**WEEK 4\_Spring REST using Spring Boot 3**

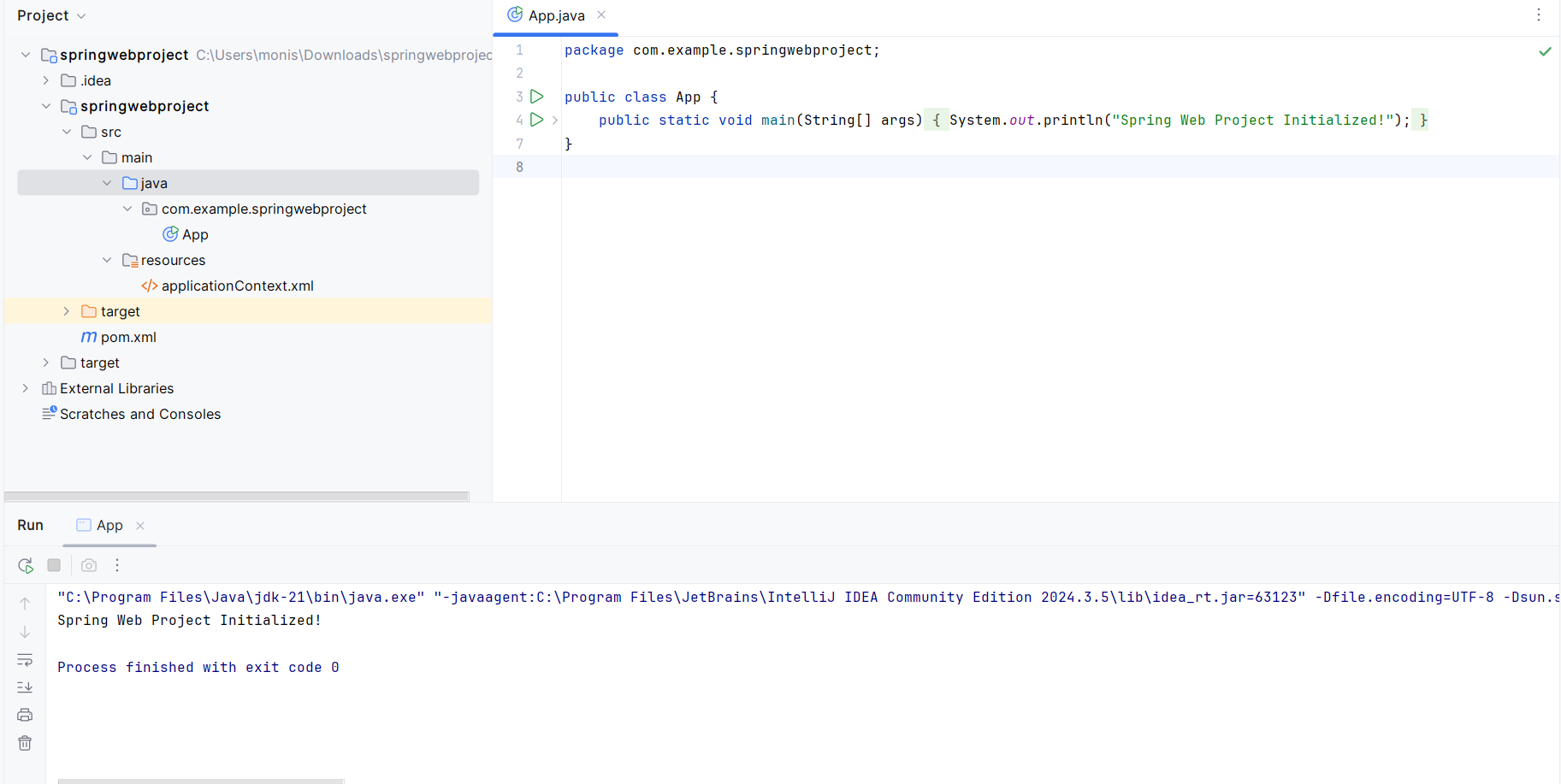
**Exercise 1: Create a Spring Web Project using Maven**

**Scenario:**

A developer wants to create a basic Spring Web application using Maven in IntelliJ to set up initial project structure.

**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>springwebproject</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <dependencies>  
 *<!-- Spring Web -->* <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-web</artifactId>  
 <version>6.1.4</version>  
 </dependency>  
  
 *<!-- Servlet API -->* <dependency>  
 <groupId>jakarta.servlet</groupId>  
 <artifactId>jakarta.servlet-api</artifactId>  
 <version>6.0.0</version>  
 <scope>provided</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 <configuration>  
 <source>17</source>  
 <target>17</target>  
 </configuration>  
 </plugin>  
 </plugins>  
 </build>  
</project>

**OUTPUT:**



**Exercise 2: Spring Core – Load Country from Spring Configuration XML**

**Scenario:**

A Spring application loads a Country bean from an XML configuration file and displays the country name using ApplicationContext.

**CODE:**

**ApplicationContext.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.example.springwebproject.Country">  
 <property name="name" value="India"/>  
 </bean>  
  
</beans>

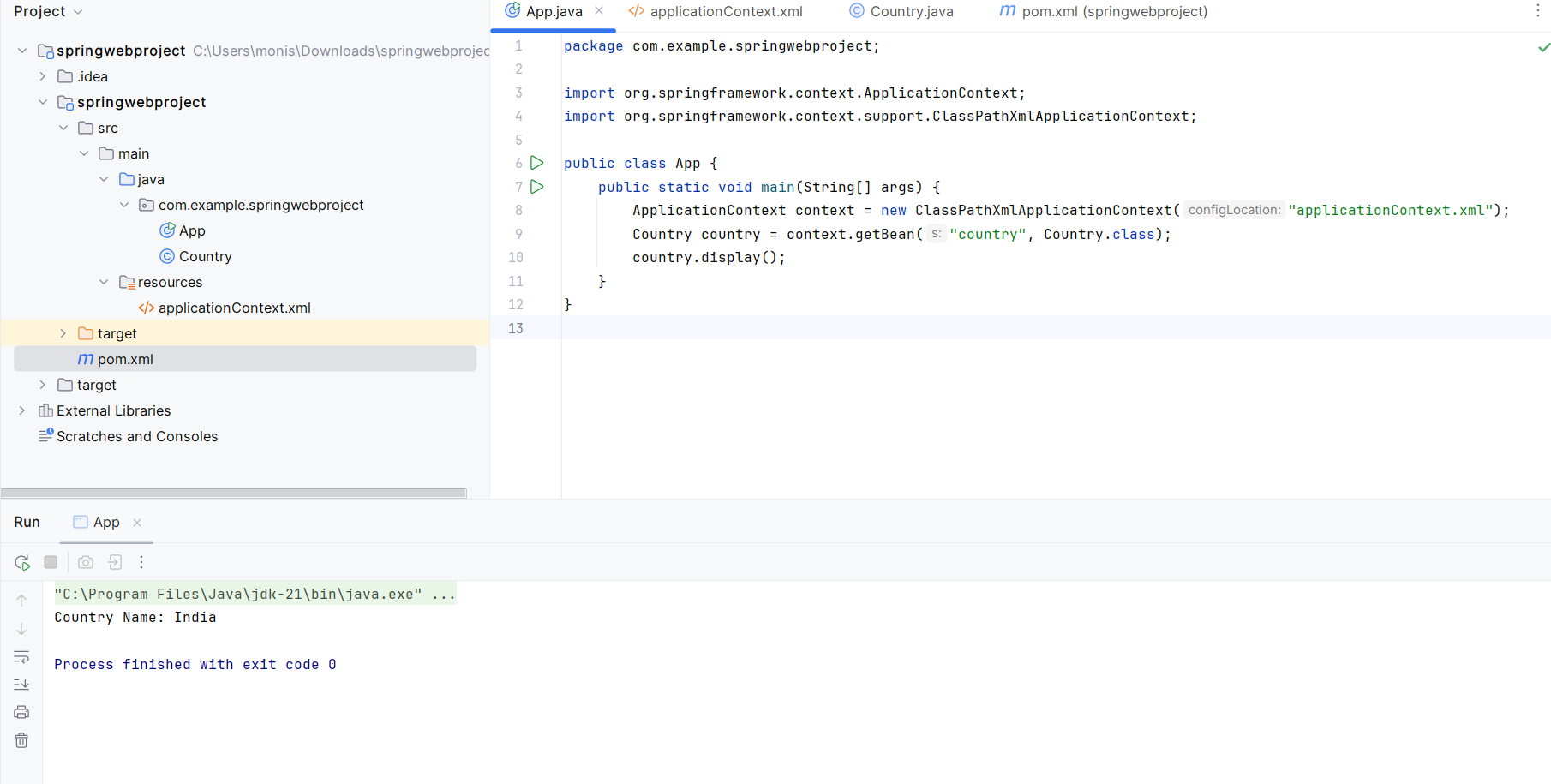
**Country.java:**

package com.example.springwebproject;  
  
public class Country {  
 private String name;  
  
 public Country() {}  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void display() {  
 System.*out*.println("Country Name: " + name);  
 }  
}

**App.java:**

package com.example.springwebproject;  
  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class App {  
 public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
 Country country = context.getBean("country", Country.class);  
 country.display();  
 }  
}

**OUTPUT:**

****

**Exercise 3: Hello World RESTful Web Service**

**Scenario:**

A RESTful endpoint is created using Spring Boot that returns “Hello World!” when accessed via a browser or HTTP client.

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

    <modelVersion>4.0.0</modelVersion>

    <groupId>com.example</groupId>

    <artifactId>helloworld</artifactId>

    <version>0.0.1-SNAPSHOT</version>

    <parent>

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

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

        <version>3.2.2</version>

    </parent>

    <dependencies>

        <dependency>

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

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

        </dependency>

    </dependencies>

    <build>

        <plugins>

            <plugin>

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

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

            </plugin>

        </plugins>

    </build>

</project>

**Hellocontroller.java:**

package com.example.helloworld;

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

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

@RestController

public class HelloController {

    @GetMapping("/")

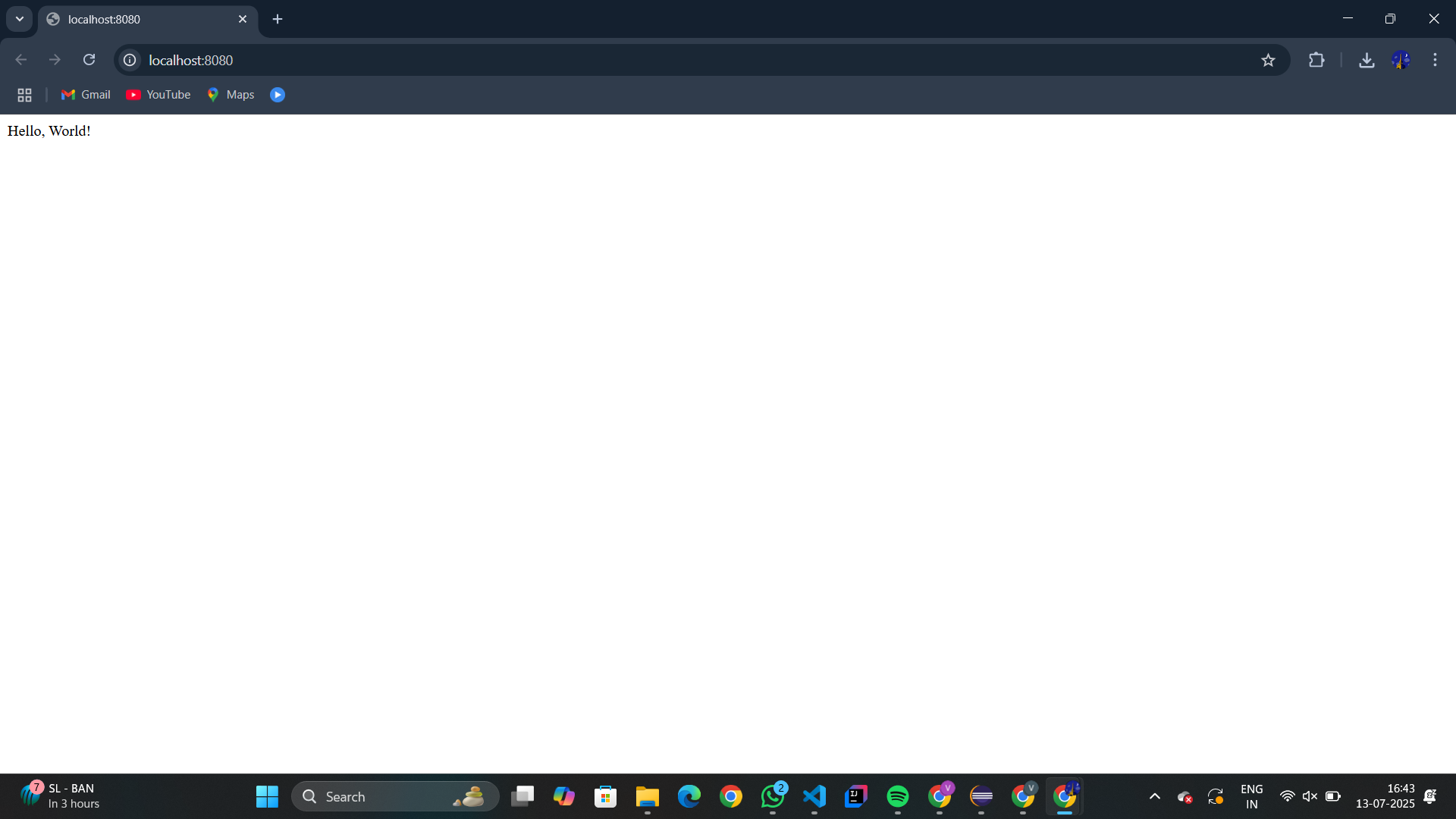
    public String hello() {

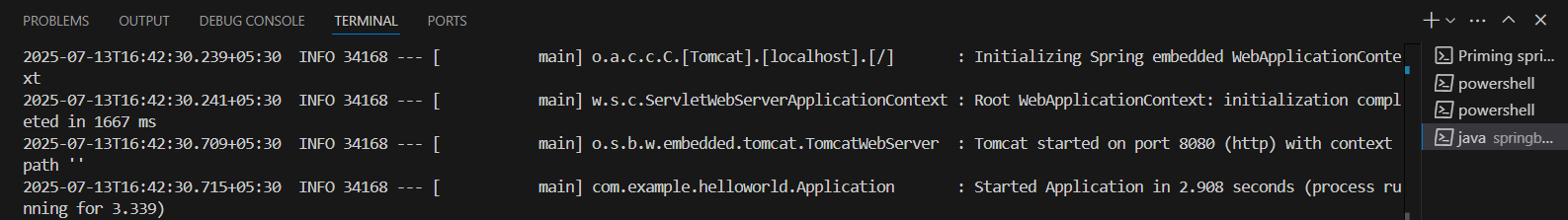
        return "Hello, World!";

    }

}

**OUTPUT:**

****

****

**Exercise 4: REST - Country Web Service**

**Scenario:**

A Spring Boot application provides a RESTful web service that returns country information in JSON format when users send a GET request to the /country endpoint.

**CODE:**

**Countrycontroller.java:**

package com.example.countryservice;

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

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

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

import java.util.List;

import java.util.Arrays;

@RestController

public class CountryController {

    private List<Country> countries = Arrays.asList(

        new Country("IN", "India"),

        new Country("US", "United States"),

        new Country("FR", "France")

    );

    @GetMapping("/countries")

    public List<Country> getAllCountries() {

        return countries;

    }

    @GetMapping("/countries/{code}")

    public Country getCountryByCode(@PathVariable String code) {

        return countries.stream()

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

                        .findFirst()

                        .orElse(null);

    }

}

**Application.java:**

package com.example.countryservice;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

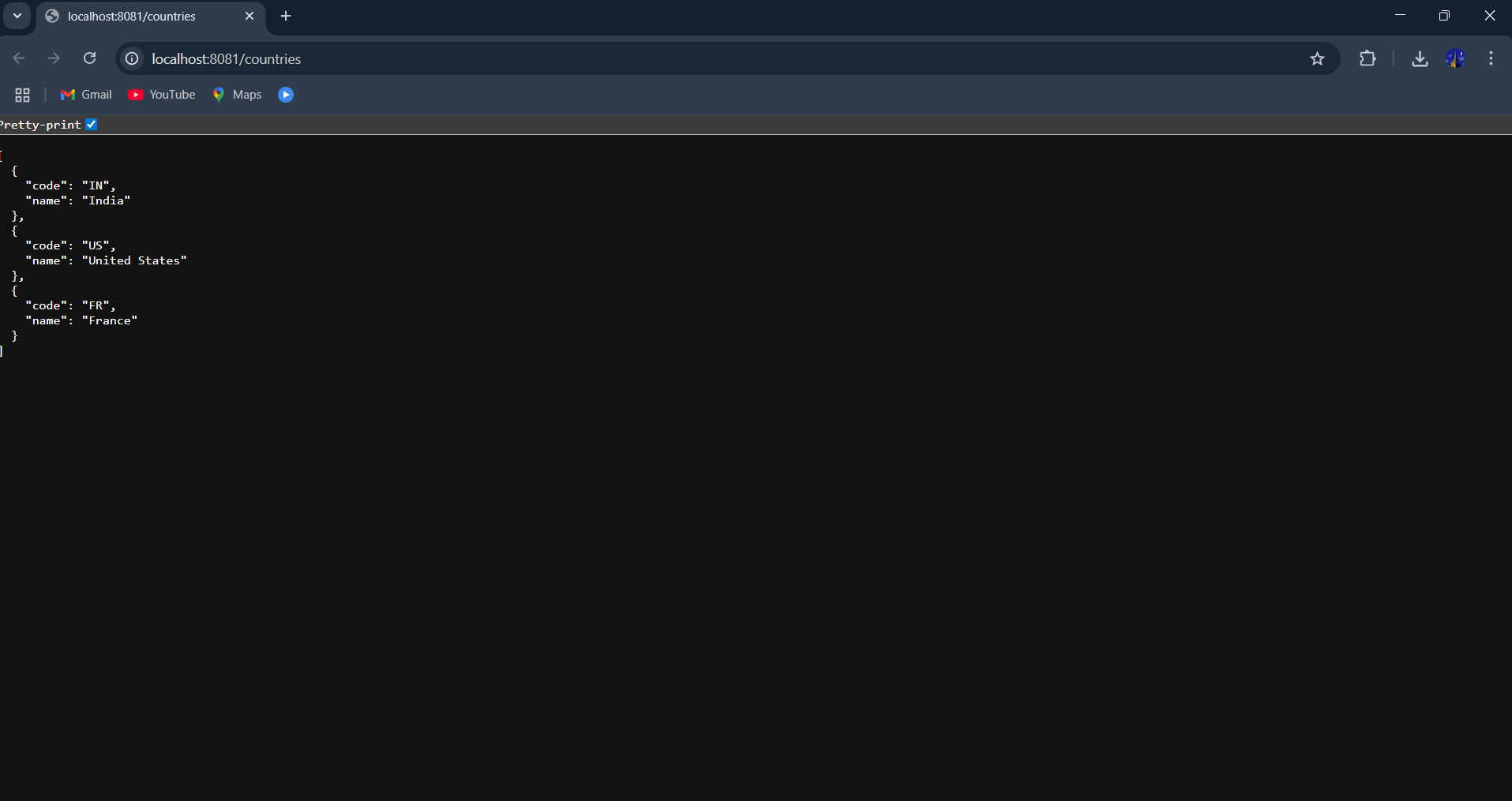
    public static void main(String[] args) {

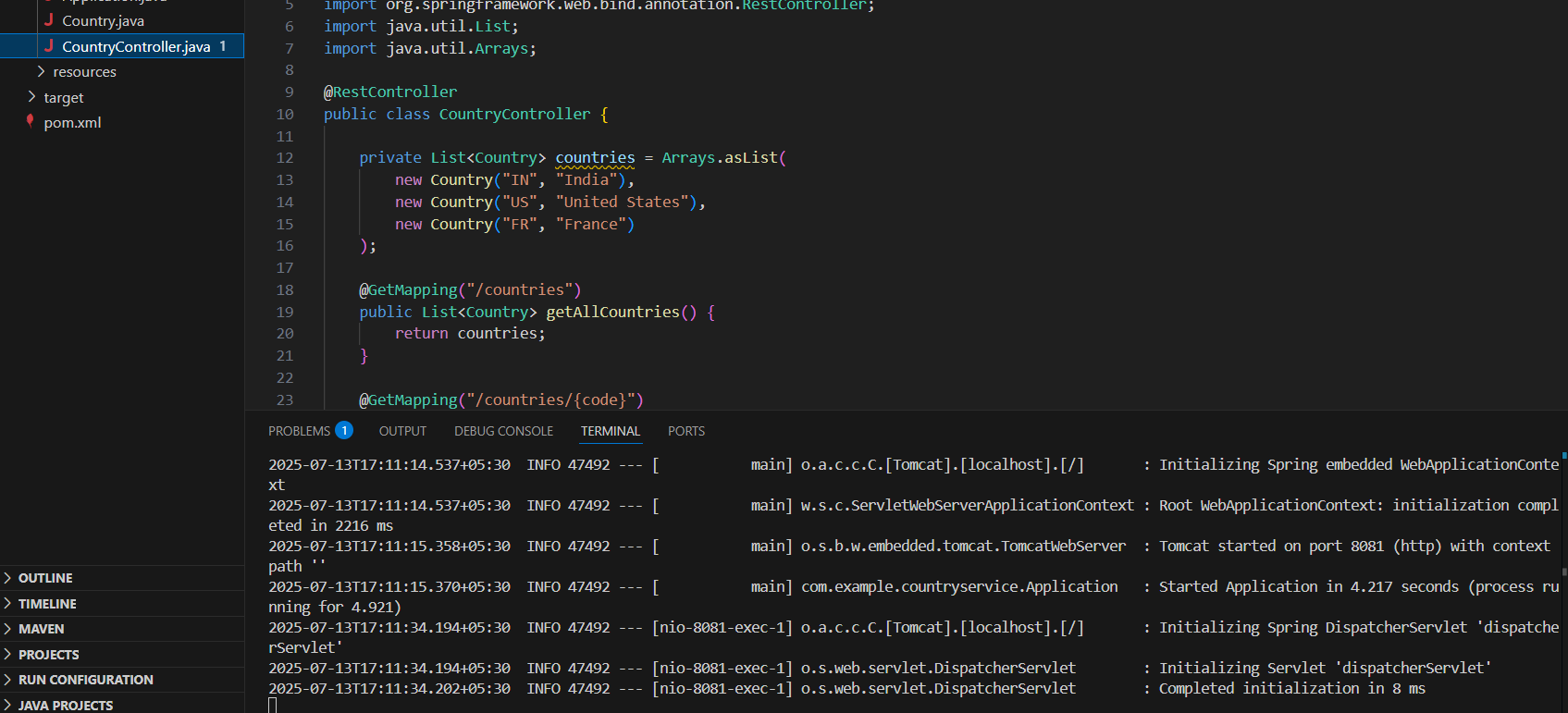
        SpringApplication.run(Application.class, args);

    }

}

**OUTPUT:**

****

****

**Exercise 5: REST - Get country based on country code**

**Scenario:**

The REST API accepts a country code as a path variable in the URL (e.g., /country/IN) and responds with detailed country information in structured JSON format.

**Code:**

**Country.java:**

package com.example.countryservice;

public class Country {

private String code;

private String name;

public Country(String code, String name) {

this.code = code;

this.name = name;

}

public String getCode() {

return code;

}

public String getName() {

return name;

}

}

**CountryController.java:**

package com.example.countryservice;

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

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

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

import java.util.List;

import java.util.Arrays;

@RestController

public class CountryController {

private List<Country> countries = Arrays.asList(

new Country("IN", "India"),

new Country("US", "United States"),

new Country("FR", "France")

);

@GetMapping("/countries")

public List<Country> getAllCountries() {

return countries;

}

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

public Country getCountryByCode(@PathVariable String code) {

return countries.stream()

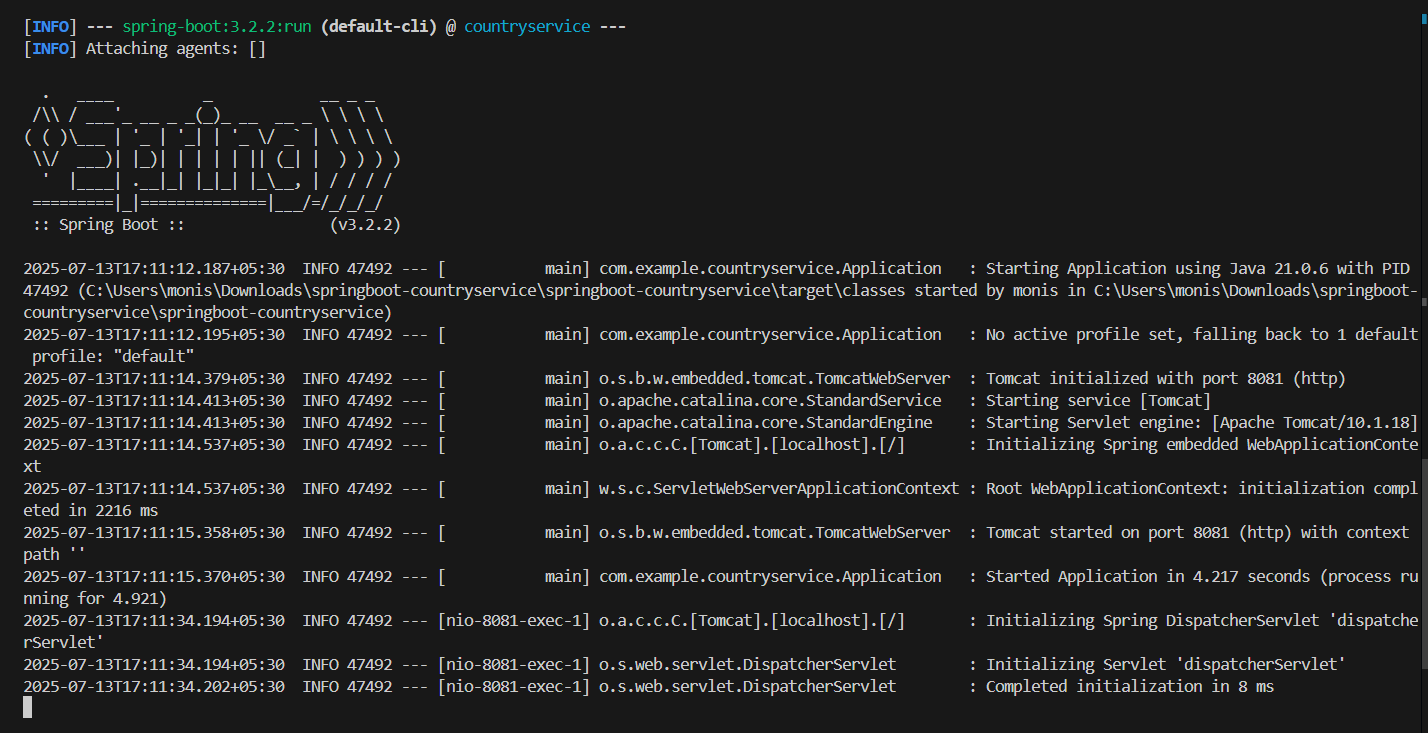
.filter(c -> c.getCode().equalsIgnoreCase(code))

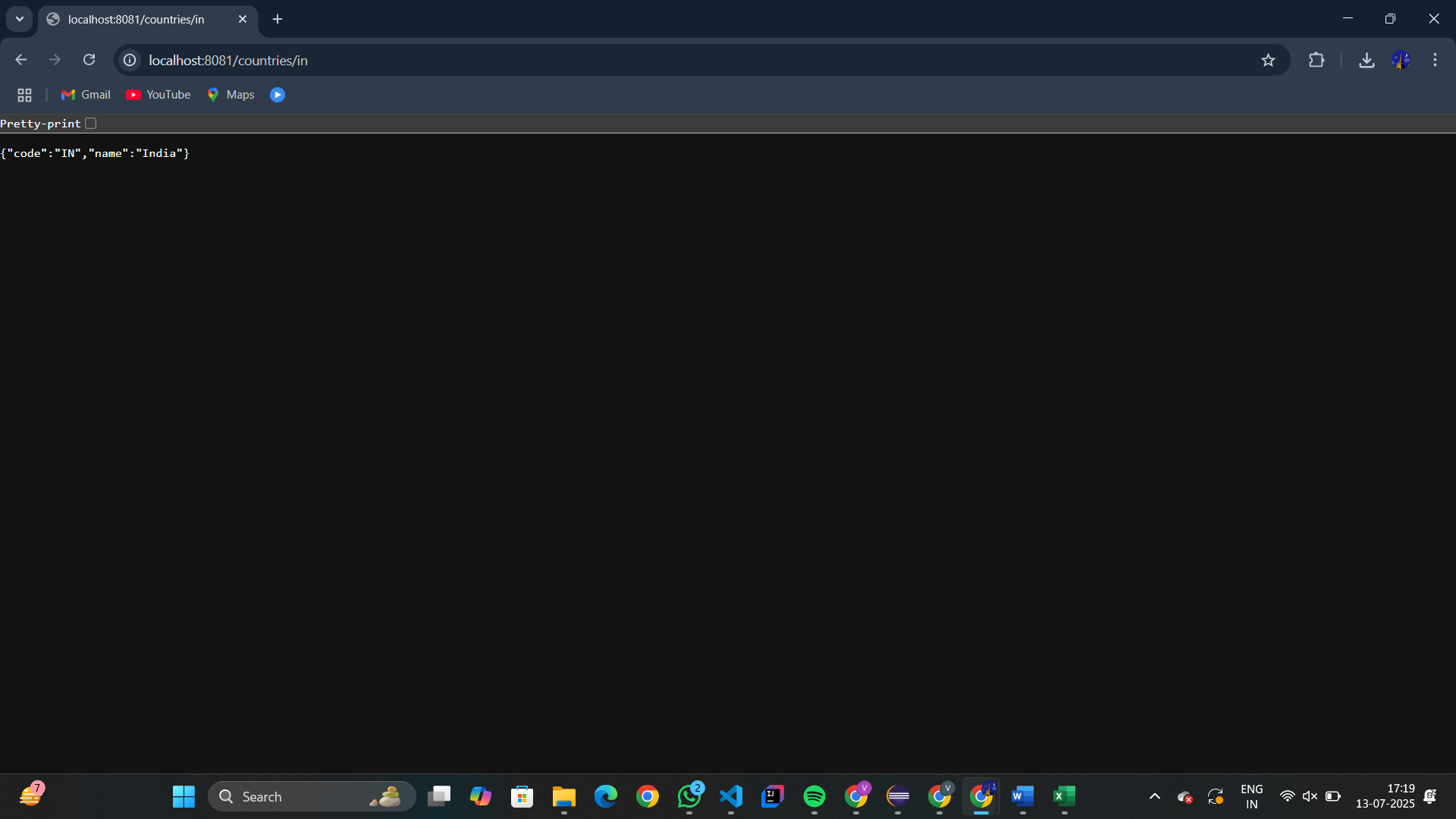
.findFirst()

.orElse(null);

}

}

**Output:**

****

**Exercise 6: Create authentication service that returns JWT**

**Scenario:**

A Spring Boot service handles user login via REST. Upon receiving valid credentials, it generates and returns a secure JWT token for subsequent authenticated API access.

**Code:**

**Authcontroller.java:**

package com.example.jwtauth;

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

import java.util.HashMap;

import java.util.Map;

@RestController

public class AuthController {

    @PostMapping("/authenticate")

    public Map<String, String> authenticate(@RequestParam String username, @RequestParam String password) {

        Map<String, String> response = new HashMap<>();

        if ("admin".equals(username) && "password".equals(password)) {

            String token = JwtUtil.generateToken(username);

            response.put("token", token);

        } else {

            response.put("error", "Invalid credentials");

        }

        return response;

    }

}

**Securecontroller.java:**

package com.example.jwtauth;

import io.jsonwebtoken.Claims;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.security.Keys;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import java.security.Key;

import java.util.HashMap;

import java.util.Map;

@RestController

public class SecureController {

    private static final Key key = JwtUtil.getKey();

    @GetMapping("/secure")

    public ResponseEntity<?> secureEndpoint(@RequestHeader("Authorization") String authHeader) {

        try {

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

                return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Missing or invalid Authorization header");

            }

            String token = authHeader.substring(7);

            Claims claims = Jwts.parserBuilder().setSigningKey(key).build().parseClaimsJws(token).getBody();

            String username = claims.getSubject();

            Map<String, String> response = new HashMap<>();

            response.put("message", "Hello, " + username + "! You have accessed a secure endpoint.");

            return ResponseEntity.ok(response);

        } catch (Exception e) {

            return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Invalid or expired token");

        }

    }

}

**Jwtutill.java:**

package com.example.jwtauth;

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 key = Keys.secretKeyFor(SignatureAlgorithm.HS256);

    private static final long EXPIRATION\_TIME = 1000 \* 60 \* 60;

    public static String generateToken(String username) {

        return Jwts.builder()

                   .setSubject(username)

                   .setIssuedAt(new Date())

                   .setExpiration(new Date(System.currentTimeMillis() + EXPIRATION\_TIME))

                   .signWith(key)

                   .compact();

    }

    public static Key getKey() {

        return key;

    }

}

**Output:**

