**Hands On 1**

**HTTP Request Response**

**Sample HTTP Request Breakdown:**

POST /api/login HTTP/1.1

Host: www.example.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)

Content-Type: application/json

Content-Length: 45

Accept: application/json

{"username":"user1", "password":"pass123"}

**Line-by-Line Explanation:**

**Line 1:**

* POST → HTTP method (used to send data to server)
* /api/login → Resource or endpoint being accessed
* HTTP/1.1 → HTTP version used for the request

**Line 2: Host**

* www.example.com → Domain of the server being requested

**Line 3: User-Agent**

* Mozilla/5.0... → Information about the client/browser making the request

**Line 4: Content-Type**

* application/json → The format of data being sent in the request body

**Line 5: Content-Length**

* 45 → Number of bytes in the request body

**Line 6: Accept**

* application/json → The client expects the response in JSON format

**Final Line (Body)**

* {"username":"user1", "password":"pass123"} → Actual data being sent (JSON payload)

**Sample HTTP Response Breakdown:**

HTTP/1.1 200 OK

Date: Sat, 12 Jul 2025 09:30:00 GMT

Server: nginx/1.21.6

Content-Type: application/json

Content-Length: 57

Connection: keep-alive

{"status":"success", "message":"Login successful"}

**Line-by-Line Explanation:**

**Line 1:**

* HTTP/1.1 → HTTP version
* 200 → Status code (means success)
* OK → Status message

**Date:**

* Sat, 12 Jul 2025 09:30:00 GMT → When the response was generated

**Server:**

* nginx/1.21.6 → The type and version of the server

**Content-Type:**

* application/json → The response body is in JSON format

**Content-Length:**

* 57 → Number of bytes in the response body

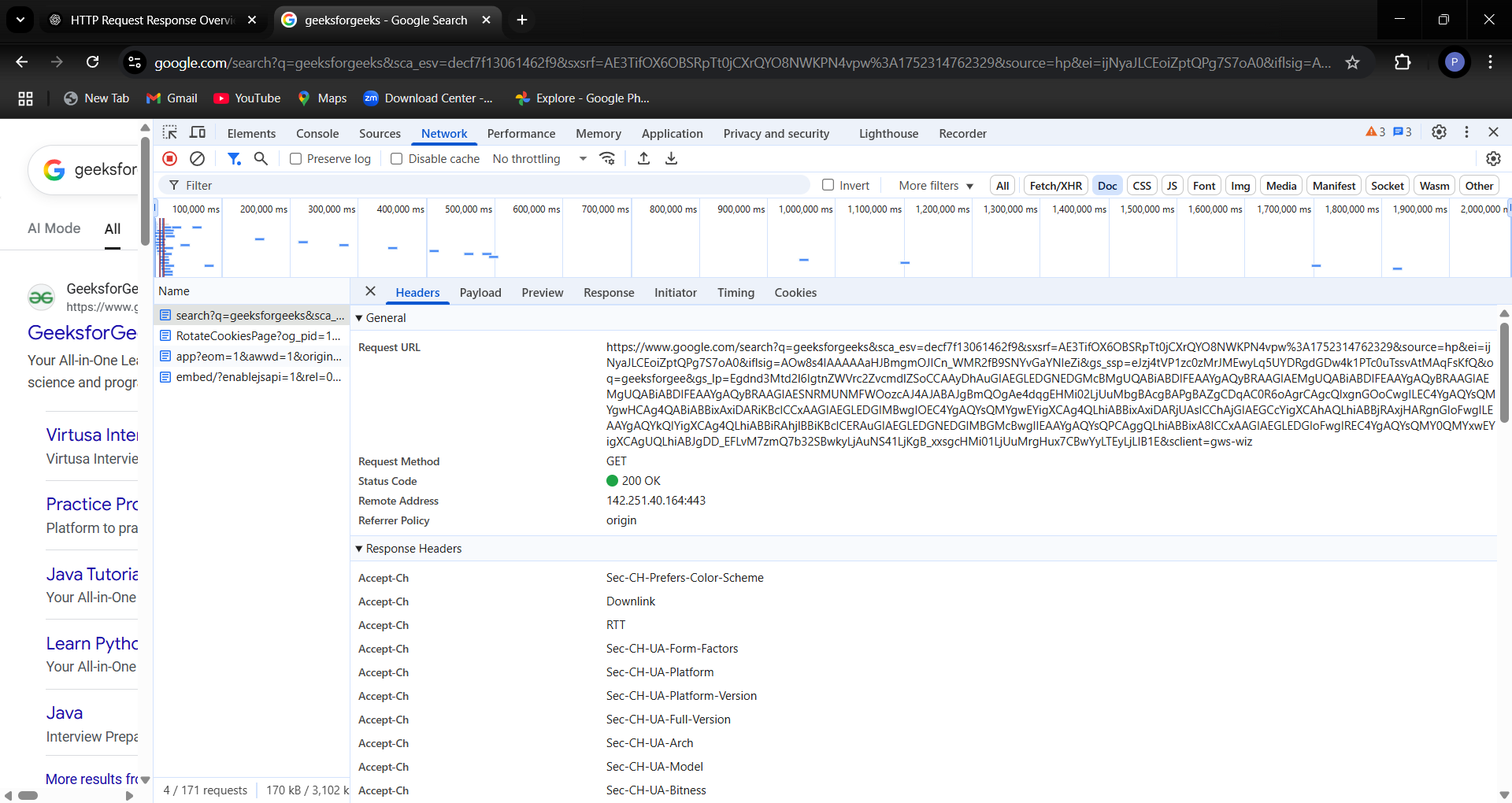
**Connection:**

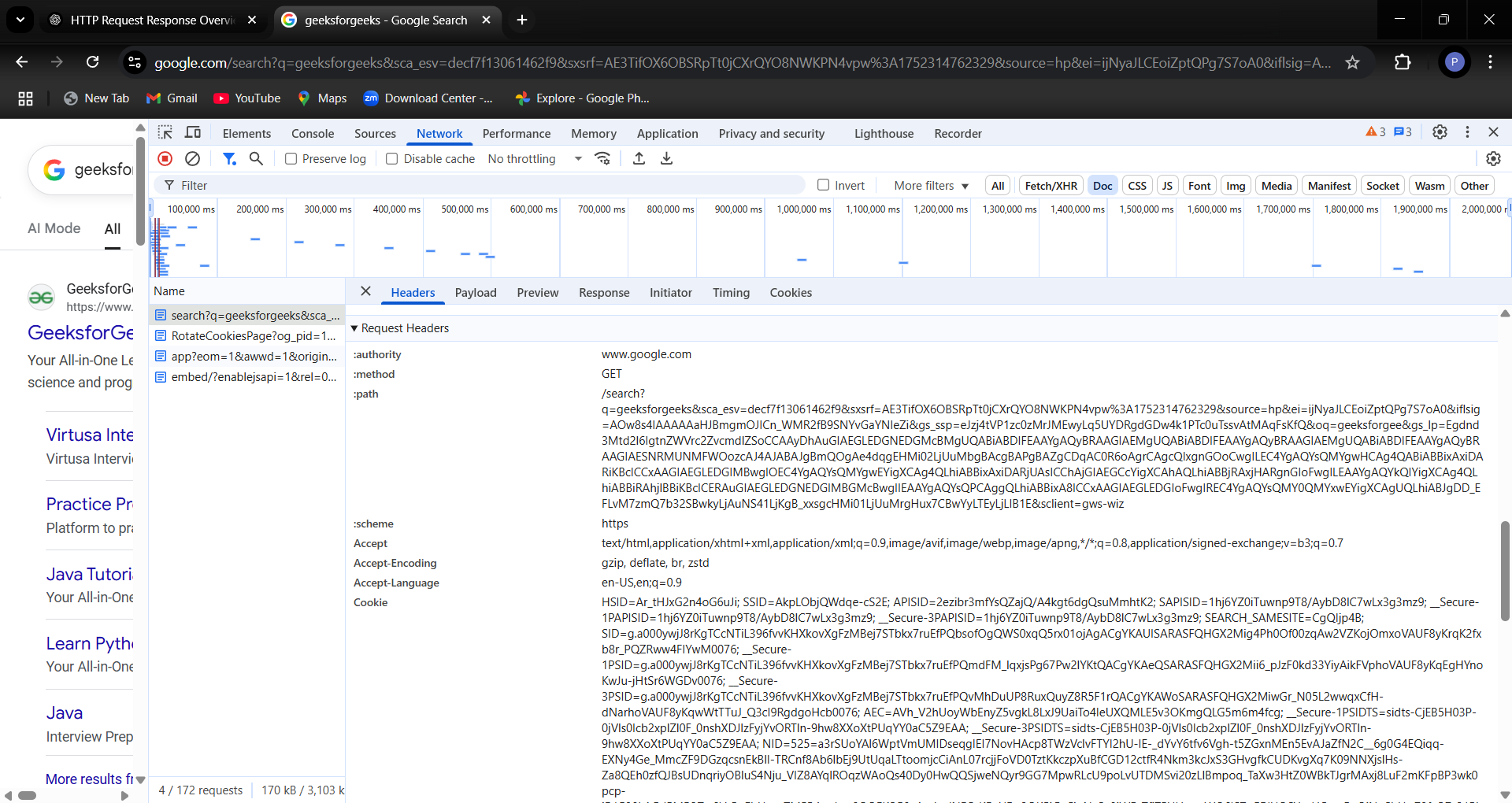
* keep-alive → Suggests that the connection remains open for further requests

**Final Line (Body):**

* {"status":"success", "message":"Login successful"} → Actual content returned by the server

**Request and response details in browser:**

****

****

**Hands On 2**

**Hello World RESTful Web Service**   
  
Write a REST service in the spring learn application created earlier, that returns the text "Hello World!!" using Spring Web Framework. Refer details below:  
  
**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!!

**Program:  
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</groupId>

<artifactId>spring-learn</artifactId>

<version>1.0.0</version>

<packaging>jar</packaging>

<parent>

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

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

<version>3.1.5</version>

<relativePath/> <!-- look up parent from Maven Central -->

</parent>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<!-- Web for REST APIs -->

<dependency>

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

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

</dependency>

<!-- Logging (already included in web starter, just explicit) -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

</dependency>

<!-- Optional: Spring DevTools for hot reload (remove if error continues) -->

<!--

<dependency>

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

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

-->

</dependencies>

<build>

<plugins>

<!-- Use Spring Boot Maven Plugin -->

<plugin>

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

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

<version>3.1.5</version>

</plugin>

</plugins>

</build>

</project>

**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*.info("START");

String message = "Hello World!!";

*LOGGER*.info("END");

return message;

}

}

**SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

*LOGGER*.info("START");

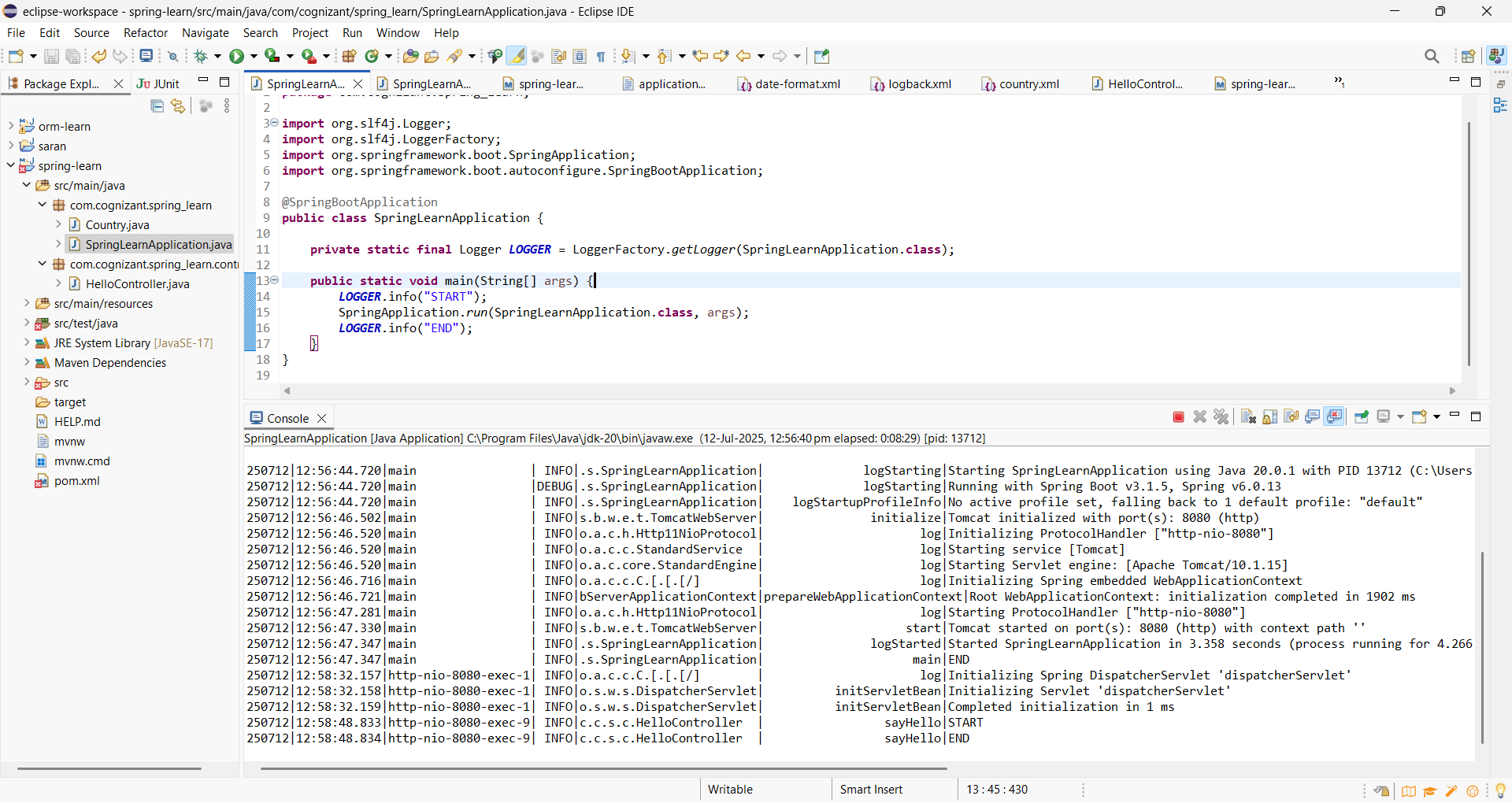
SpringApplication.*run*(SpringLearnApplication.class, args);

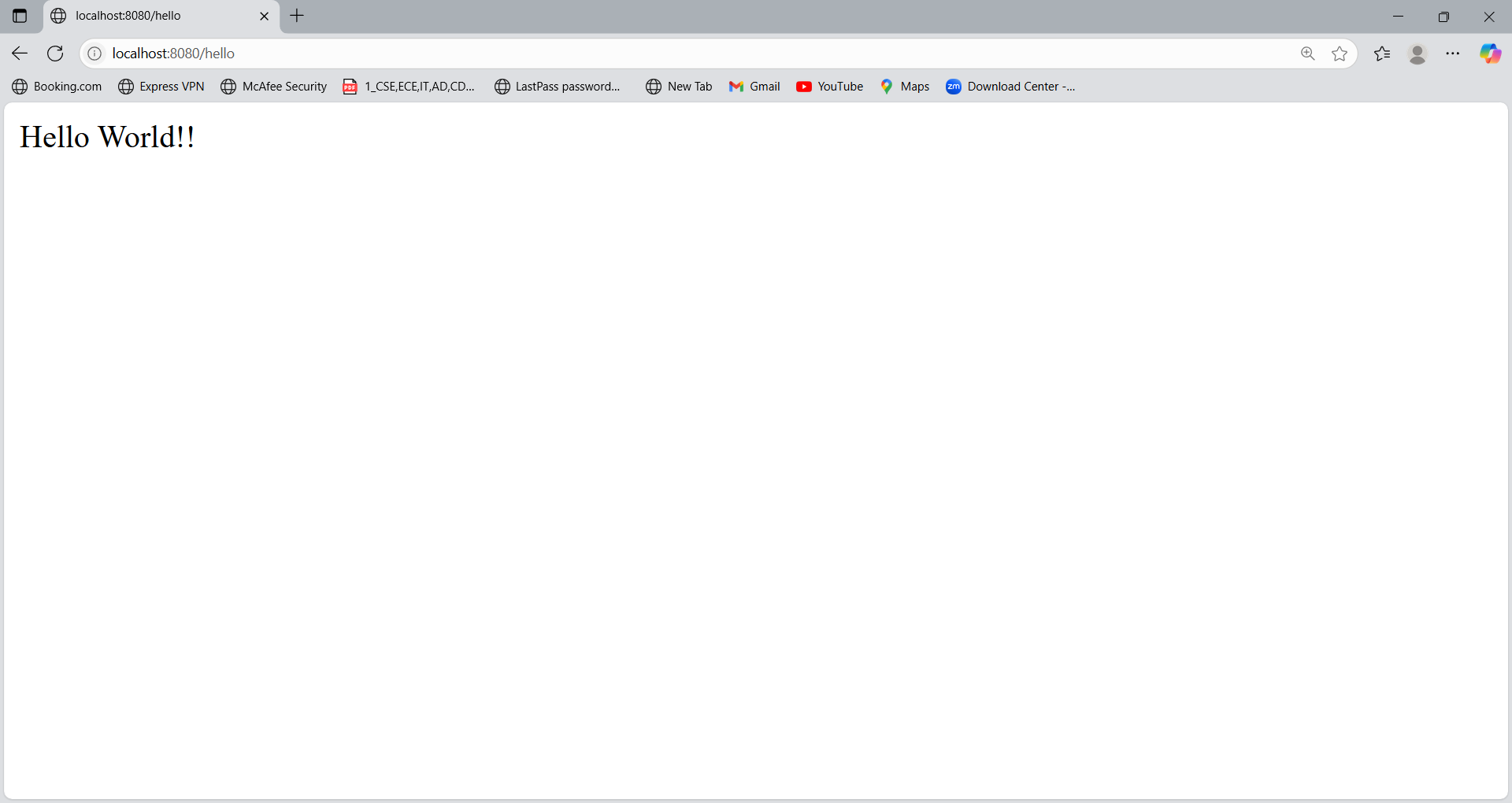
*LOGGER*.info("END");

}

}

**Output:**

****

****

**Hands On 3**

**REST - Country Web Service**Write a REST service that returns India country details in the earlier created spring learn application.  
  
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": "IN",

  "name": "India"

}

* **What happens in the controller method?**
* @RequestMapping("/country") maps the URL http://localhost:8083/country to this method.
* When the endpoint is hit (e.g., from browser/Postman), the method getCountryIndia() is called.
* Inside this method:
* ApplicationContext loads the Spring configuration from country.xml.
* It fetches the bean with id="in" which is an instance of Country (configured in XML).
* The bean is returned directly as the method's response**.**
* **How the bean is converted into JSON reponse?**
* Spring Boot includes the **Jackson** library (com.fasterxml.jackson.databind) by default.
* Jackson uses **reflection** to inspect the Country class and convert its fields into a JSON object.

**For example:**

Country country = new Country();

country.setCode("IN");

country.setName("India");

**Jackson converts it to:**

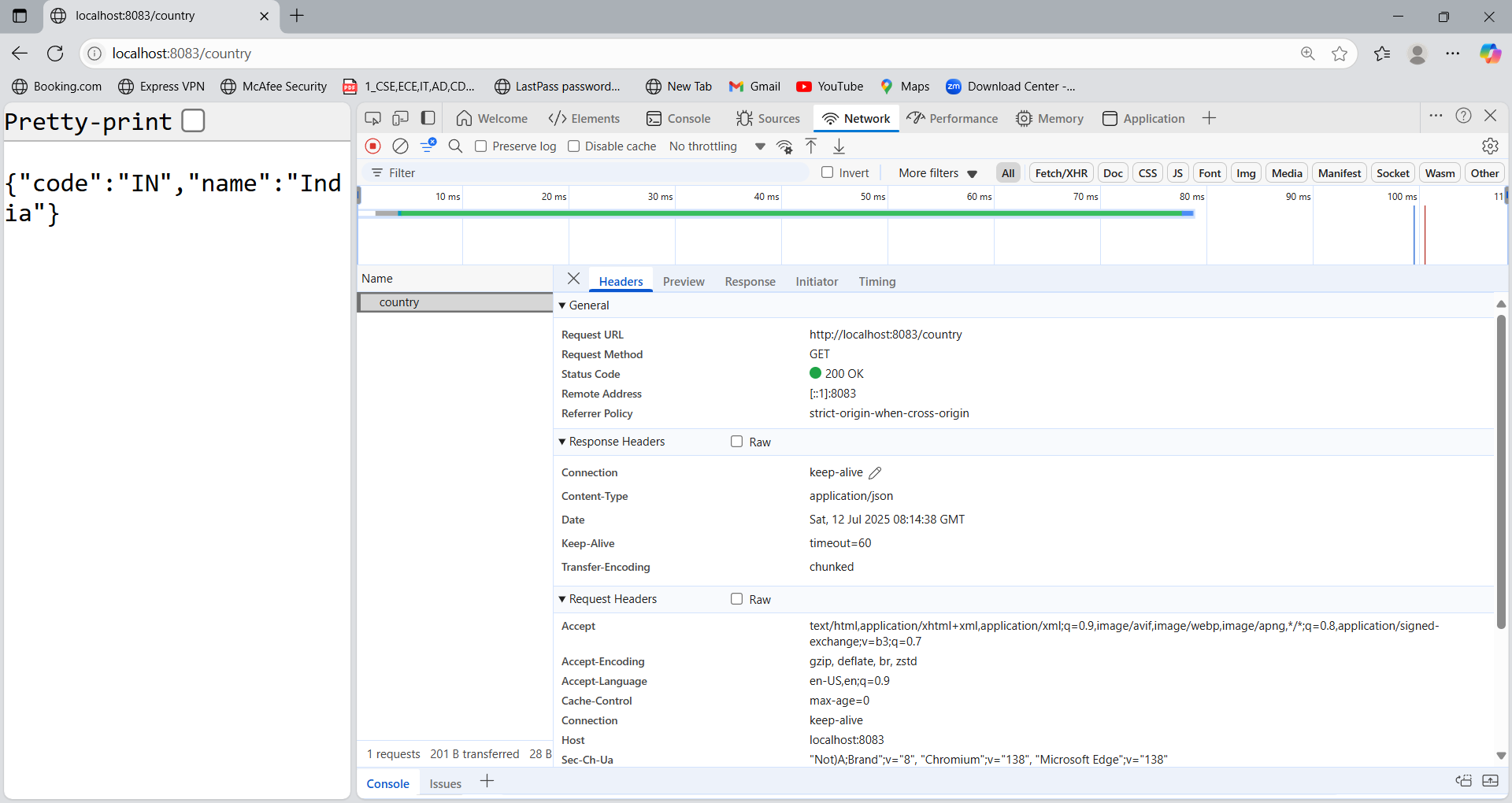
{

"code": "IN",

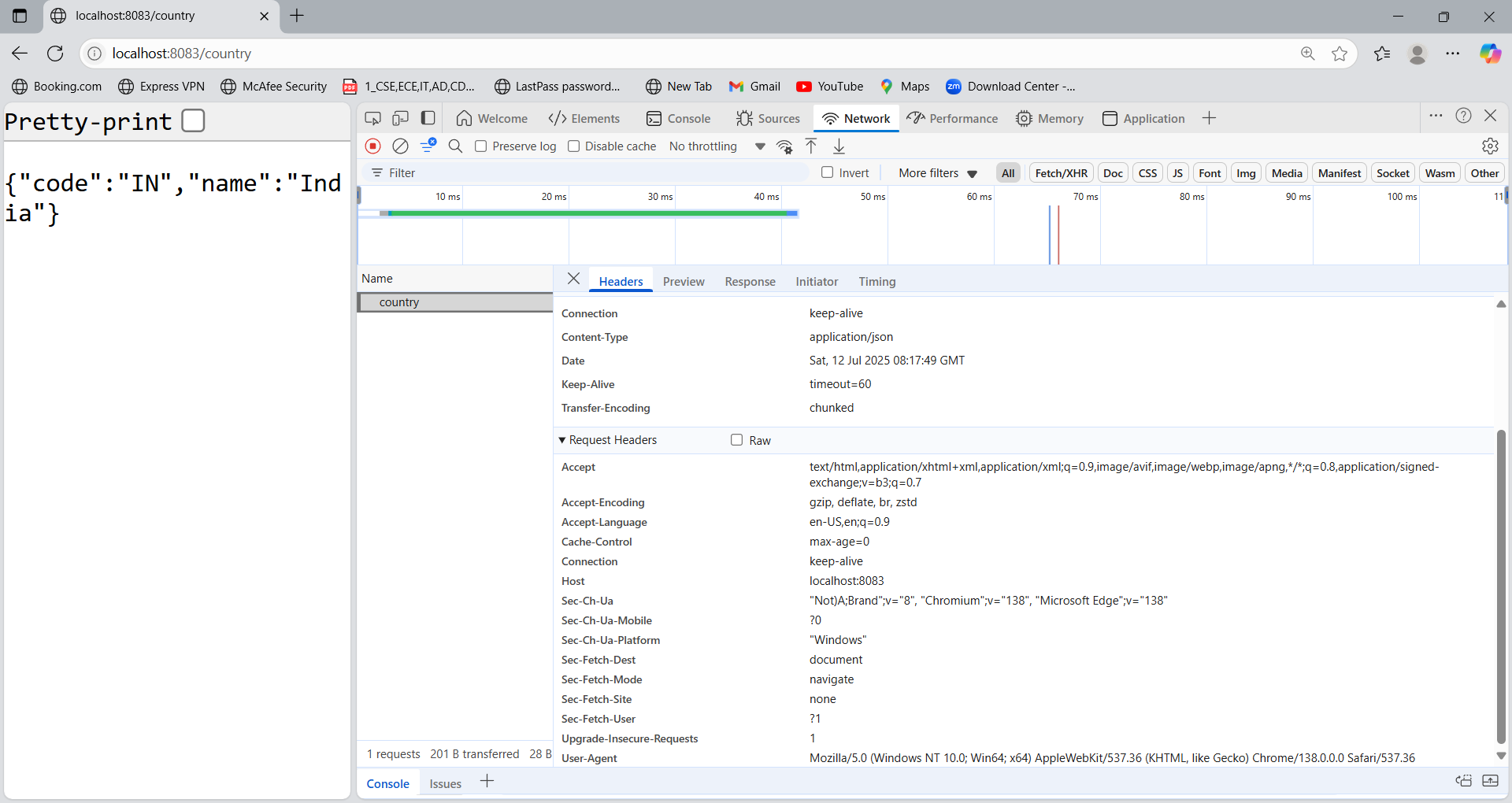
"name": "India"

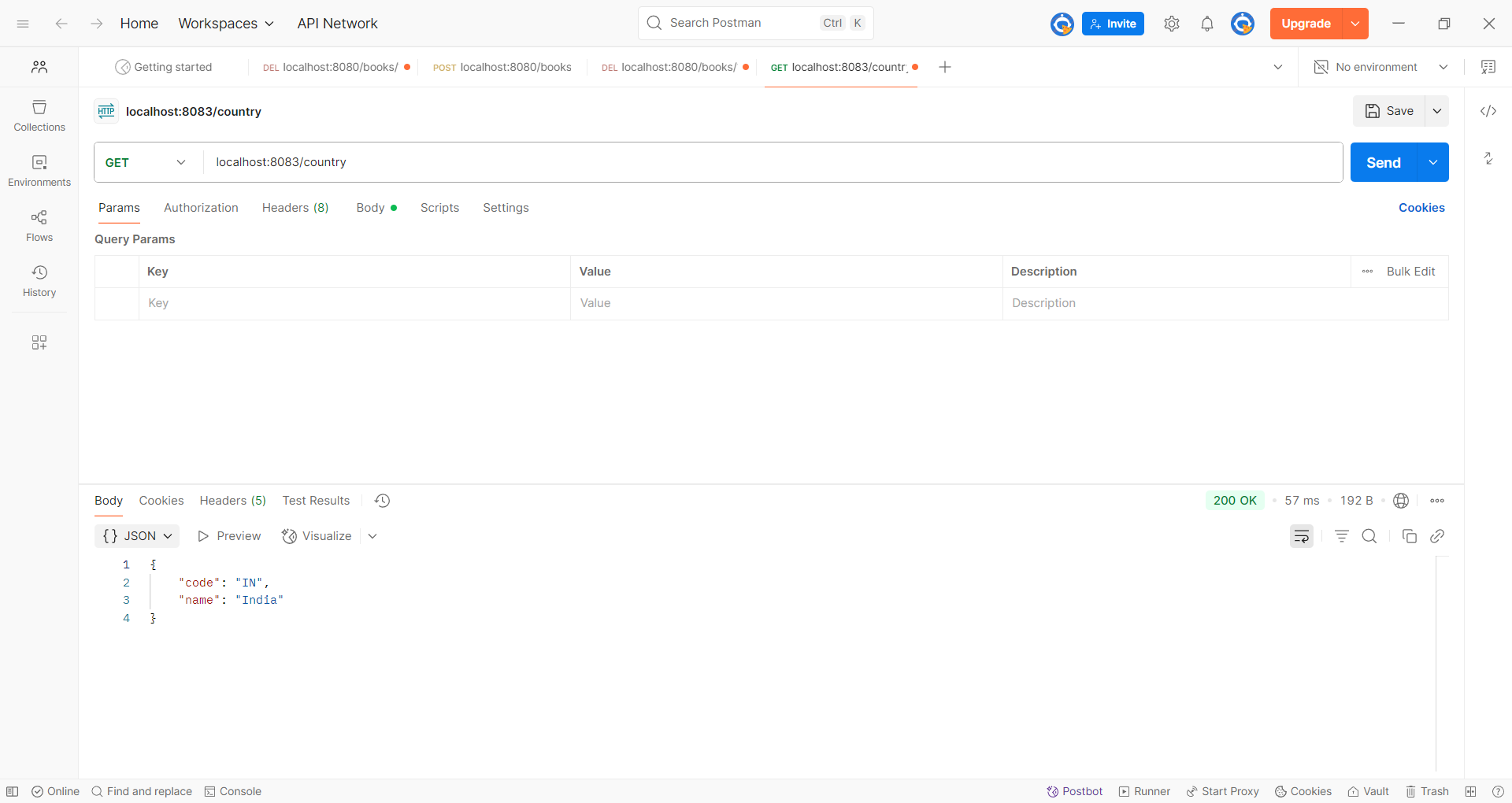
}

* **In network tab of developer tools show the HTTP header details received**

****

* **In postman click on "Headers" tab to view the HTTP header details received**

****

****

**Program:**

**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</groupId>

<artifactId>spring-learn</artifactId>

<version>1.0.0</version>

<packaging>jar</packaging>

<parent>

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

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

<version>3.1.5</version>

<relativePath/> <!-- look up parent from Maven Central -->

</parent>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<!-- Web for REST APIs -->

<dependency>

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

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

</dependency>

<!-- Logging (already included in web starter, just explicit) -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

</dependency>

<!-- Optional: Spring DevTools for hot reload (remove if error continues) -->

<!--

<dependency>

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

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

-->

</dependencies>

<build>

<plugins>

<!-- Use Spring Boot Maven Plugin -->

<plugin>

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

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

<version>3.1.5</version>

</plugin>

</plugins>

</build>

</project>

**country.xml**

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

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

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

</bean>

</beans>

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

*LOGGER*.info("END");

return country;

}

}

**SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

*LOGGER*.info("START");

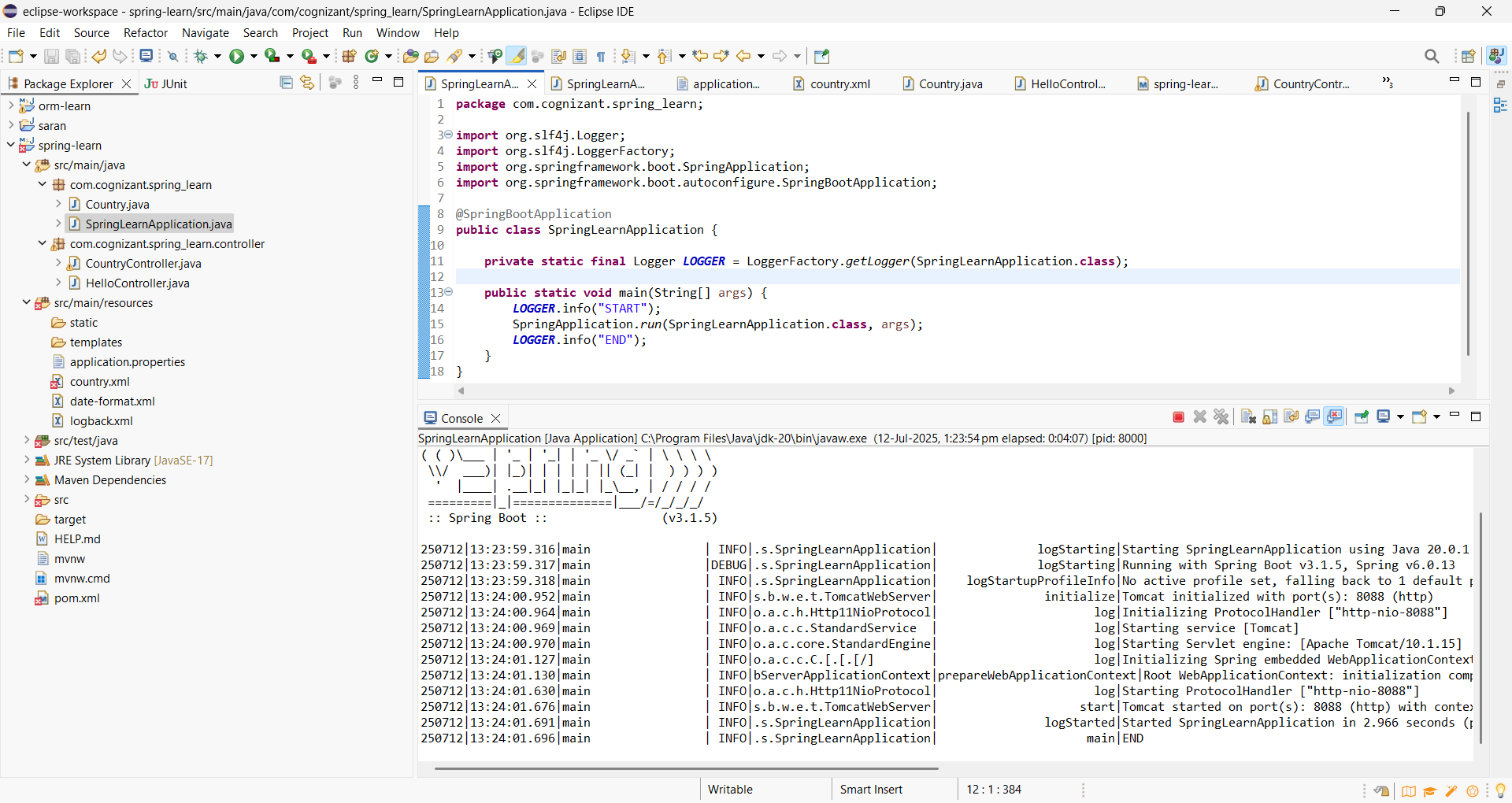
SpringApplication.*run*(SpringLearnApplication.class, args);

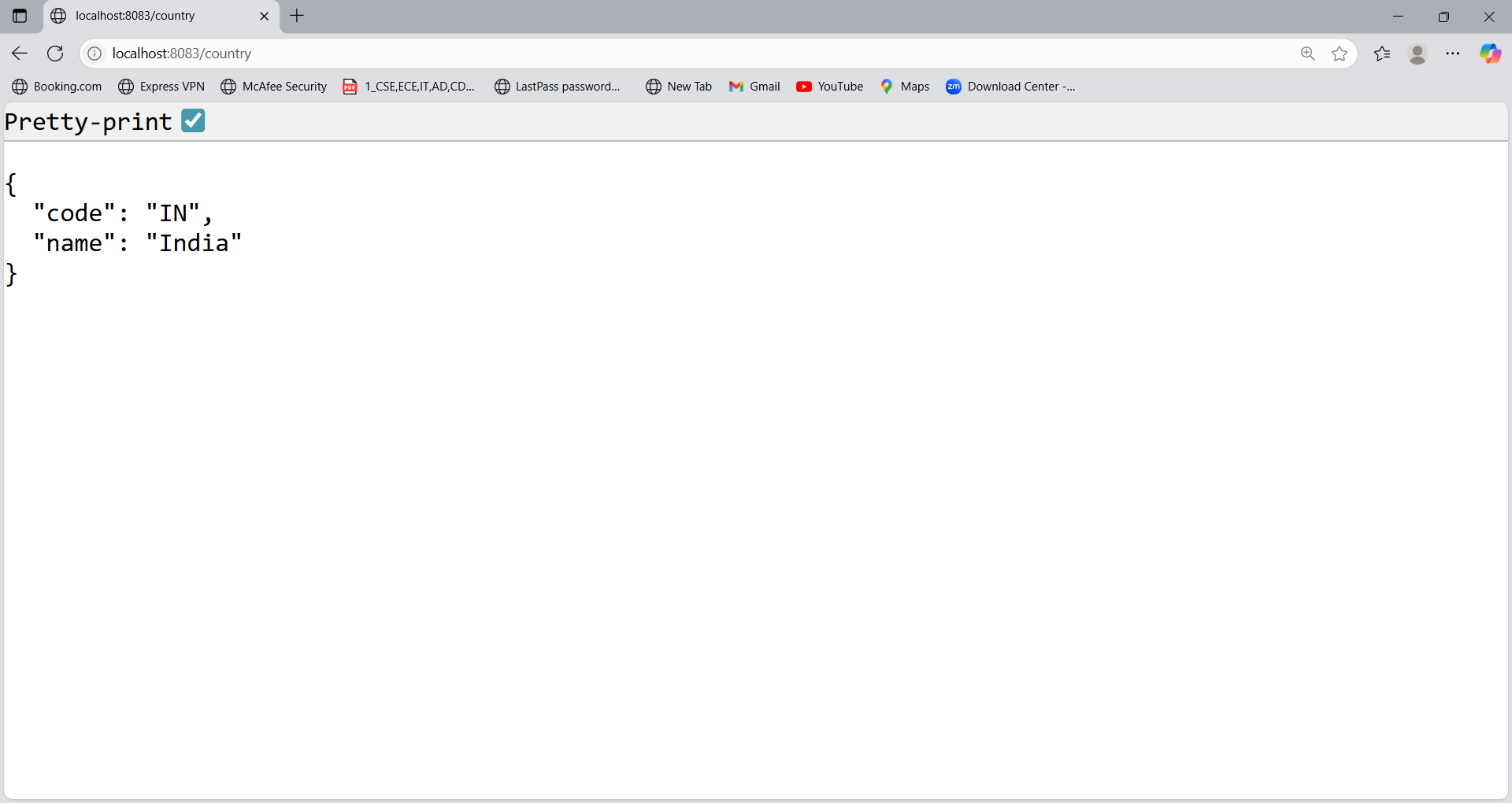
*LOGGER*.info("END");

}

}

**Output:**

****

****

**Hands On 4**

**REST - Get all countries**Write a REST service that returns all the countries.  
  
**Controller**: com.cognizant.spring-learn.controller.CountryController **Method Annotation**: @GetMapping("/countries")  
**Method Name**: getAllCountries()  
**Method Implementation:** Load country list from country.xml and return  
 **Sample Request**: http://localhost:8083/countries **Sample Response:**

[

  { "code": "IN", "name": "India"},

  { "code": "US", "name": "United States"},

  { "code": "JP", "name": "Japan"},

  { "code": "DE", "name": "Germany"}

]

**Program:**

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

<!-- List of all countries -->

<bean id="countryList" class="java.util.ArrayList">

<constructor-arg>

<list>

<ref bean="in"/>

<ref bean="us"/>

<ref bean="jp"/>

<ref bean="de"/>

</list>

</constructor-arg>

</bean>

</beans>

**Country.java**

package com.cognizant.spring\_learn;

public class Country {

private String code;

private String name;

public Country() {}

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;

}

}

**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.GetMapping;

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

import java.util.List;

@RestController

public class CountryController {

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

@GetMapping("/countries")

public List<Country> getAllCountries() {

LOGGER.info("START");

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

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

LOGGER.info("END");

return countryList;

}

}

**SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

*LOGGER*.info("START");

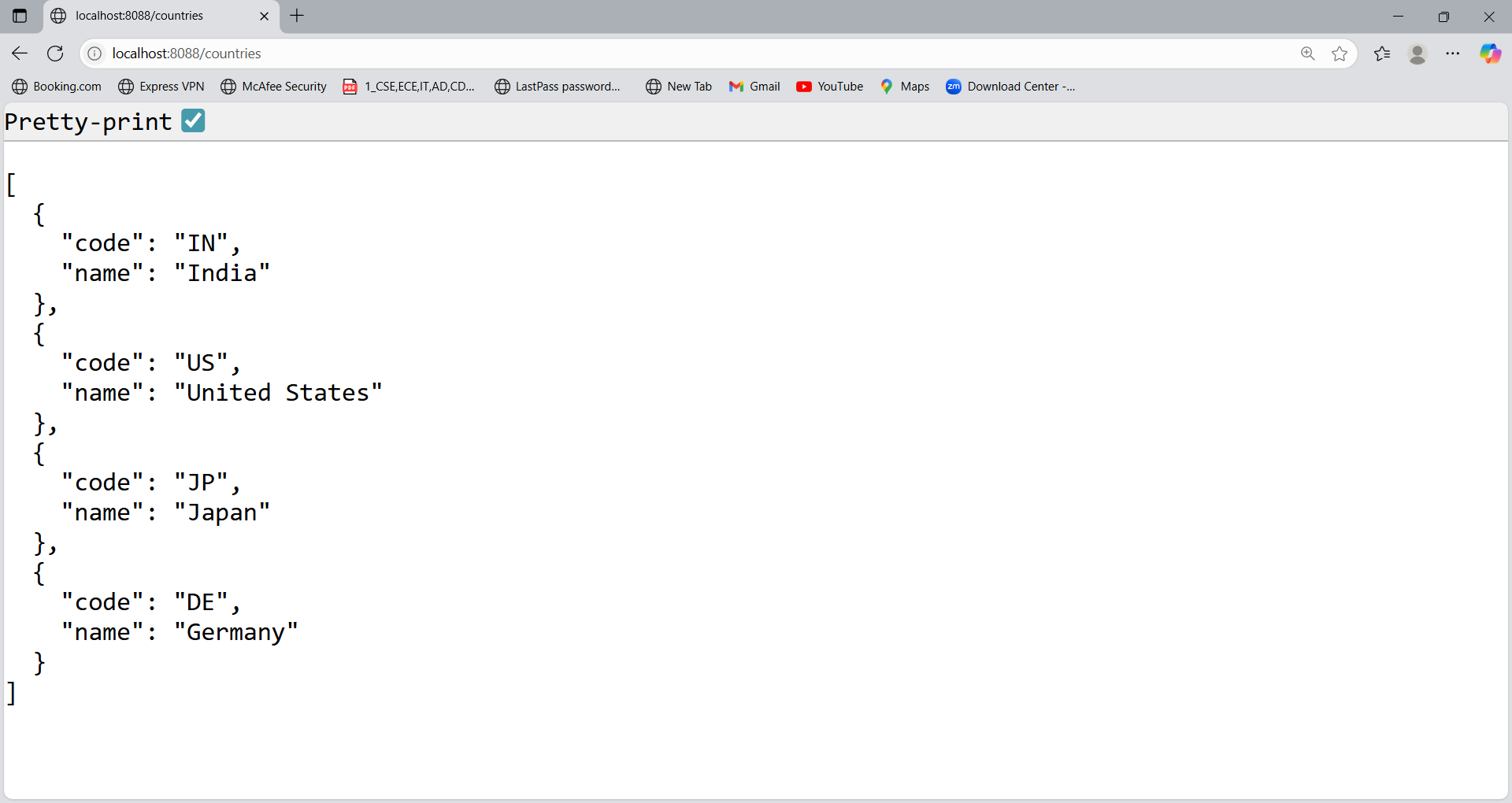
SpringApplication.*run*(SpringLearnApplication.class, args);

*LOGGER*.info("END");

}

}

**Output:**

****

**Hands On 5**

**REST - Get country based on country code**Write a REST service that returns a specific country based on country code. The country code should be case insensitive.  
  
**Controller:** com.cognizant.spring-learn.controller.CountryController **Method Annotation:**@GetMapping("/countries/{code}")  
**Method Name:** getCountry(String code)  
**Method Implemetation:** Invoke countryService.getCountry(code)**Service Method:**com.cognizant.spring-learn.service.CountryService.getCountry(String code) **Service Method Implementation:**

* Get the country code using @PathVariable
* Get country list from country.xml
* Iterate through the country list
* Make a case insensitive matching of country code and return the country.
* Lambda expression can also be used instead of iterating the country list

**Sample Request**: http://localhost:8083/country/in  
  
**Sample Response:**

{

  "code": "IN",

  "name": "India"

}

**Program:**

**Country.java**

package com.cognizant.spring\_learn;

public class Country {

private String code;

private String name;

public Country() {}

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;

}

}

**CountryController.java**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.Country;

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

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

@RestController

public class CountryController {

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

@Autowired

private CountryService countryService;

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

public Country getCountry(@PathVariable String code) {

LOGGER.info("START");

Country country = countryService.getCountry(code);

LOGGER.info("END");

return country;

}

}

**CountryService.java**

package com.cognizant.spring\_learn.service;

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.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

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

public Country getCountry(String code) {

LOGGER.info("START");

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

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

// Using Lambda expression for case-insensitive match

Country result = countryList.stream()

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

.findFirst()

.orElse(null);

LOGGER.info("END");

return result;

}

}

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

<!-- List of all countries -->

<bean id="countryList" class="java.util.ArrayList">

<constructor-arg>

<list>

<ref bean="in"/>

<ref bean="us"/>

<ref bean="jp"/>

<ref bean="de"/>

</list>

</constructor-arg>

</bean>

</beans>

**SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

*LOGGER*.info("START");

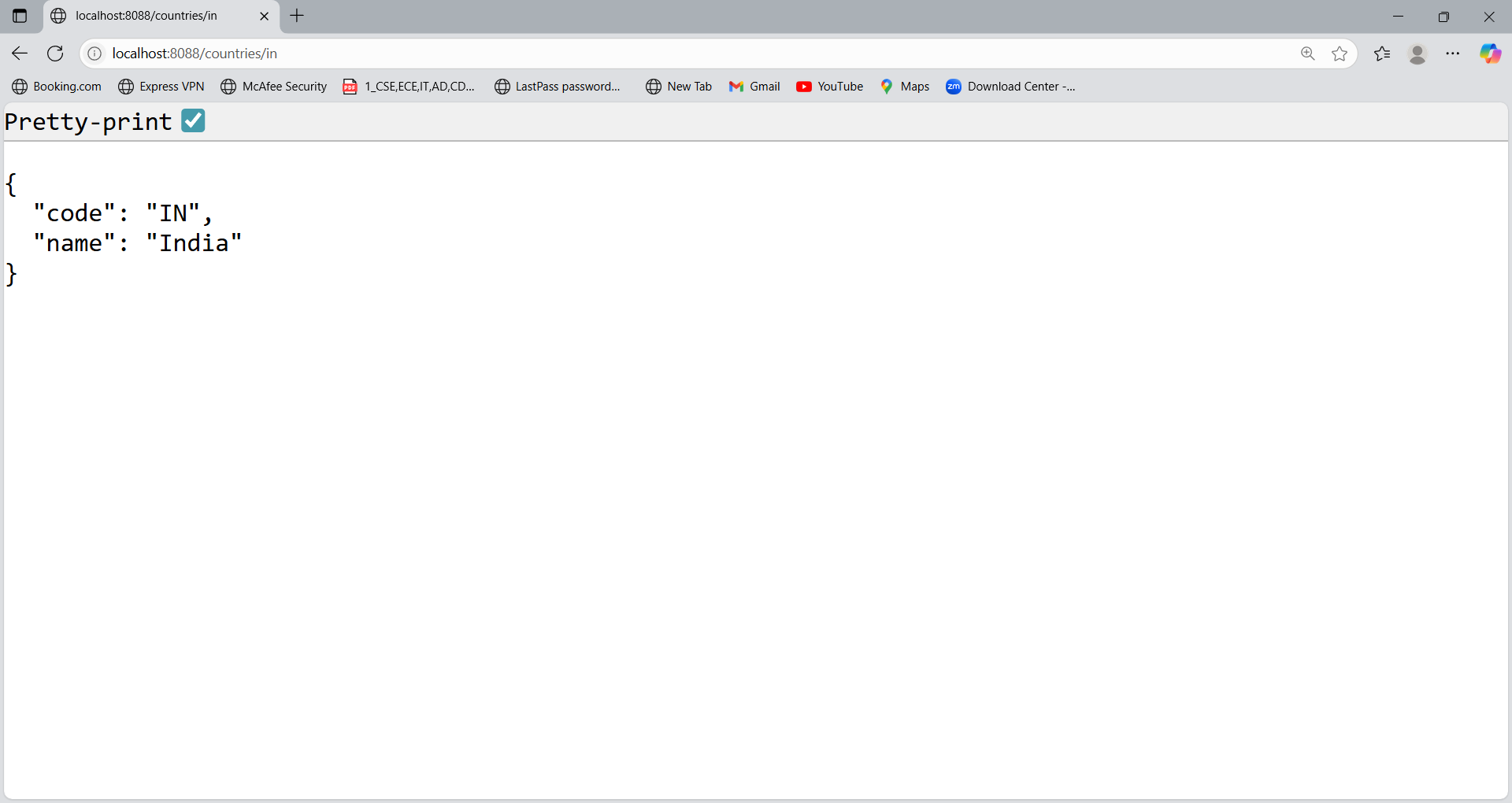
SpringApplication.*run*(SpringLearnApplication.class, args);

*LOGGER*.info("END");

}

}

**Output:**

****

**Hands On 6**

**REST - Get country exceptional scenario**In the previous hands on where we implemented getting country based on country code, what happens if the country code provided in the URL is not present.  
  
**Refer steps below to implement**

* Create a new exception class com.cognizant.springlearn.service.exception.CountryNotFoundException
* Include below specified annotation at the class level in CountryNotFoundException class

**@ResponseStatus(value = HttpStatus.NOT\_FOUND, reason = "Country not found")**

* In CountryService.getCountry() method include the logic to throw CountryNotFoundException if the country code does not exists in the list.
* In CountryController.getCountry() method include throws clause in method signature. This will respond to the caller of the web service with appropriate error message in JSON format.
* Test the service in postman and using curl command. Refer below for executing curl command.

**Steps to invoke RESTful Web Service using curl command**

* Open Git Bash
* Execute the below command

**curl -i http://localhost:8090/country/az**

**Sample Request:** http://localhost:8083/country/az  
 **Sample Response:**

{

"timestamp": "2019-10-02T03:27:54.521+0000",

"status": 404,

"error": "Not Found",

"message": "Country not found",

"path": "/country/az"

}

**Program:**

**CountryNotFoundException.java**

package com.cognizant.spring\_learn.service.exception;

import org.springframework.http.HttpStatus;

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

@ResponseStatus(value = HttpStatus.NOT\_FOUND, reason = "Country not found")

public class CountryNotFoundException extends Exception {

public CountryNotFoundException() {

super("Country not found");

}

}

**CountryService.java**

package com.cognizant.spring\_learn.service;

import com.cognizant.spring\_learn.Country;

import com.cognizant.spring\_learn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

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

public Country getCountry(String code) throws CountryNotFoundException {

LOGGER.info("START");

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

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

Country result = countryList.stream()

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

.findFirst()

.orElseThrow(CountryNotFoundException::new);

LOGGER.info("END");

return result;

}

}

**CountryController.java**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.Country;

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

import com.cognizant.spring\_learn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

@RestController

public class CountryController {

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

@Autowired

private CountryService countryService;

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

public Country getCountry(@PathVariable String code) throws CountryNotFoundException {

LOGGER.info("START");

Country country = countryService.getCountry(code);

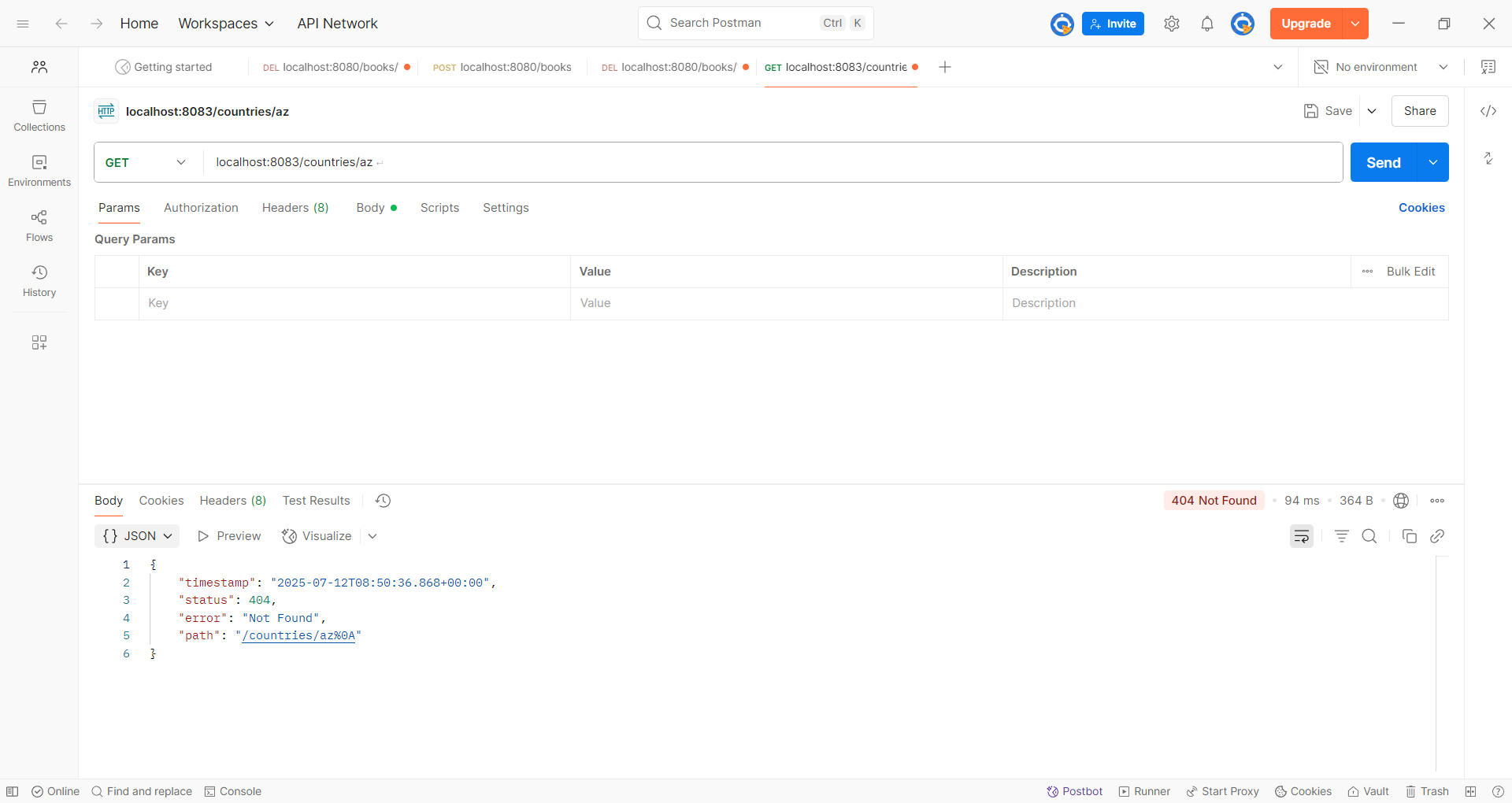
LOGGER.info("END");

return country;

}

}

**Output:**

****

**Hands On 7**

**MockMVC - Test get country service**Using MockMVC test the get country service.  
 **Create a test cases to test the following aspects:**

* Test is the CountryController is loaded
* Invoke the service to get country and check in the response if it contains code as "IN" and name as "India"

**Program:**

**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</groupId>

<artifactId>spring-learn</artifactId>

<version>1.0.0</version>

<packaging>jar</packaging>

<parent>

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

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

<version>3.1.5</version>

<relativePath/> <!-- look up parent from Maven Central -->

</parent>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<!-- Web for REST APIs -->

<dependency>

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

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

</dependency>

<!-- Logging (already included in web starter, just explicit) -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<!-- Optional: Spring DevTools for hot reload (remove if error continues) -->

<!--

<dependency>

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

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

-->

</dependencies>

<build>

<plugins>

<!-- Use Spring Boot Maven Plugin -->

<plugin>

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

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

<version>3.1.5</version>

</plugin>

</plugins>

</build>

</project>

**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.GetMapping;

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

@RestController

public class CountryController {

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

@GetMapping("/country")

public Country getCountryIndia() {

LOGGER.info("START");

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

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

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

LOGGER.info("END");

return country;

}

}

**SpringLearnApplicationTest.java**

package com.cognizant.spring\_learn;

import com.cognizant.spring\_learn.controller.CountryController;

import org.junit.jupiter.api.Test;

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

import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;

import org.springframework.boot.test.context.SpringBootTest;

import org.springframework.test.web.servlet.MockMvc;

import org.springframework.test.web.servlet.ResultActions;

import static org.junit.jupiter.api.Assertions.assertNotNull;

import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;

import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.\*;

@SpringBootTest

@AutoConfigureMockMvc

public class SpringLearnApplicationTests {

@Autowired

private CountryController countryController;

@Autowired

private MockMvc mvc;

// Test if controller is loaded

@Test

public void contextLoads() {

assertNotNull(countryController);

}

// Test /country endpoint

@Test

public void testGetCountry() throws Exception {

ResultActions actions = mvc.perform(get("/country"));

actions.andExpect(status().isOk());

actions.andExpect(jsonPath("$.code").exists());

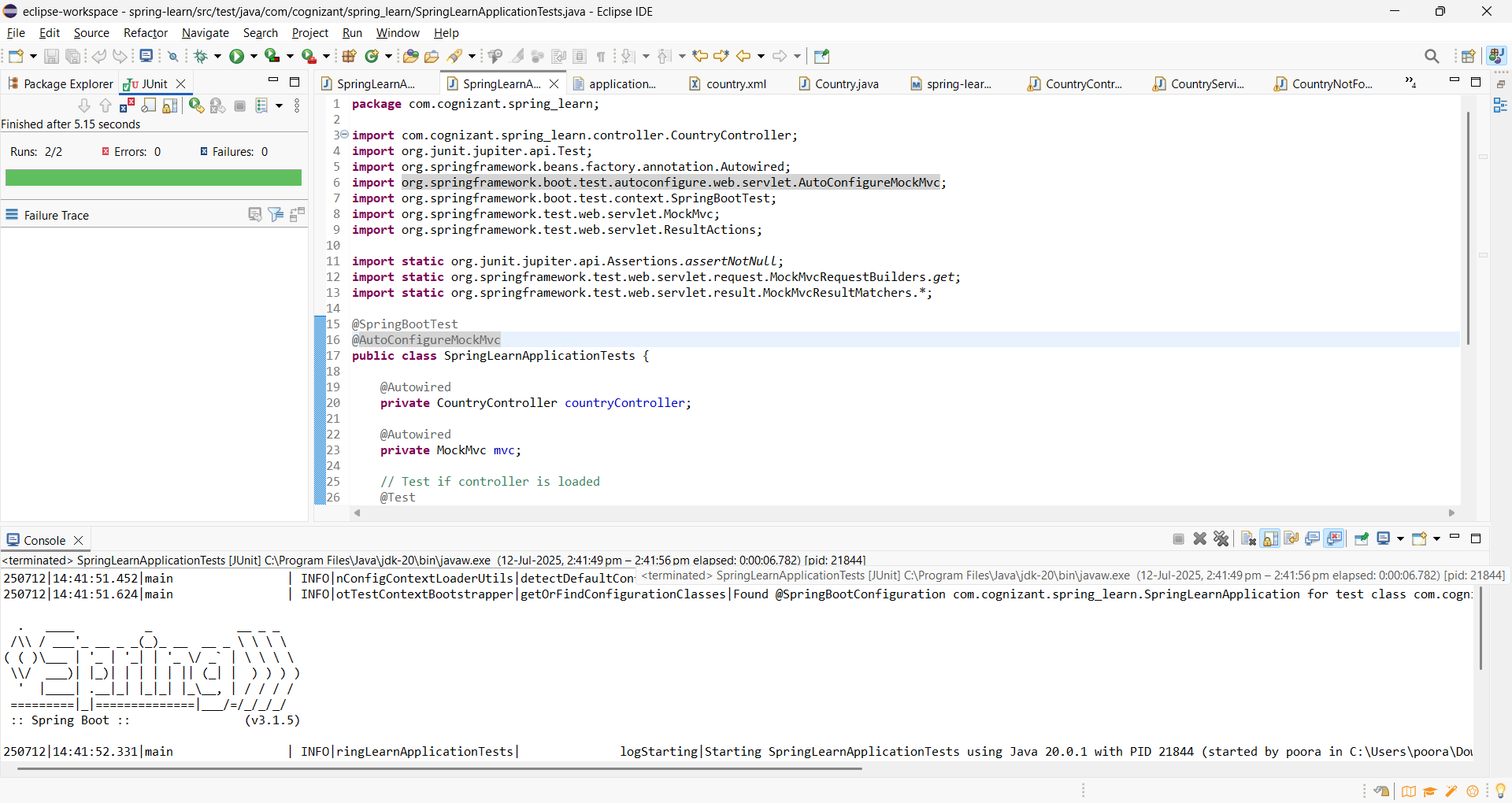
actions.andExpect(jsonPath("$.code").value("IN"));

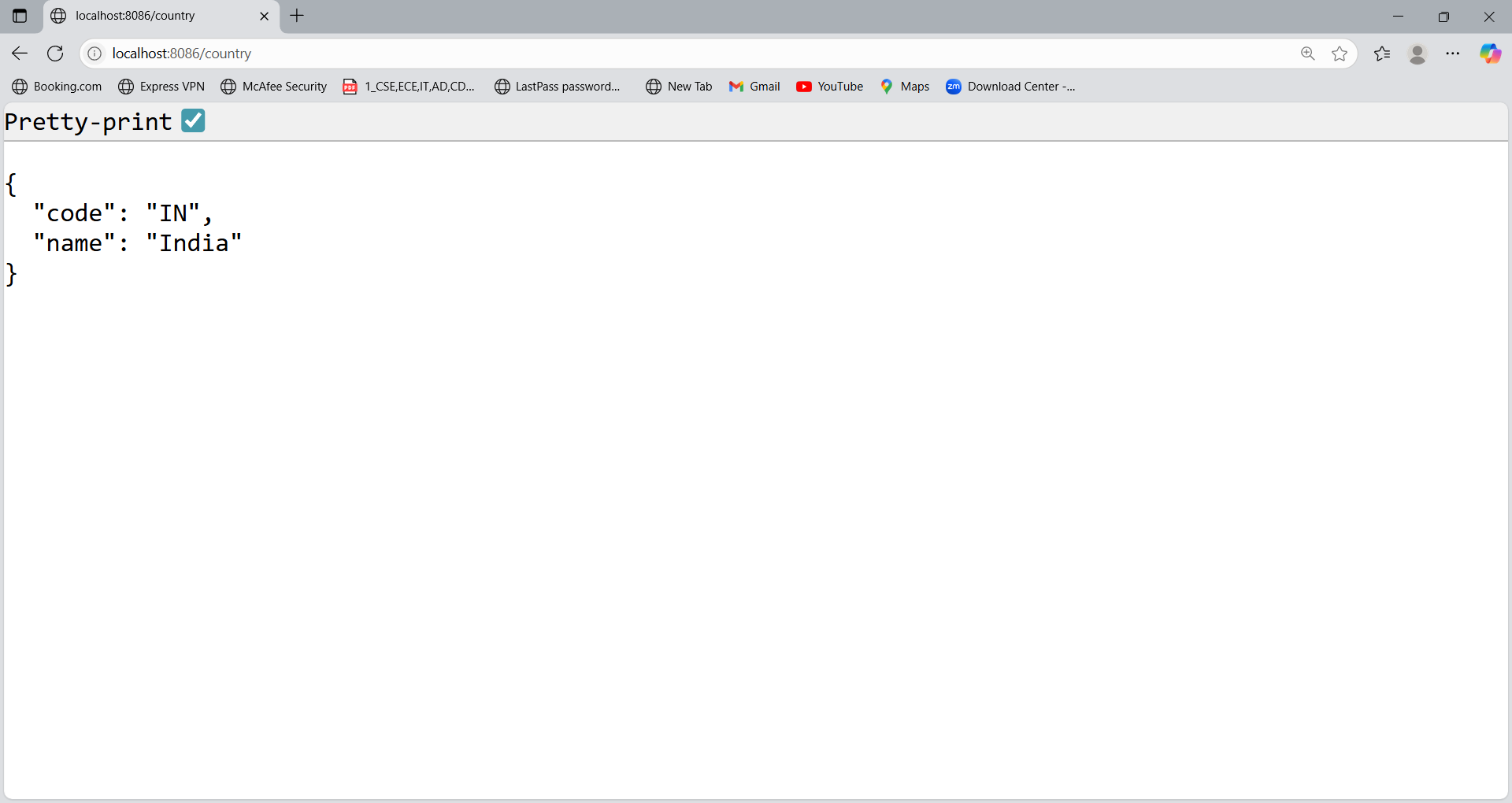
actions.andExpect(jsonPath("$.name").exists());

actions.andExpect(jsonPath("$.name").value("India"));

}}

**Output:**

****

****

**Hands On 8**

**MockMVC - Test get country service for exceptional scenario**Include MockMVC test that checks if correct response is received when there is an error.

**Program:  
CountryNotFoundException.java**

package com.cognizant.spring\_learn.exception;

public class CountryNotFoundException extends Exception {

public CountryNotFoundException(String message) {

super(message);

}

}

**CountryController.java**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.exception.CountryNotFoundException;

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.\*;

@RestController

public class CountryController {

@Autowired

private CountryService countryService;

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

public Country getCountry(@PathVariable String code) throws CountryNotFoundException {

return countryService.getCountry(code);

}

}

**GlobalExceptionHandler.java**

package com.cognizant.spring\_learn.exception;

import org.springframework.http.HttpStatus;

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

@RestControllerAdvice

public class GlobalExceptionHandler {

@ExceptionHandler(value = CountryNotFoundException.class)

@ResponseStatus(HttpStatus.BAD\_REQUEST)

public String handleCountryNotFoundException(CountryNotFoundException ex) {

return ex.getMessage();

}

}

**CountryService.java**

package com.cognizant.spring\_learn.service;

import com.cognizant.spring\_learn.exception.CountryNotFoundException;

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

import org.springframework.stereotype.Service;

import java.util.HashMap;

import java.util.Map;

@Service

public class CountryService {

private static Map<String, Country> countryMap = new HashMap<>();

static {

countryMap.put("IN", new Country("IN", "India"));

countryMap.put("US", new Country("US", "United States"));

}

public Country getCountry(String code) throws CountryNotFoundException {

Country country = countryMap.get(code.toUpperCase());

if (country == null) {

throw new CountryNotFoundException("Country Not found");

}

return country;

}

}

**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;

}

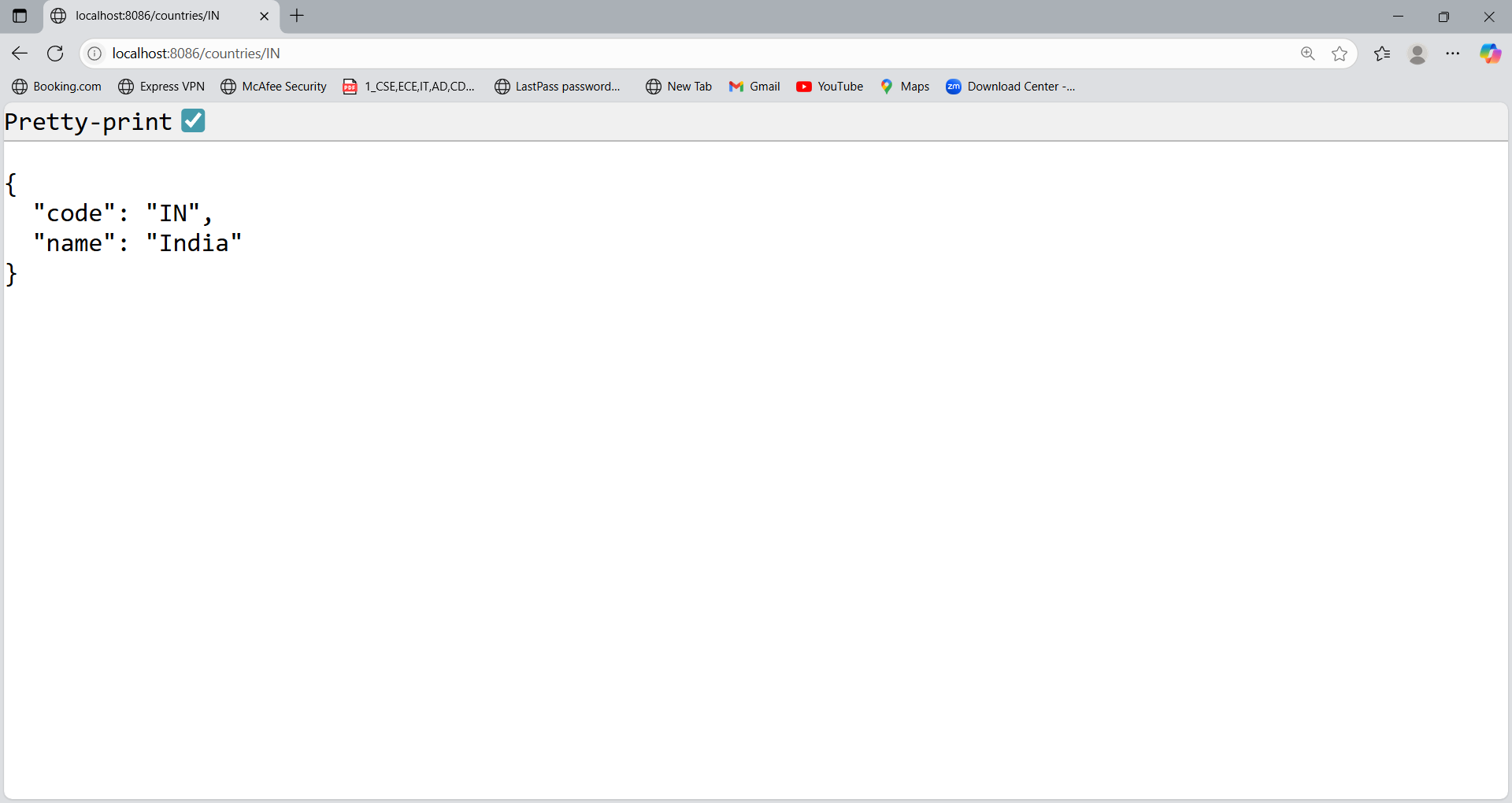
public void setName(String name) {

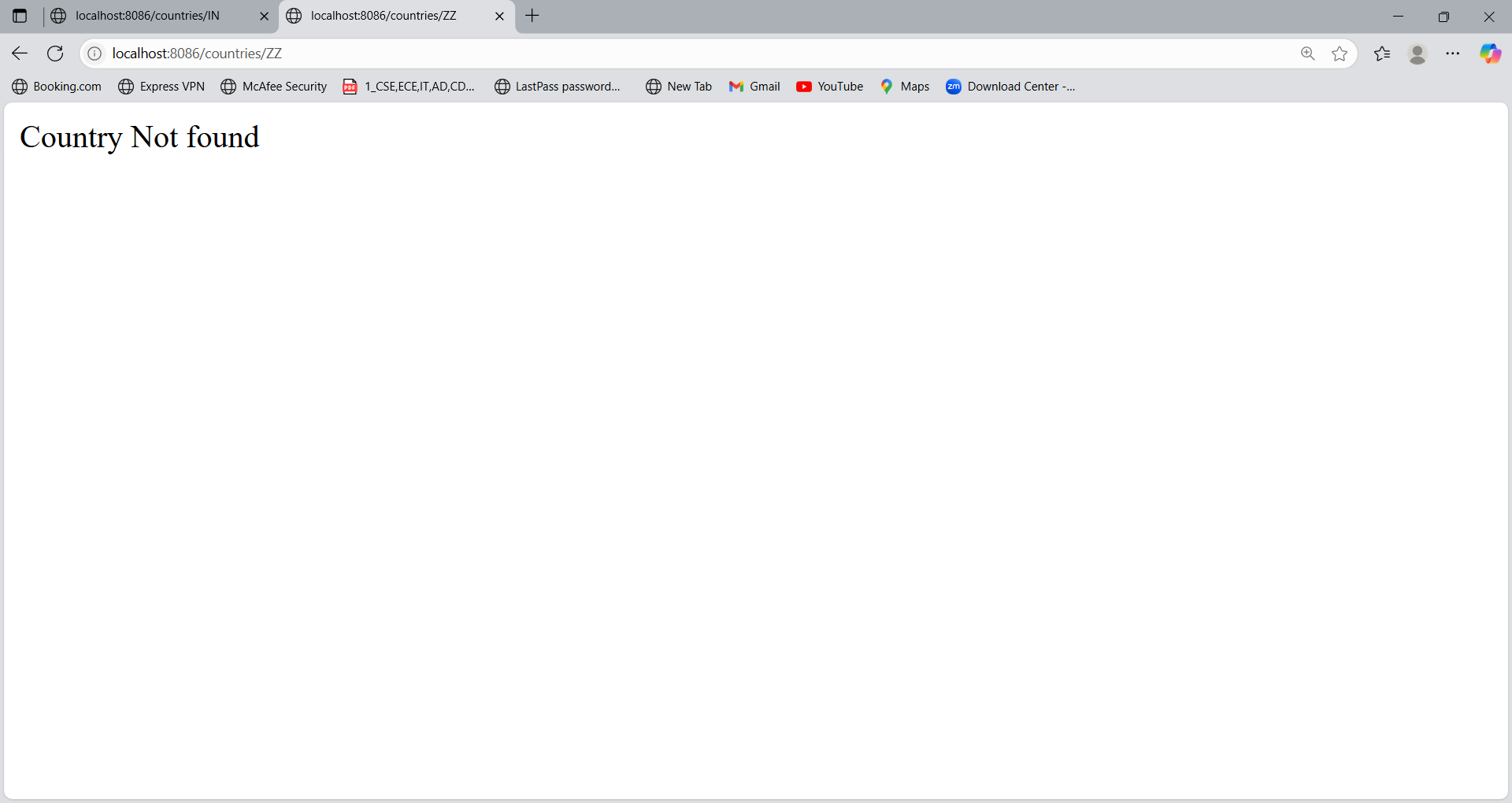
this.name = name;

}

}

**Output:**

****

****