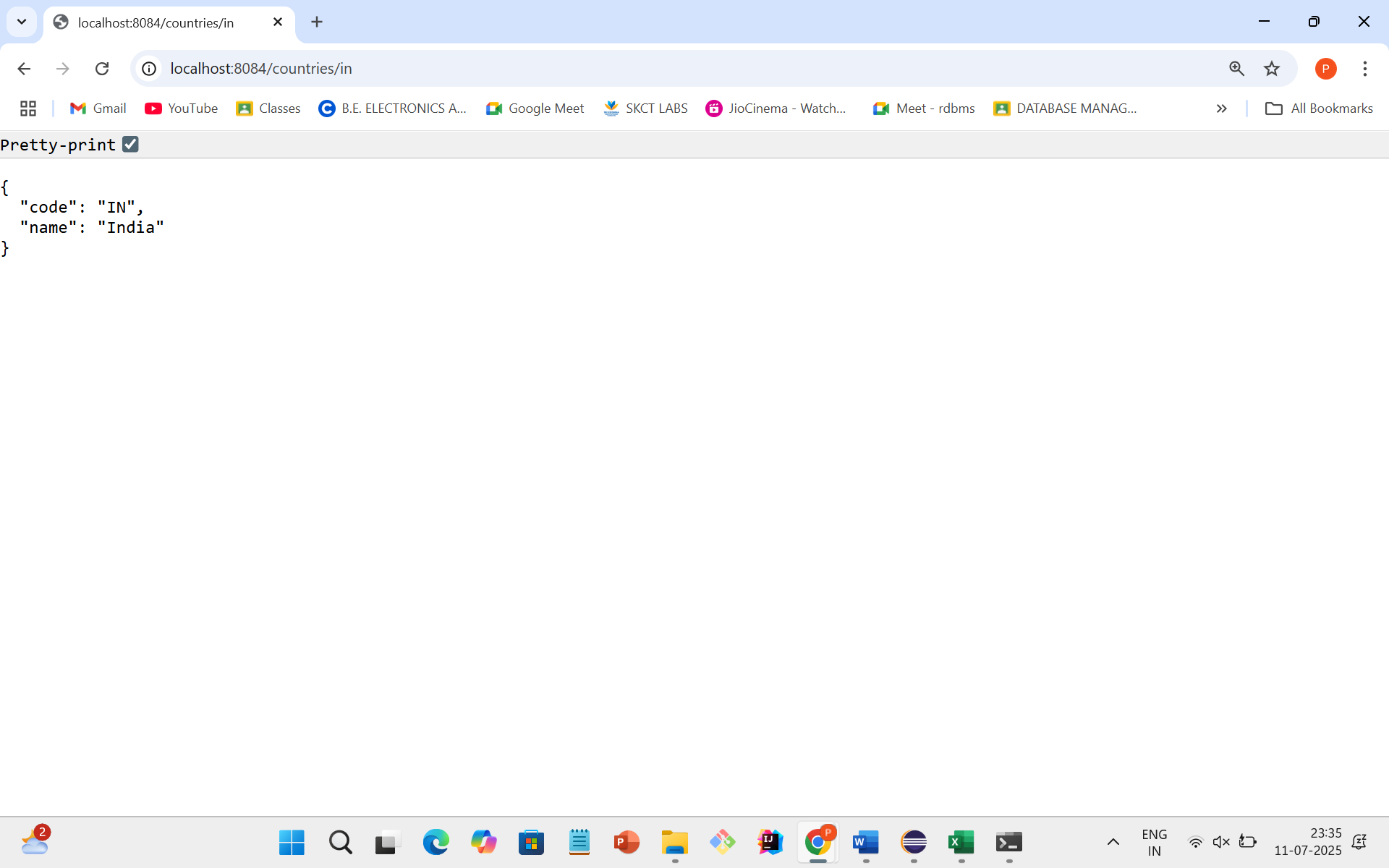
SpringApplication.*run*(SpringLearnApplication.**class**, args);

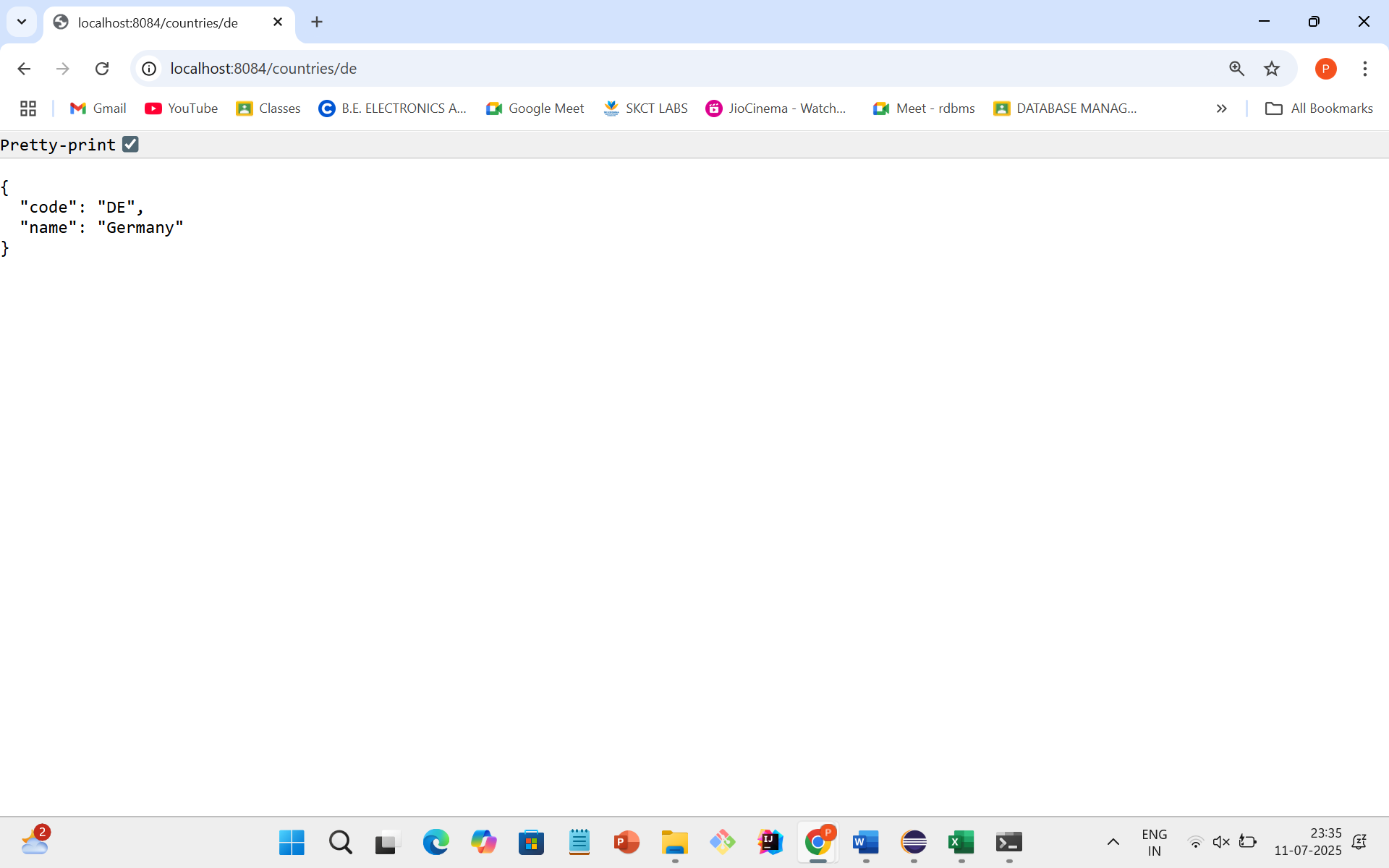
}

}

OUTPUT:







**Create authentication service that returns JWT** 

CODE:

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.example</groupId>

<artifactId>jwtauthentication</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>jwtauthentication</name>

<description>Demo project for Spring Boot</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-security</artifactId>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

JwtUtil.java

**package** com.example.jwtauthentication.util;

**import** io.jsonwebtoken.Jwts;

**import** io.jsonwebtoken.SignatureAlgorithm;

**import** org.springframework.stereotype.Component;

**import** java.util.Date;

@Component

**public** **class** JwtUtil {

**private** **final** String SECRET\_KEY = "secretkey123";

**public** String generateToken(String username) {

**long** now = System.*currentTimeMillis*();

**return** Jwts.*builder*()

.setSubject(username)

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

.setExpiration(**new** Date(now + 1000 \* 60 \* 60)) // valid for 1 hour

.signWith(SignatureAlgorithm.***HS256***, SECRET\_KEY)

.compact();

}

}

AuthenticationController.java

**package** com.example.jwtauthentication.controller;

**import** org.springframework.http.HttpHeaders;

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

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

**import** java.util.Base64;

@RestController

**public** **class** AuthenticationController {

@GetMapping("/authenticate")

**public** ResponseEntity<?> authenticate(@RequestHeader(HttpHeaders.***AUTHORIZATION***) String authHeader) {

**try** {

// Check if header starts with Basic

**if** (authHeader == **null** || !authHeader.startsWith("Basic ")) {

**return** ResponseEntity.*status*(401).body("Missing or invalid Authorization header");

}

// Decode base64 encoded username:password

String base64Credentials = authHeader.substring("Basic ".length());

**byte**[] decodedBytes = Base64.*getDecoder*().decode(base64Credentials);

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

String[] values = credentials.split(":", 2);

String username = values[0];

String password = values[1];

// Dummy check (replace with actual validation later)

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

// Generate dummy JWT token (for now)

String fakeJwt = "eyJhbGciOiJIUzI1NiJ9.fake.jwt.token";

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

} **else** {

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

}

} **catch** (Exception e) {

**return** ResponseEntity.*status*(500).body("Error: " + e.getMessage());

}

}

}

SecurityConfig.java

**package** com.example.jwtauthentication.config;

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

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

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

**import** org.springframework.security.core.userdetails.User;

**import** org.springframework.security.core.userdetails.UserDetailsService;

**import** org.springframework.security.provisioning.InMemoryUserDetailsManager;

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

@Configuration

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

@Bean

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

http.csrf().disable()

.authorizeHttpRequests()

.anyRequest().authenticated()

.and()

.httpBasic();

**return** http.build();

}

@Bean

**public** UserDetailsService userDetailsService() {

**return** **new** InMemoryUserDetailsManager(

User.withDefaultPasswordEncoder()

.username("user")

.password("pwd")

.roles("USER")

.build()

);

}

}

JwtauthenticatonApplication.java:

**package** com.example.jwtauthentication;

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

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

@SpringBootApplication

**public** **class** JwtauthenticationApplication {

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

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

}

}

OUTPUT:

