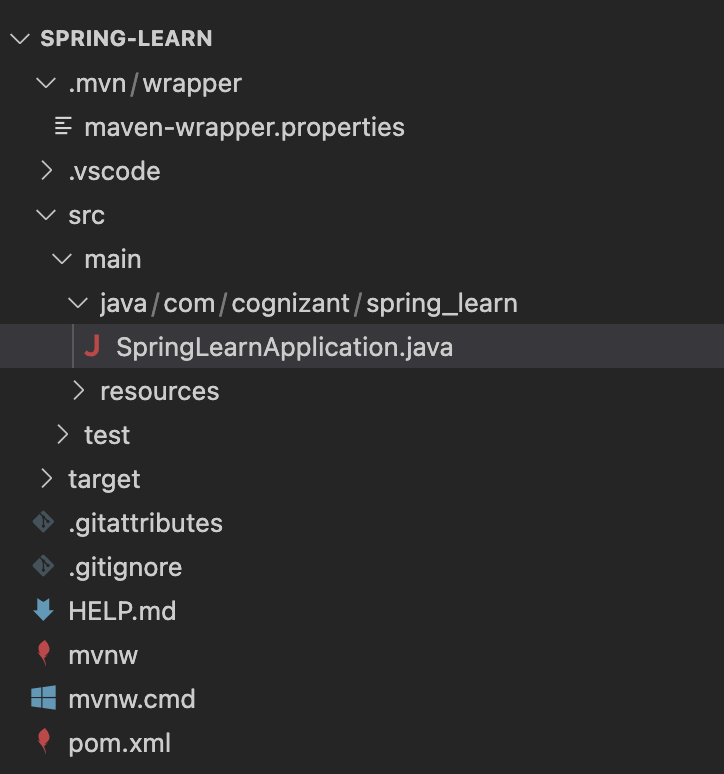
**Hands-on 1: Create a Spring Web Project using Maven**



**SpringLearnApplication.java:**

package com.cognizant.spring\_learn;

public class SpringLearnApplication {

public static void main(String[] args) {

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

System.out.println("Simulating Spring Boot project structure in plain Java.");

System.out.println("This would normally launch Spring Boot context with @SpringBootApplication.");

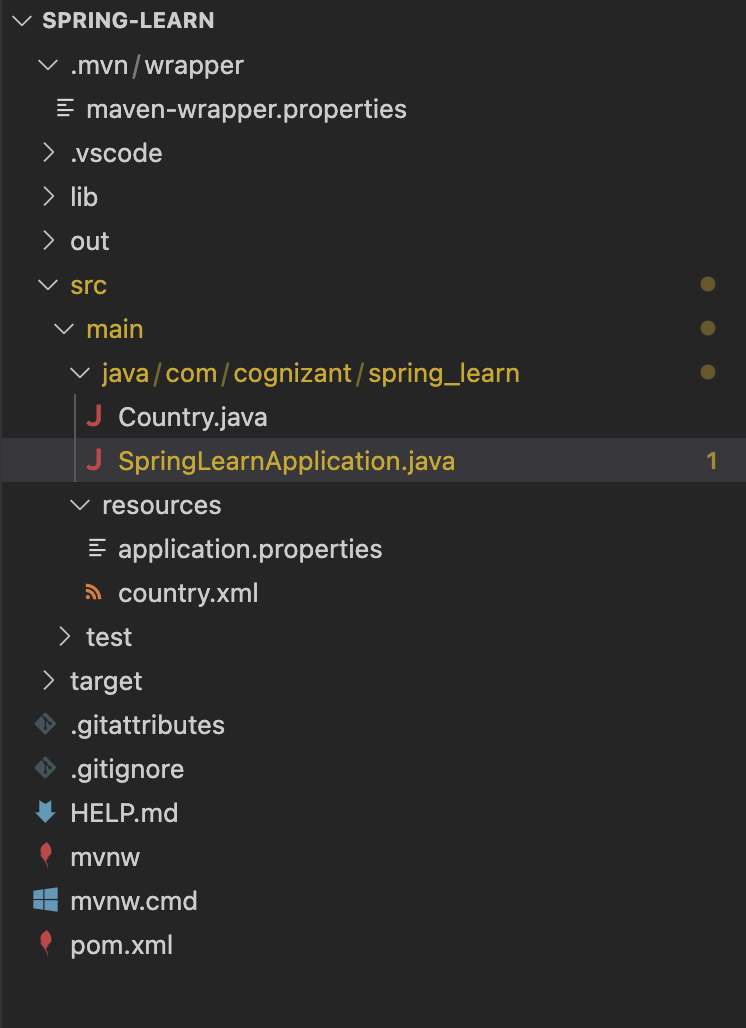
}

}

Output:



**Hands-on 2: Spring Core – Load SimpleDateFormat from Spring Configuration XML**

****

**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.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SpringLearnApplication {

public static void main(String[] args) {

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

displayCountry();

}

public static void displayCountry() {

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

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

System.out.println("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

https://www.springframework.org/schema/beans/spring-beans.xsd">

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

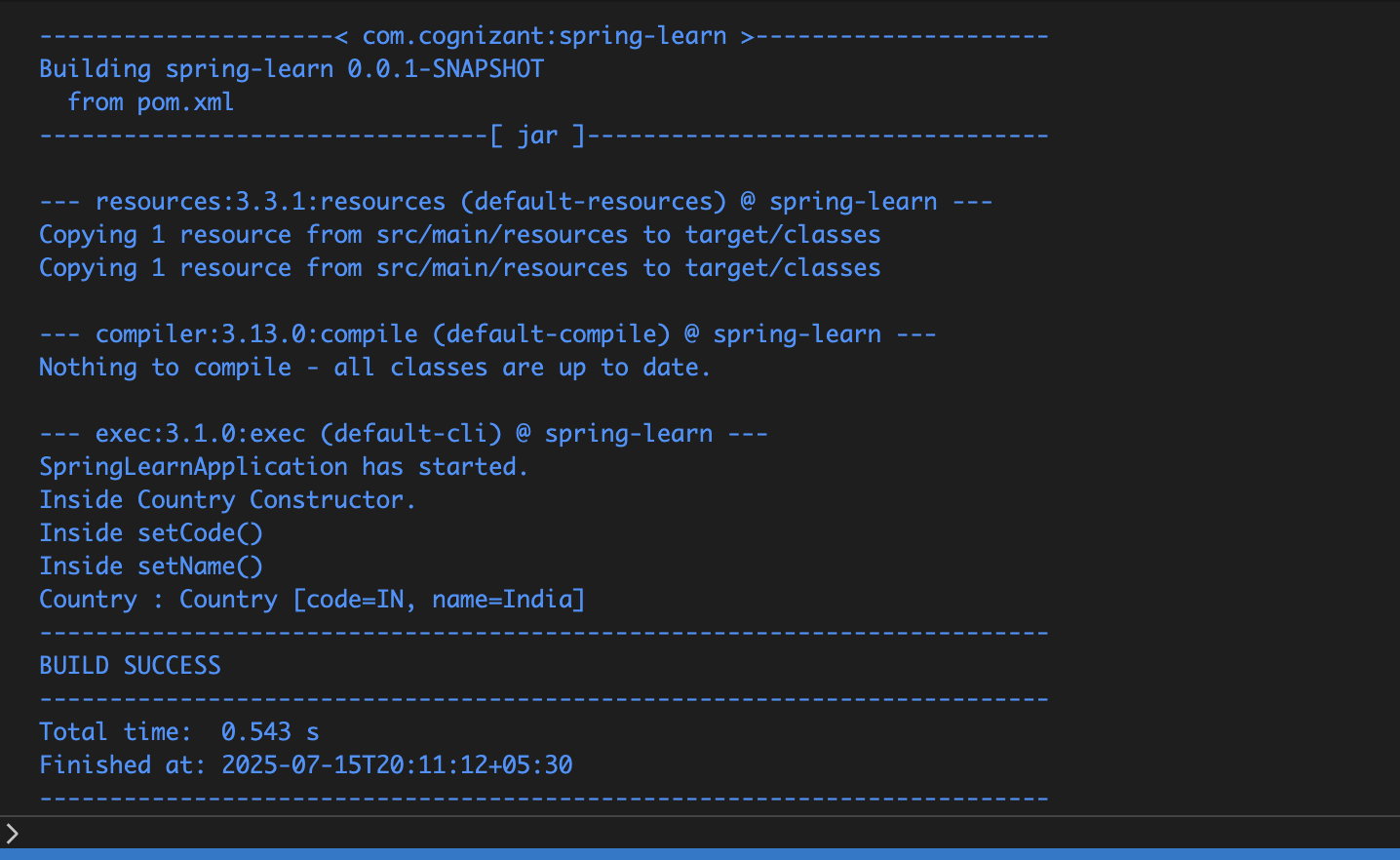
<property name="code" value="IN" />

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

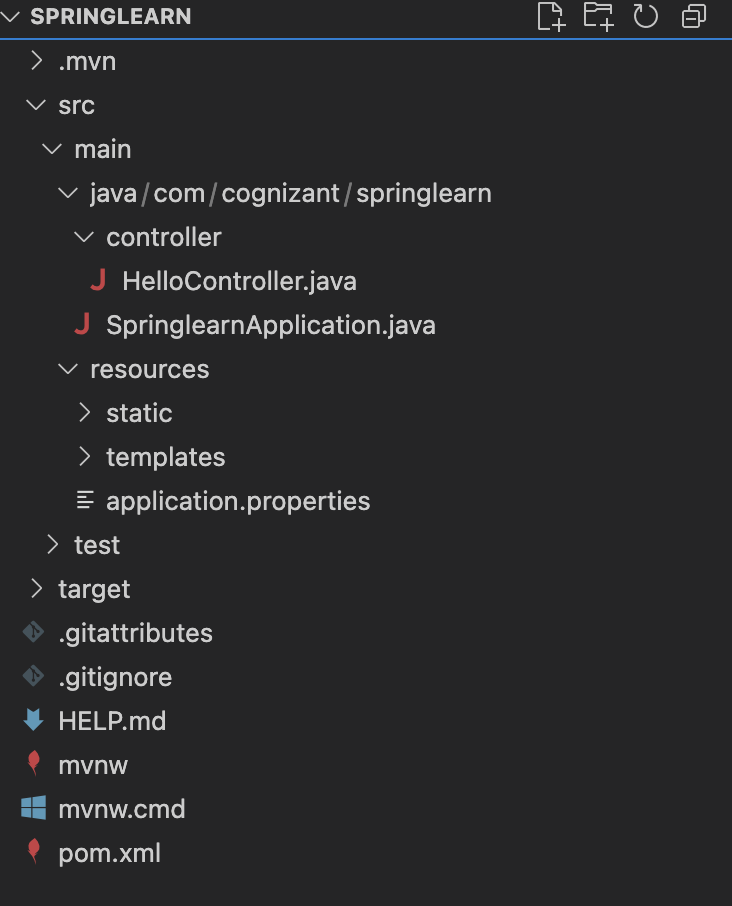
</bean>

</beans>

Output:



**Hands-on 3: Hello World RESTful Web Service**



**HelloController.java:**

package com.cognizant.springlearn.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");

LOGGER.info("End");

return "Hello World!!";

}

}

**SpringLearnApplication.java:**

package com.cognizant.springlearn;

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);

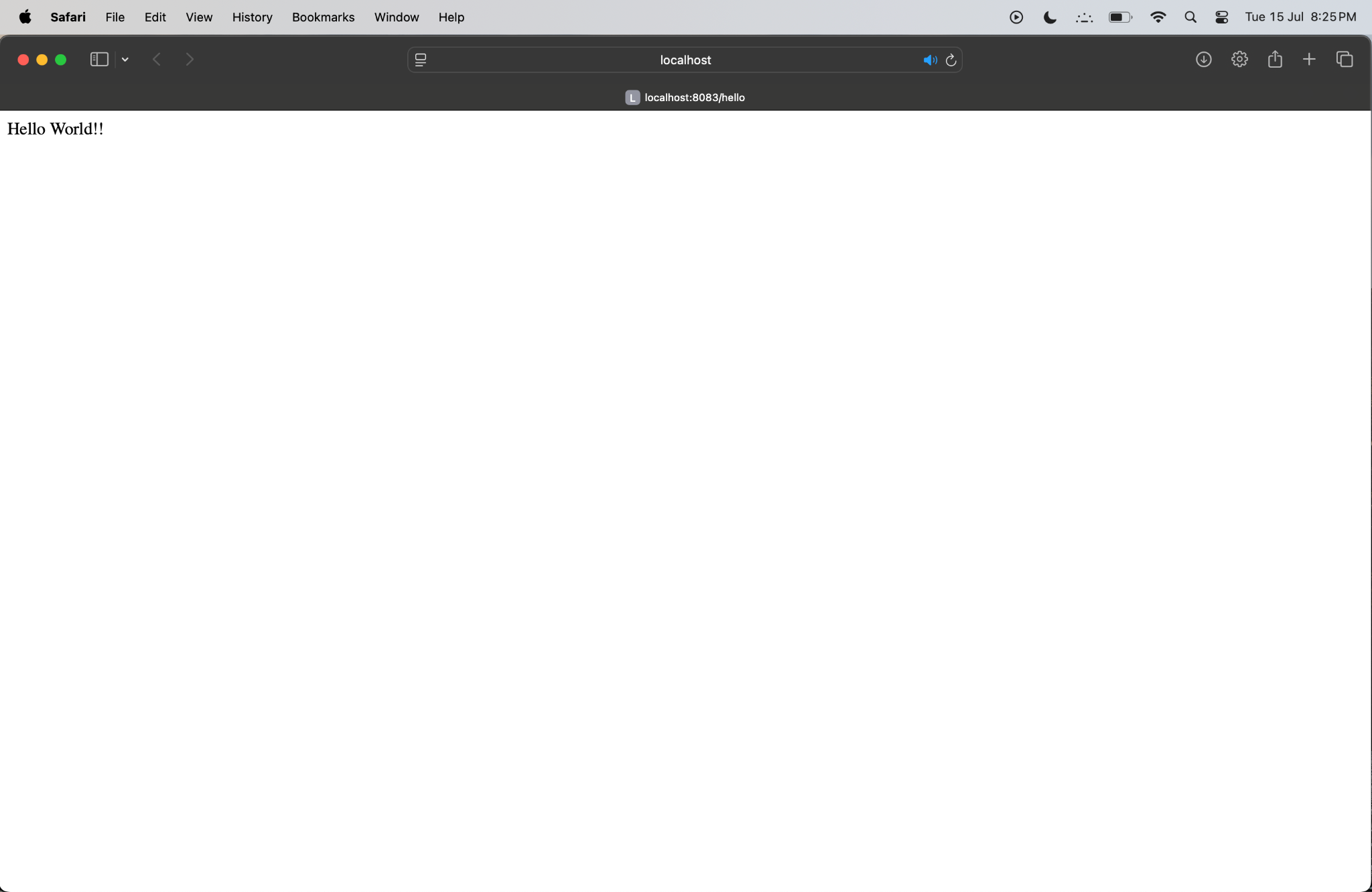
}

}

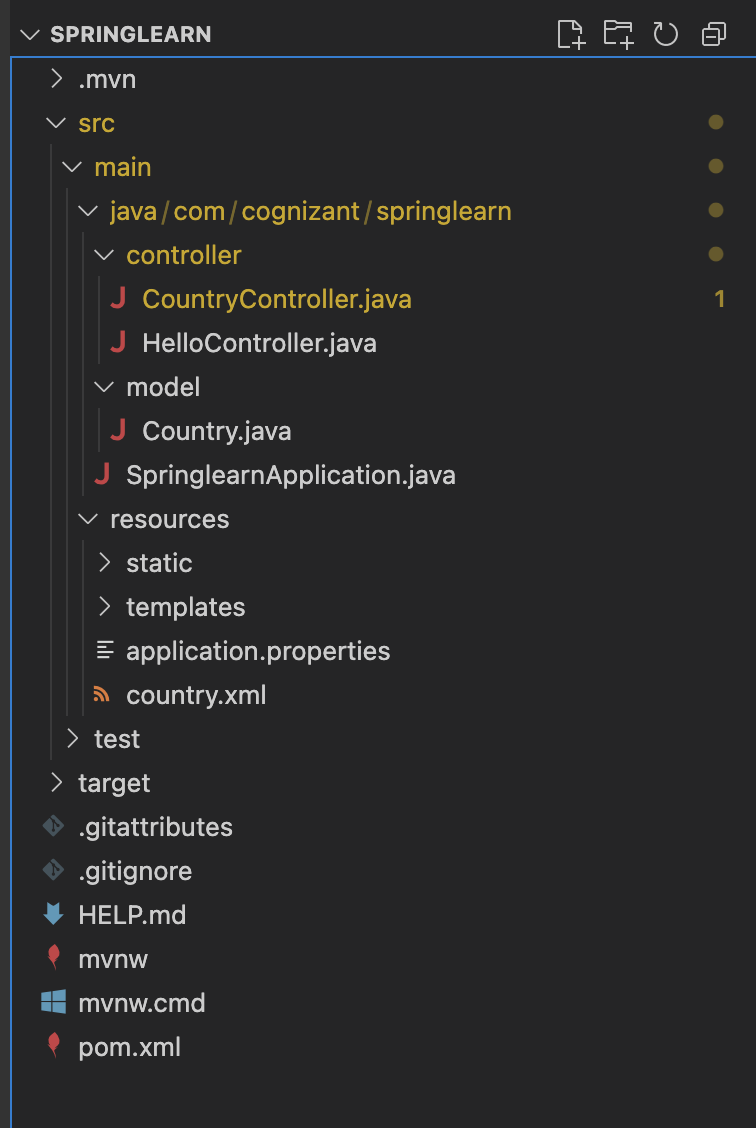
**Application.properties:**

server.port=8083

Output:



**Hands-on 4: REST - Country Web Service**



Country.java:

package com.cognizant.springlearn.model;

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 + "]";

}

}

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.springlearn.model.Country">

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

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

</bean>

</beans>

CountryController.java:

package com.cognizant.springlearn.controller;

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;

import com.cognizant.springlearn.model.Country;

@RestController

public class CountryController {

@RequestMapping("/country")

public Country getCountryIndia() {

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

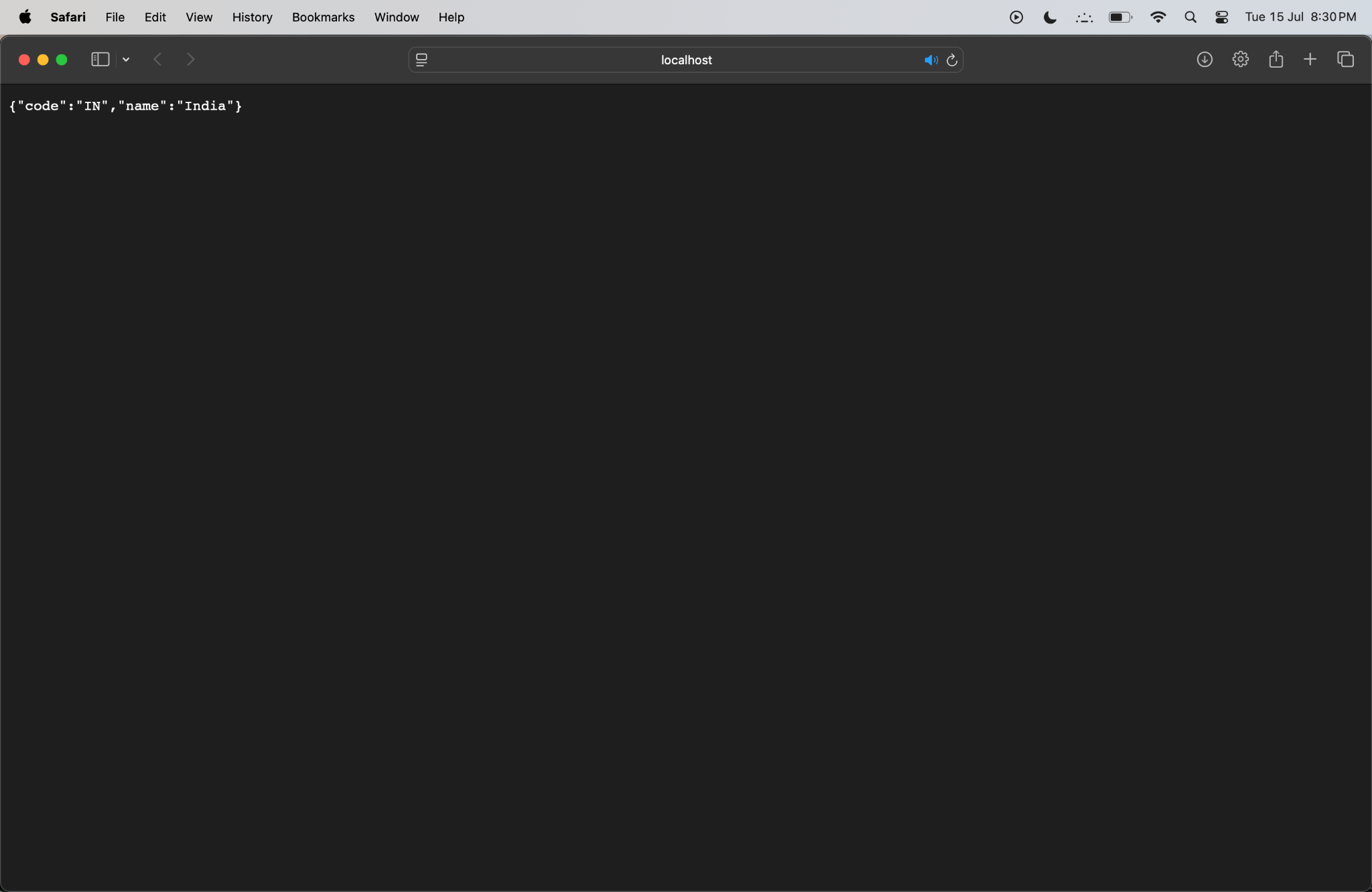
Country country = (Country) context.getBean("country");

return country;

}

}

Output:



**Hands-On 5: Get country based on country code**

**CountryService.java:**

package com.cognizant.springlearn.service;

import java.util.List;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import com.cognizant.springlearn.model.Country;

@Service

public class CountryService {

public Country getCountry(String code) {

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

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

return countries.stream()

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

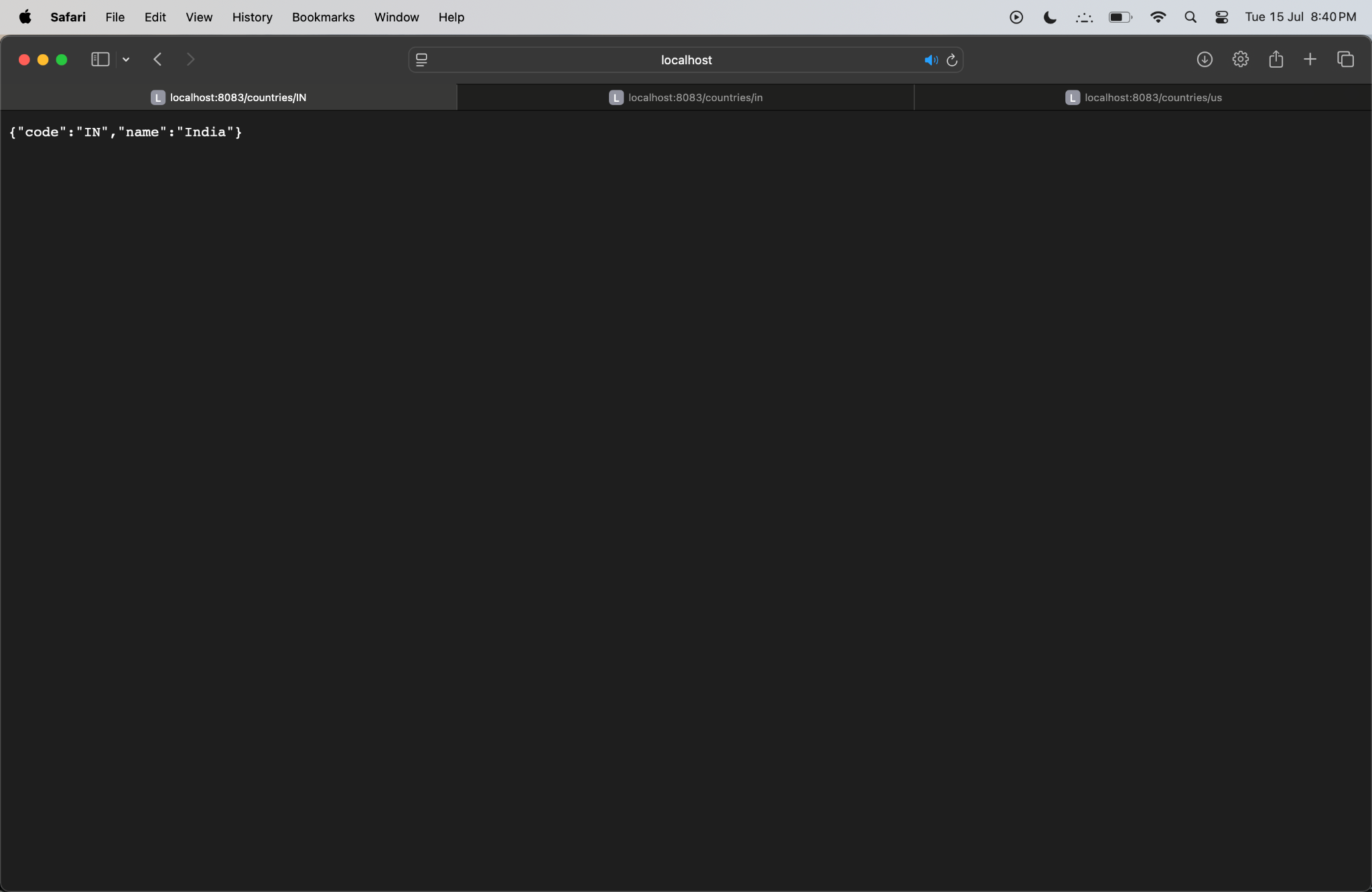
.findFirst()

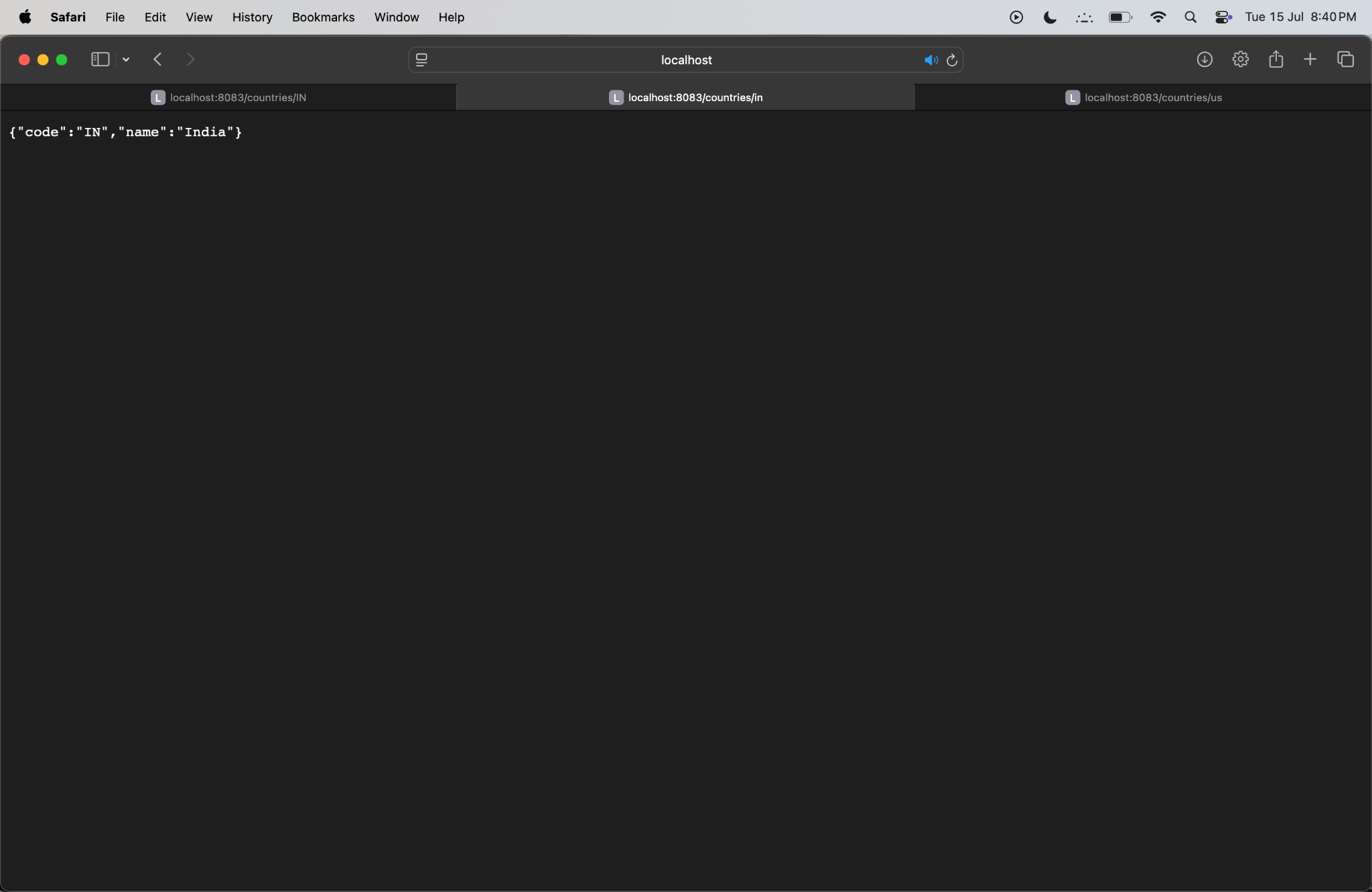
.orElse(null);

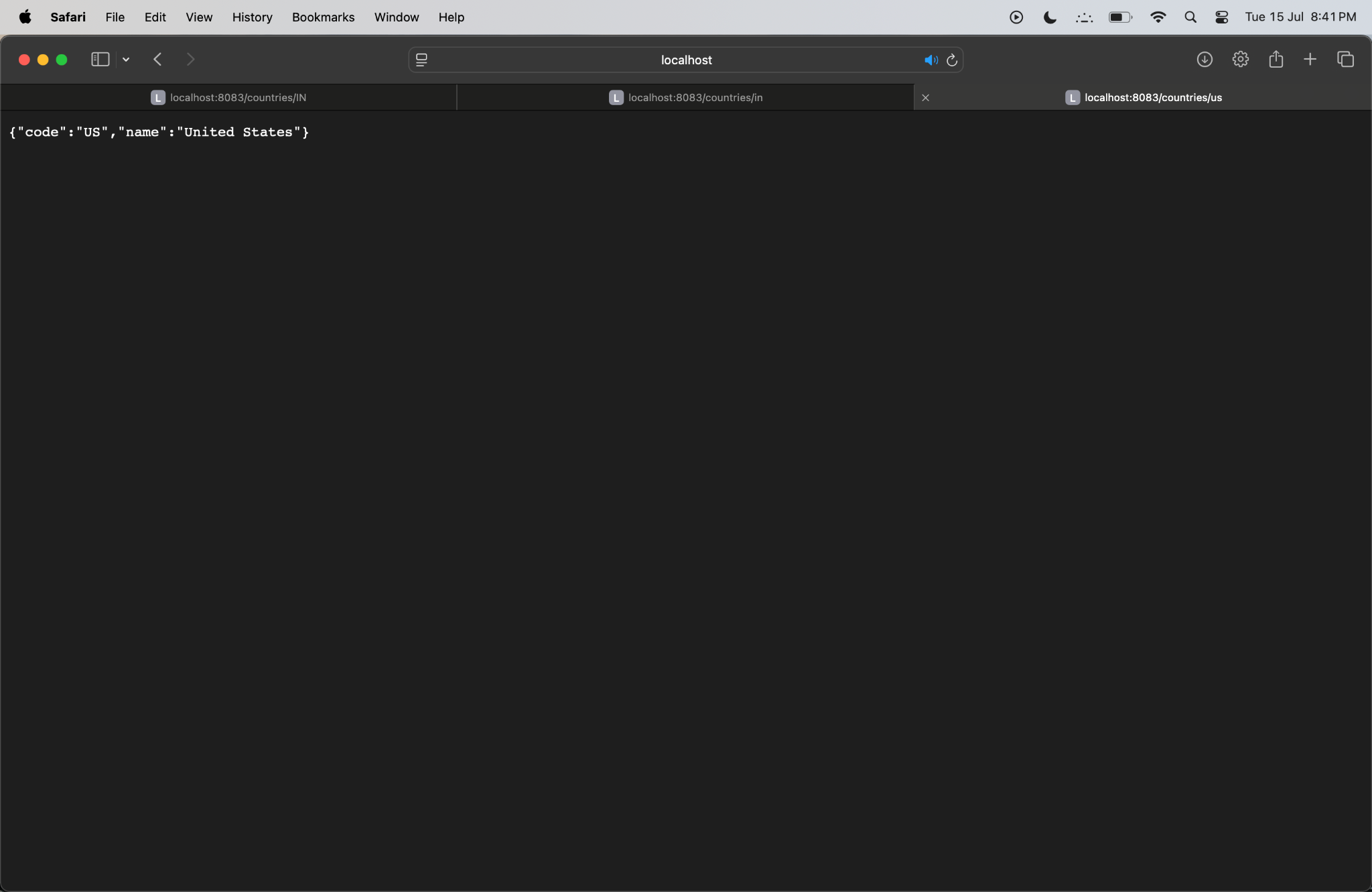
}

}

Output:







**Hands-on 6: Create authentication service that returns JWT**

**SecurityConfig.java:**

package com.cognizant.jwt.config;

import org.springframework.context.annotation.Bean;

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

import org.springframework.security.web.SecurityFilterChain;

import org.springframework.context.annotation.Configuration;

@Configuration

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

.authorizeHttpRequests(auth -> auth

.requestMatchers("/authenticate").authenticated() .anyRequest().permitAll()

)

.httpBasic() .and()

.csrf().disable(); return http.build();

}

}

**AuthController.java:**

package com.cognizant.jwt.controller;

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

import org.springframework.http.ResponseEntity;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.context.SecurityContextHolder;

@RestController

public class AuthController {

@GetMapping("/authenticate")

public ResponseEntity<?> authenticateUser() {

Authentication auth = SecurityContextHolder.getContext().getAuthentication();

String username = auth.getName();

String token = "token-for-user-" + username;

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

}

}

Application.properties:

server.port=8090

spring.security.user.name=user

spring.security.user.password=pwd

**Output:** 