**Objectives**

* Explain in detail about HTTP Request and Response
  + HTTP Request and Response, HTTP Request Format, HTTP Response Format, Request URL, Request Method, Content-Type, User-Agent
    - Ref - https://en.wikipedia.org/wiki/Hypertext\_Transfer\_Protocol

* Explain the need and benefits of RESTful Web Services
  + REST stands for REpresentational State Transfer, lightweight, maintainable, scalable, underlying protocol is HTTP; composed of resources, verbs, header, body, response status code, client-server technology
    - Ref - https://www.chakray.com/advantages-of-rest-api/

* Demonstrate implementation of RESTful Web Service using GET method
  + @RestController, @GetMapping, invoking get request from browser, invoking get request from postman, bean transformation to JSON, get method with parameter, return array, @PathVariable
    - Dispatcher Servlet - https://docs.spring.io/spring/docs/5.1.9.RELEASE/spring-framework-reference/web.html#mvc-servlet
    - Spring REST (Getting Started) - https://spring.io/guides/gs/rest-service/
    - Request Mapping - https://docs.spring.io/spring/docs/5.1.9.RELEASE/spring-framework-reference/web.html#mvc-ann-requestmapping

* Demonstrate implementation of end to end testing of RESTful Web Service using MockMVC
  + @AutoConfigureMockMvc, MockMvc, @Test, get(), perform(), andExpect(), status().isOk(), jsonPath().exists, jsonPath().value(), status().isBadRequest(), status().reason test execution in Eclipse, test execution in command line using maven
    - Server Side Testing - https://docs.spring.io/spring/docs/5.1.9.RELEASE/spring-framework-reference/testing.html#spring-mvc-test-server

**HTTP Request Response**   
  
**Sample HTTP Request Breakdown:**

GET /hello.txt HTTP/1.1

User-Agent: curl/7.16.3 libcurl/7.16.3 OpenSSL/0.9.7l zlib/1.2.3

Host: www.example.com

Accept-Language: en, mi

**Line-by-Line Explanation:**

* **Line 1**:
  + GET → HTTP method
  + /hello.txt → Resource (path of the file or endpoint)
  + HTTP/1.1 → HTTP version
* **Line 2**: User-Agent → Info about the client making the request (e.g., browser, curl)
* **Line 3**: Host → Domain name of the server
* **Line 4**: Accept-Language → Preferred languages client can handle

**Sample HTTP Response Breakdown:**

HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT

Server: Apache

Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT

ETag: "34aa387-d-1568eb00"

Accept-Ranges: bytes

Content-Length: 51

Vary: Accept-Encoding

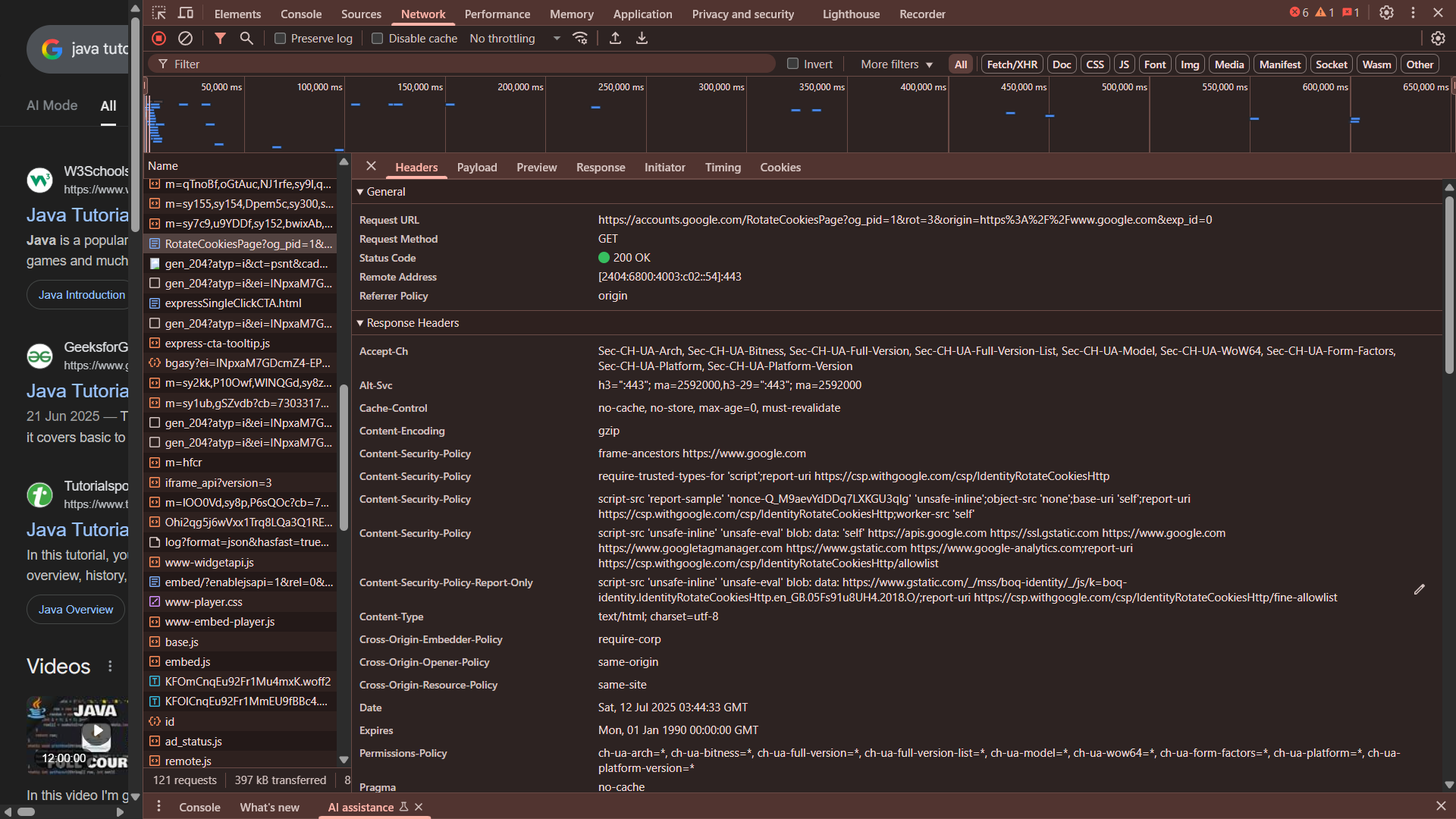
Content-Type: text/plain

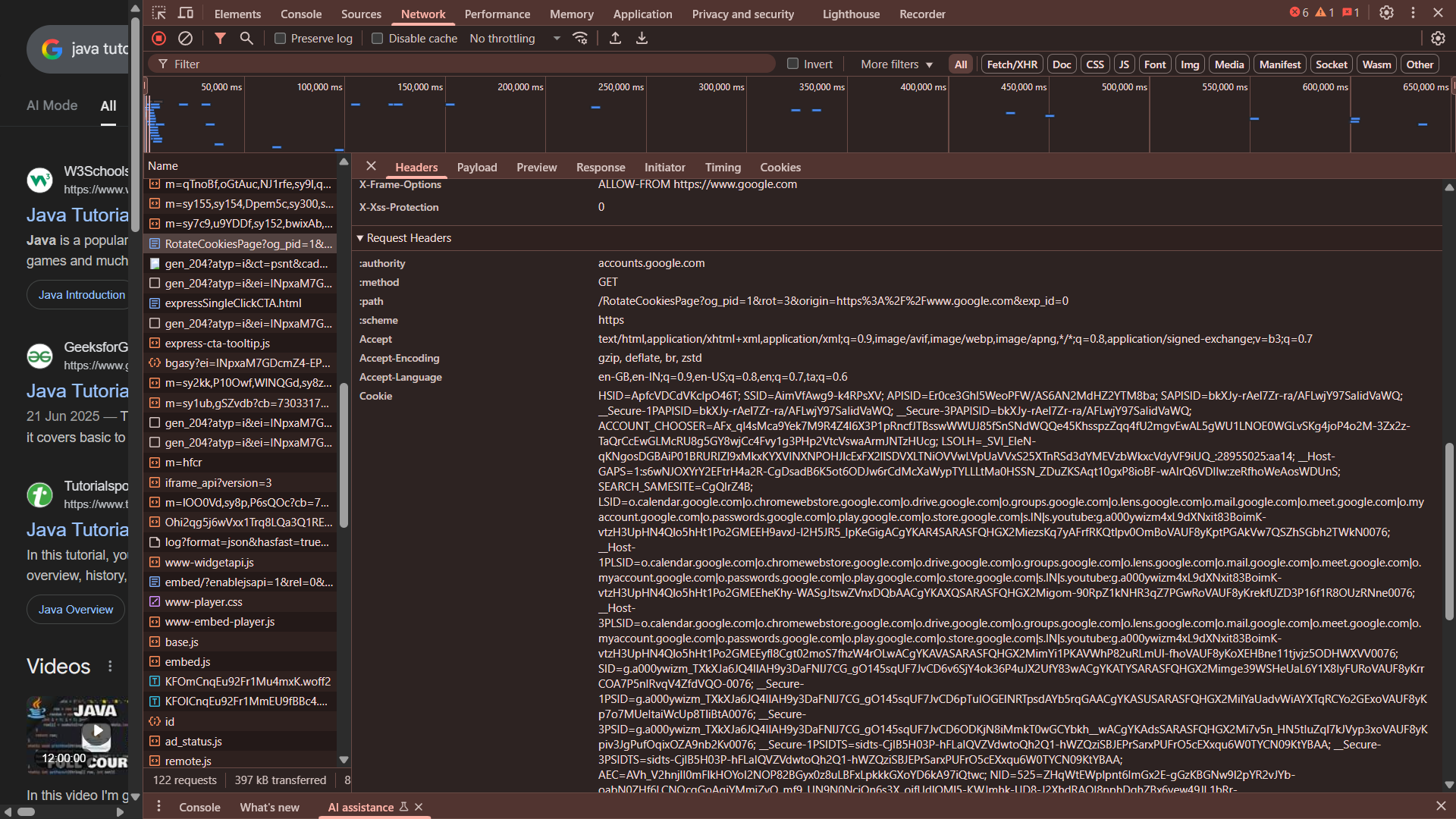
Hello World! My payload includes a trailing CRLF.

**Line-by-Line Explanation:**

* **Line 1**:
  + HTTP/1.1 → HTTP version
  + 200 → Status code (success)
  + OK → Status message
* **Date**: When the response was sent
* **Content-Type**: Format of response (determines how browser displays it)  
  Examples:
  + text/plain → plain text
  + text/html → HTML page
  + application/json → JSON
  + image/png → image bytes
* **Final Line**: Actual content/body of the response.

Request and response details in browser:





**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.

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 java.util.List;

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(scanBasePackages = "com.cognizant.spring\_learn")

public class SpringLearnApplication {

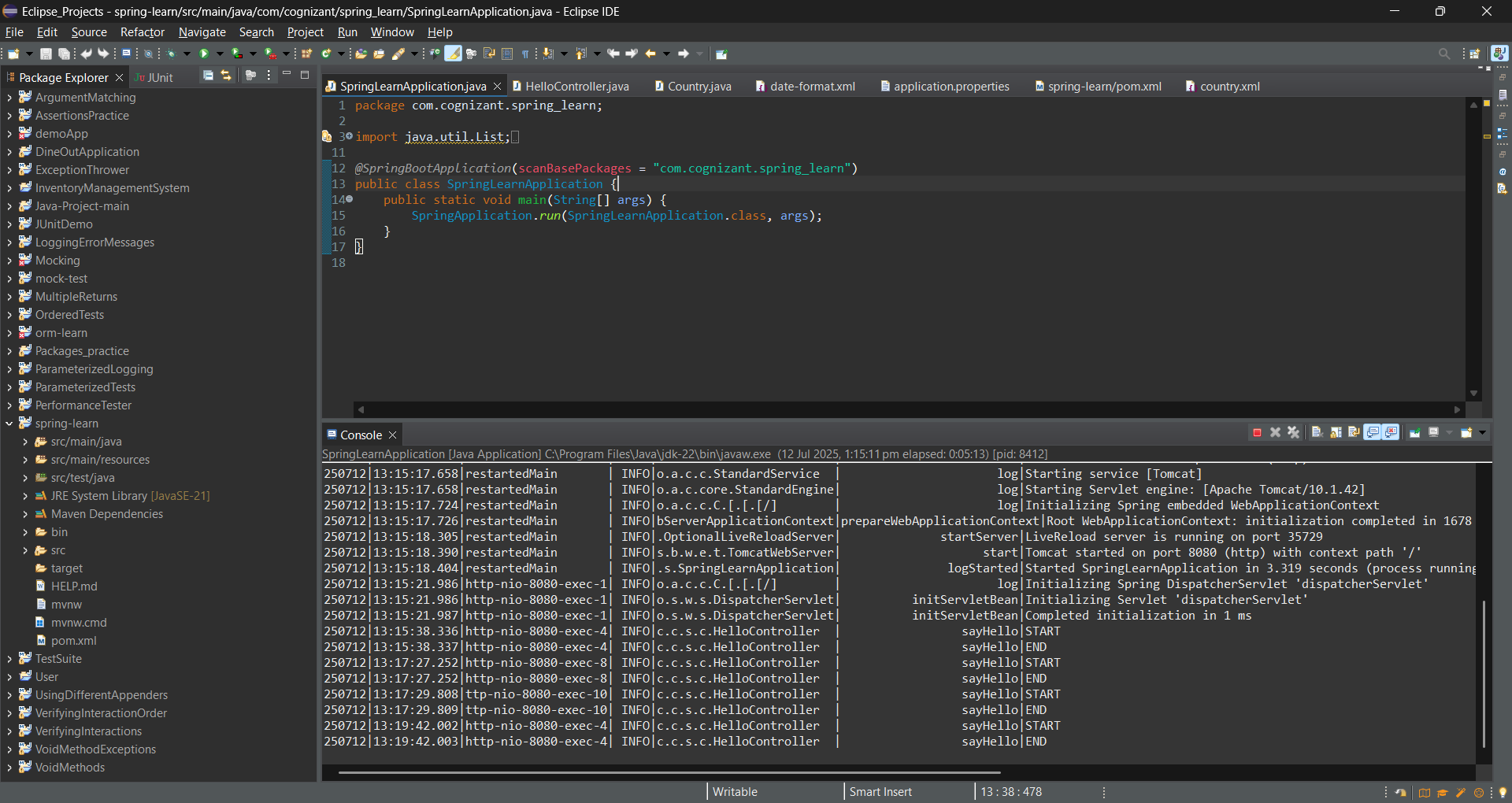
public static void main(String[] args) {

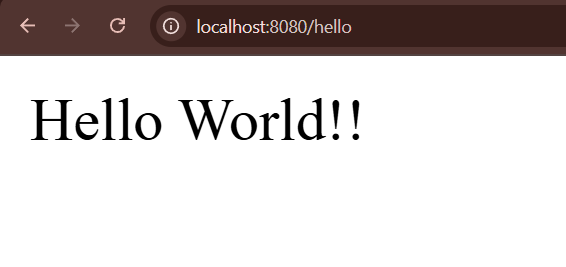
SpringApplication.run(SpringLearnApplication.class, args);

}

}

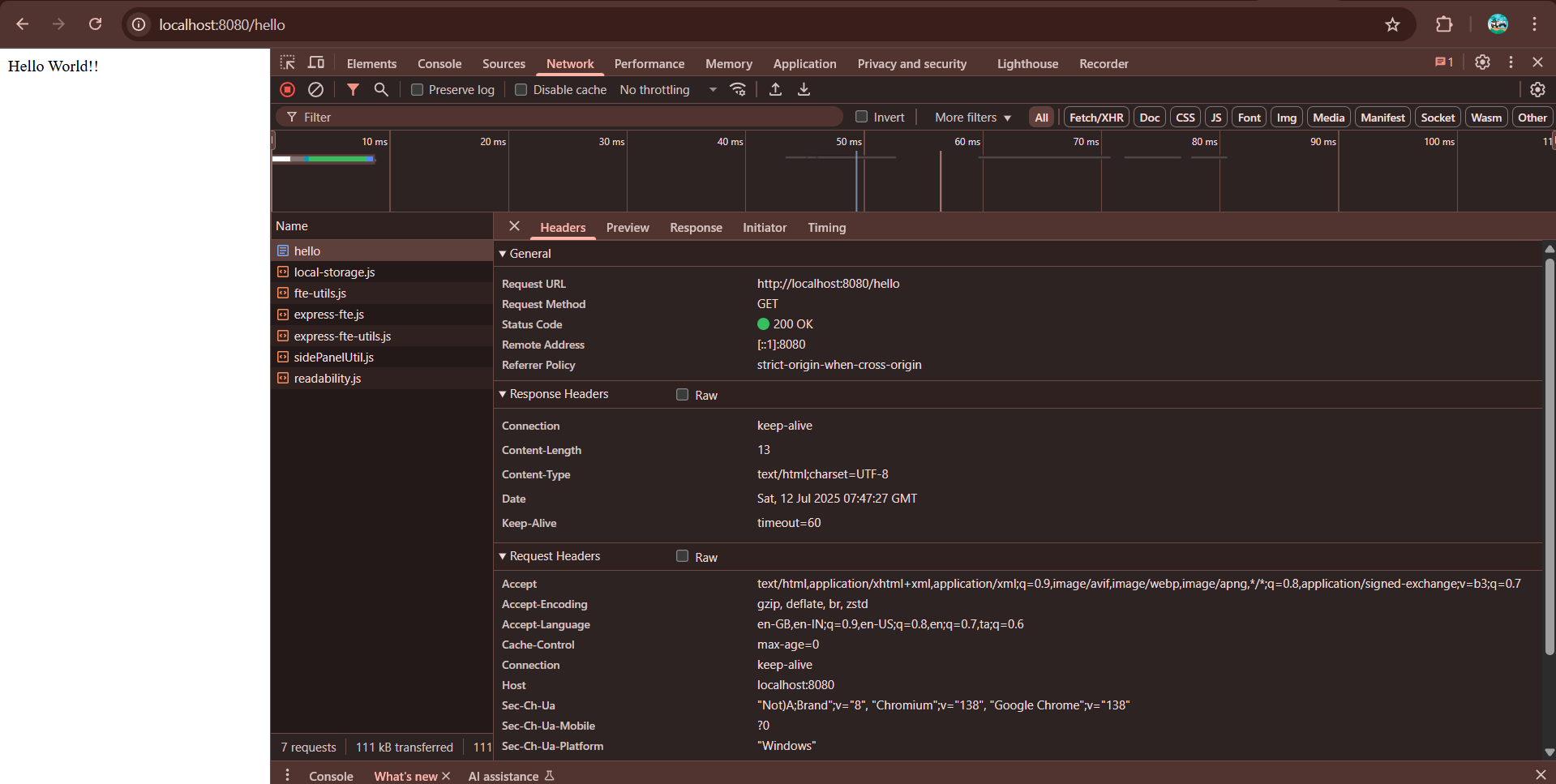
Output:



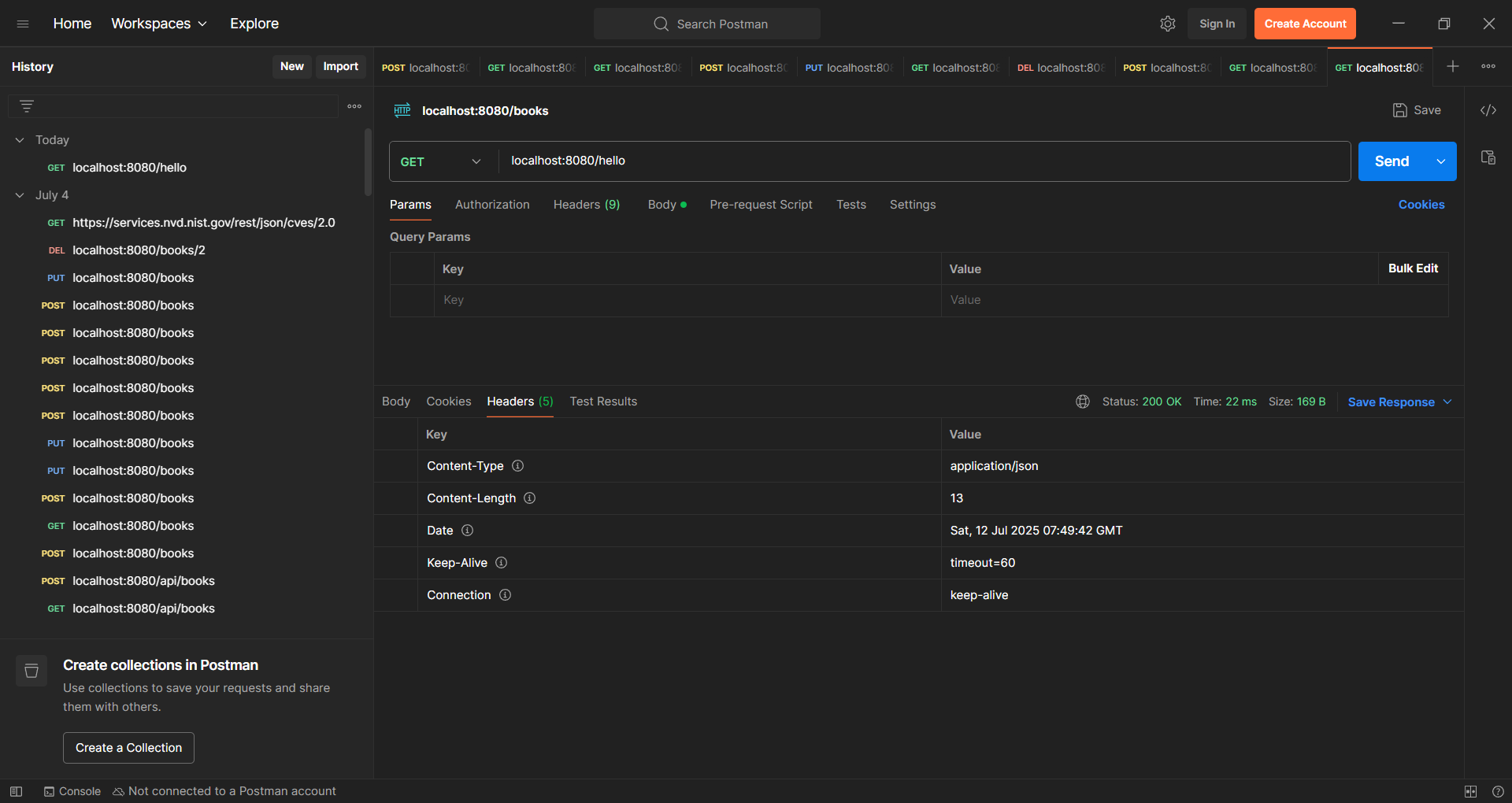


Try the URL http://localhost:8083/hello in both chrome browser and postman.  
  
SME to explain the following aspects:

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



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

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

*LOGGER*.info("END");

return country;

}

}

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

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

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

</bean>

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

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

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

</bean>

<!-- List of countries -->

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

<constructor-arg>

<list>

<ref bean="in"/>

<ref bean="us"/>

<ref bean="de"/>

<ref bean="jp"/>

</list>

</constructor-arg>

</bean>

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

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

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

</bean>

</beans>

**SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import java.util.List;

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(scanBasePackages = "com.cognizant.spring\_learn")

public class SpringLearnApplication {

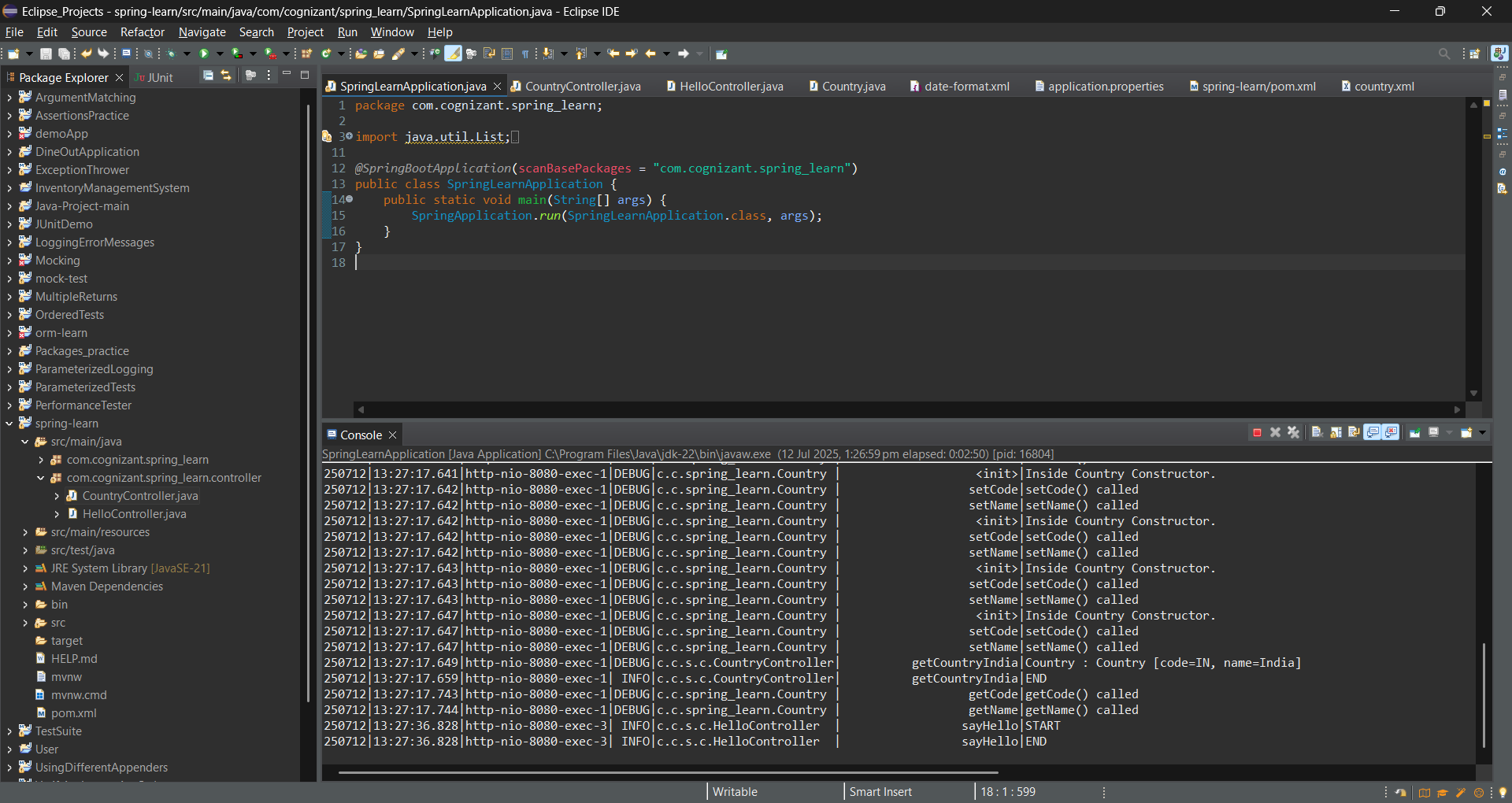
public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

}

**Output:**





**SME to explain the following aspects:**

**What happens in the controller method?**

* The @RequestMapping("/country") maps the URL to the method.
* A Spring ApplicationContext loads the country.xml configuration.
* It fetches the country bean (which has code = IN and name = India).
* The method returns the Country object directly.

### **How is the Java Bean converted to JSON?**

Spring Boot uses **Jackson** by default to convert Java objects (POJOs) to JSON.  
Since @RestController includes @ResponseBody, the object is automatically serialized to JSON like this:

Json:

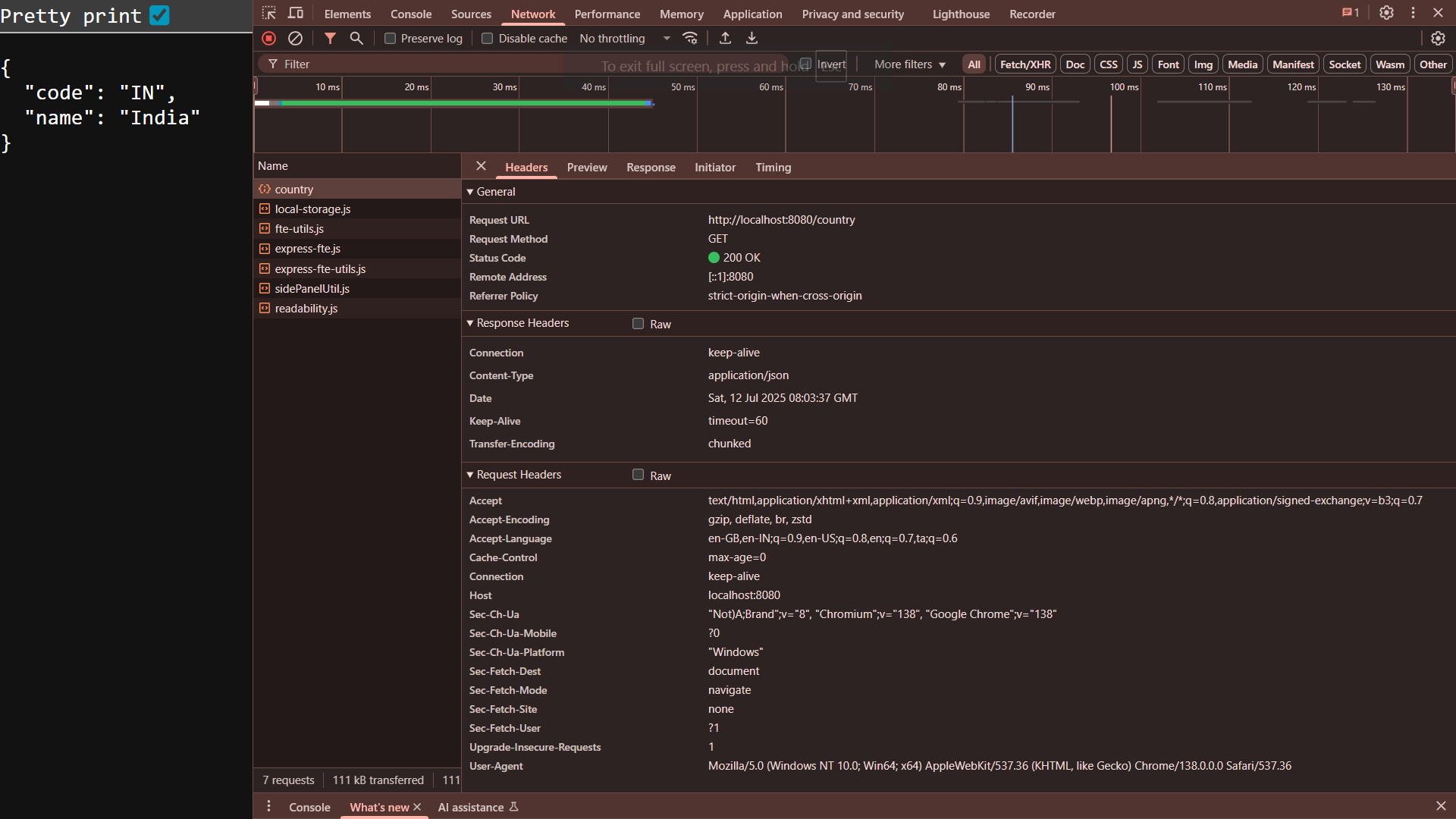
{

"code": "IN",

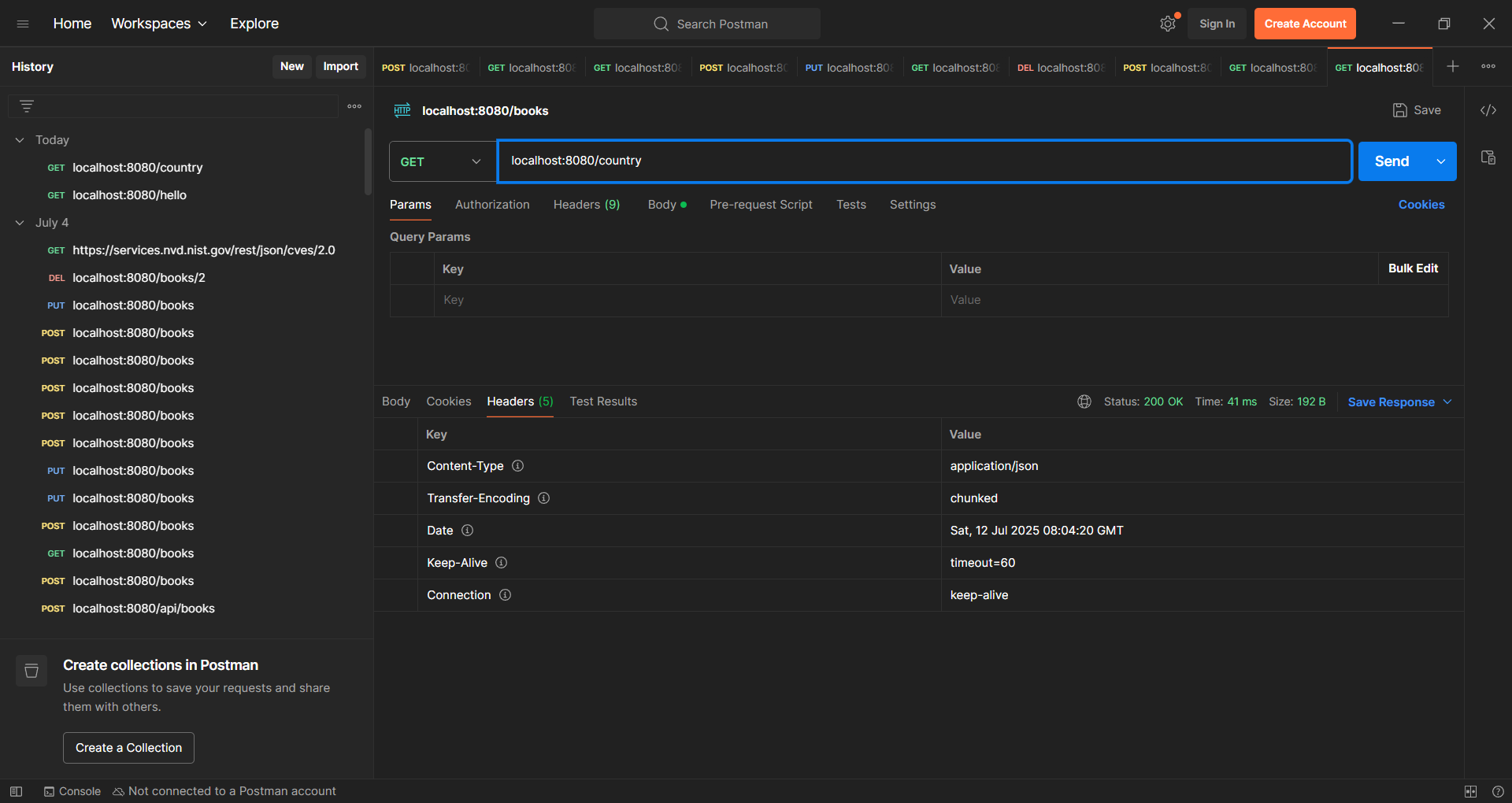
"name": "India"

}

**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**



**REST - Get all countries**   
  
**CoutryController.java**

package com.cognizant.spring\_learn.controller;

import java.util.List;

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.RequestMapping;

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

import com.cognizant.spring\_learn.Country;

*@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("country", Country.class);

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

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

return country;

}

*@GetMapping*("/countries")

public List<Country> getAllCountries() {

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

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

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

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

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

return countryList;

}

}

**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>

<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>

<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 java.util.List;

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(scanBasePackages = "com.cognizant.spring\_learn")

public class SpringLearnApplication {

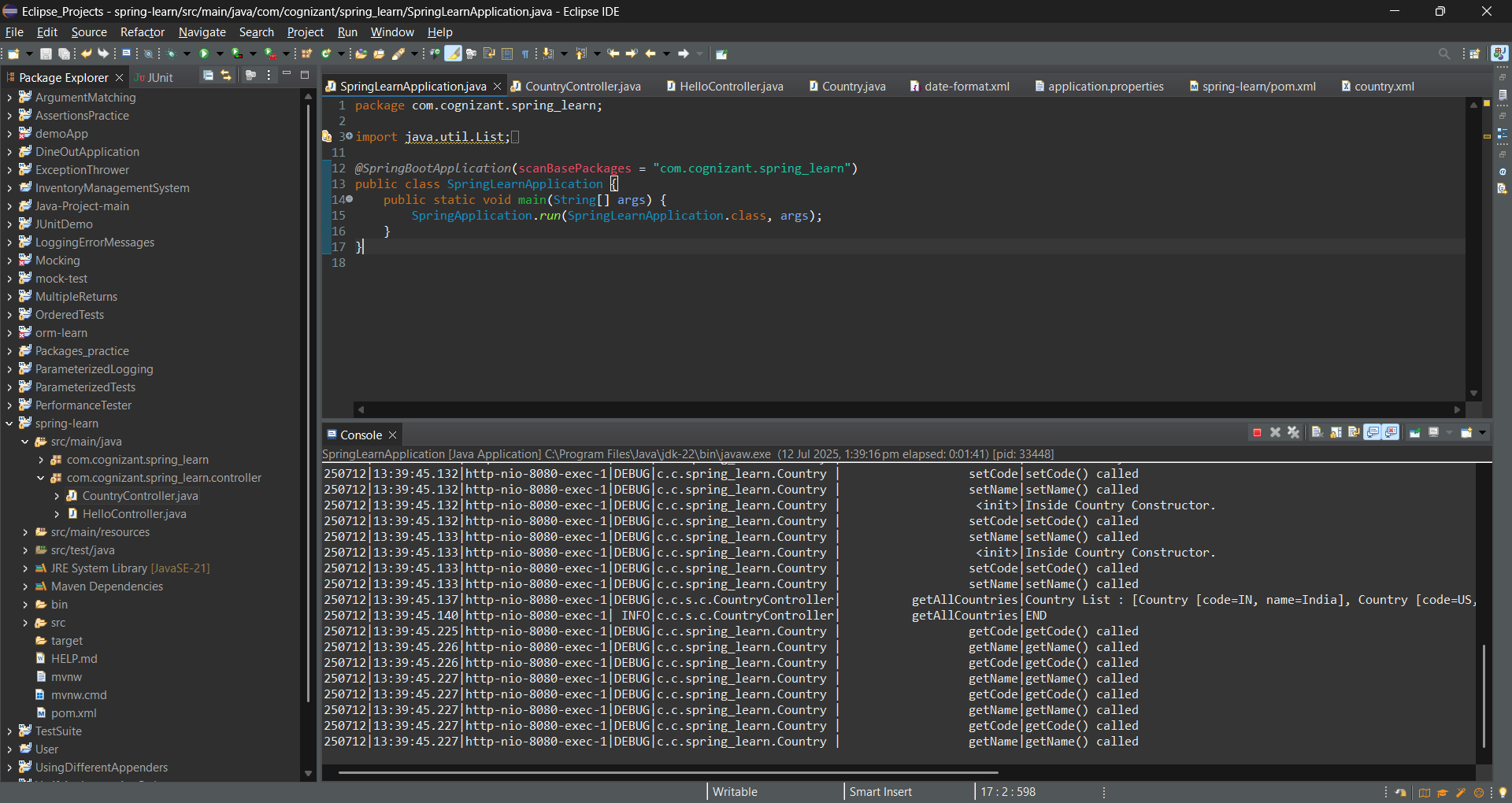
public static void main(String[] args) {

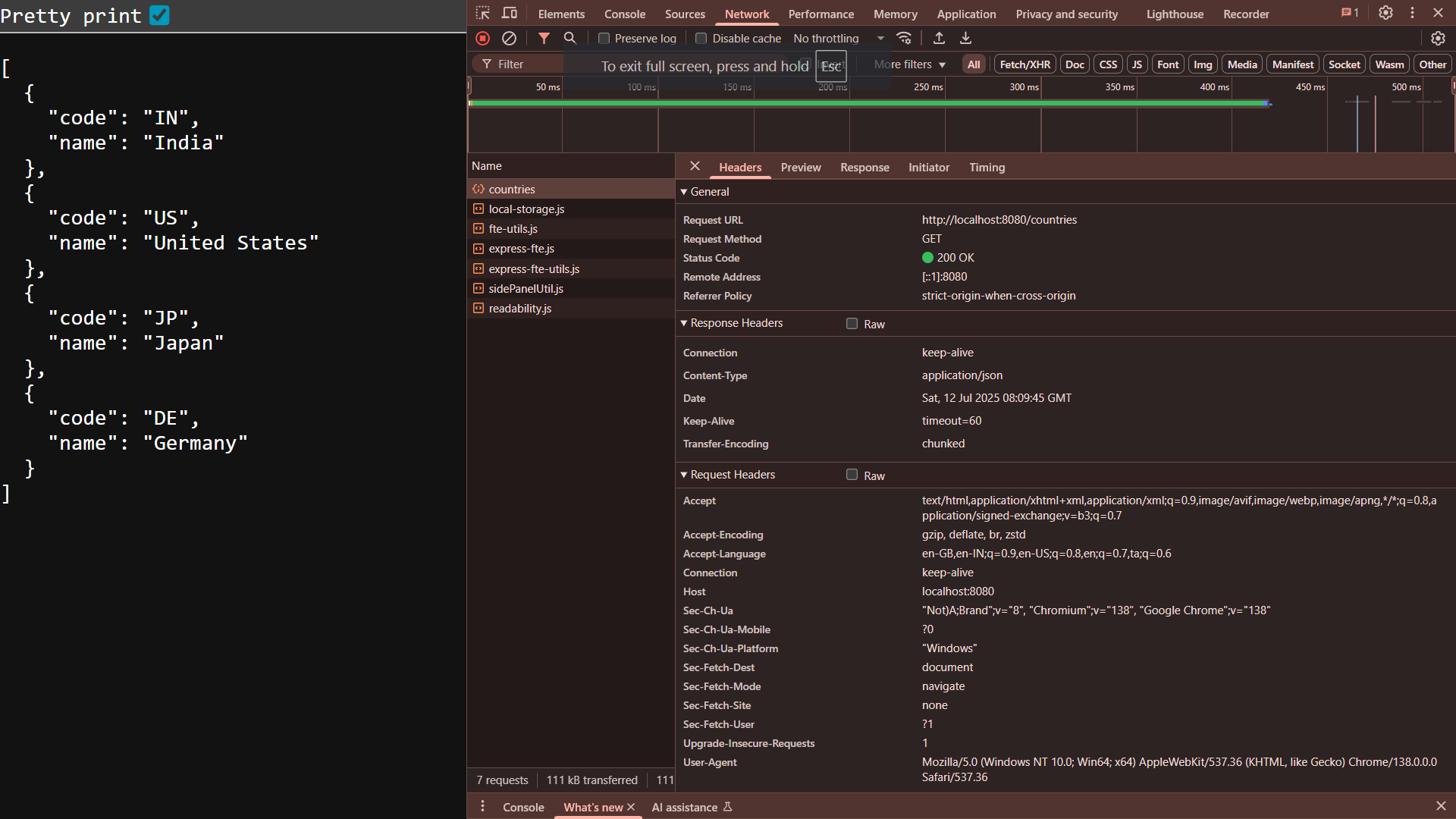
SpringApplication.run(SpringLearnApplication.class, args);

}

}

**Output:**





**REST - Get country based on country code**   
  
**CountryService.java**

package com.cognizant.spring\_learn.service;

import com.cognizant.spring\_learn.Country;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

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

.findFirst()

.orElse(null); // or throw an exception for 404

}

}

**Country.java**

<?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">

<!-- Individual country beans -->

<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>

<!-- Default country bean for /country API -->

<bean id="country" 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 java.util.List;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

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

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

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

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

import com.cognizant.spring\_learn.Country;

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

*@RestController*

public class CountryController {

*@Autowired*

private CountryService countryService;

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("country", Country.class);

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

*LOGGER*.info("END");

return country;

}

*@GetMapping*("/countries")

public List<Country> getAllCountries() {

*LOGGER*.info("START");

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

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

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

*LOGGER*.info("END");

return countryList;

}

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

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

*LOGGER*.info("START");

Country country = countryService.getCountry(code);

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

*LOGGER*.info("END");

return country;

}

}

**SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import java.util.List;

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(scanBasePackages = "com.cognizant.spring\_learn")

public class SpringLearnApplication {

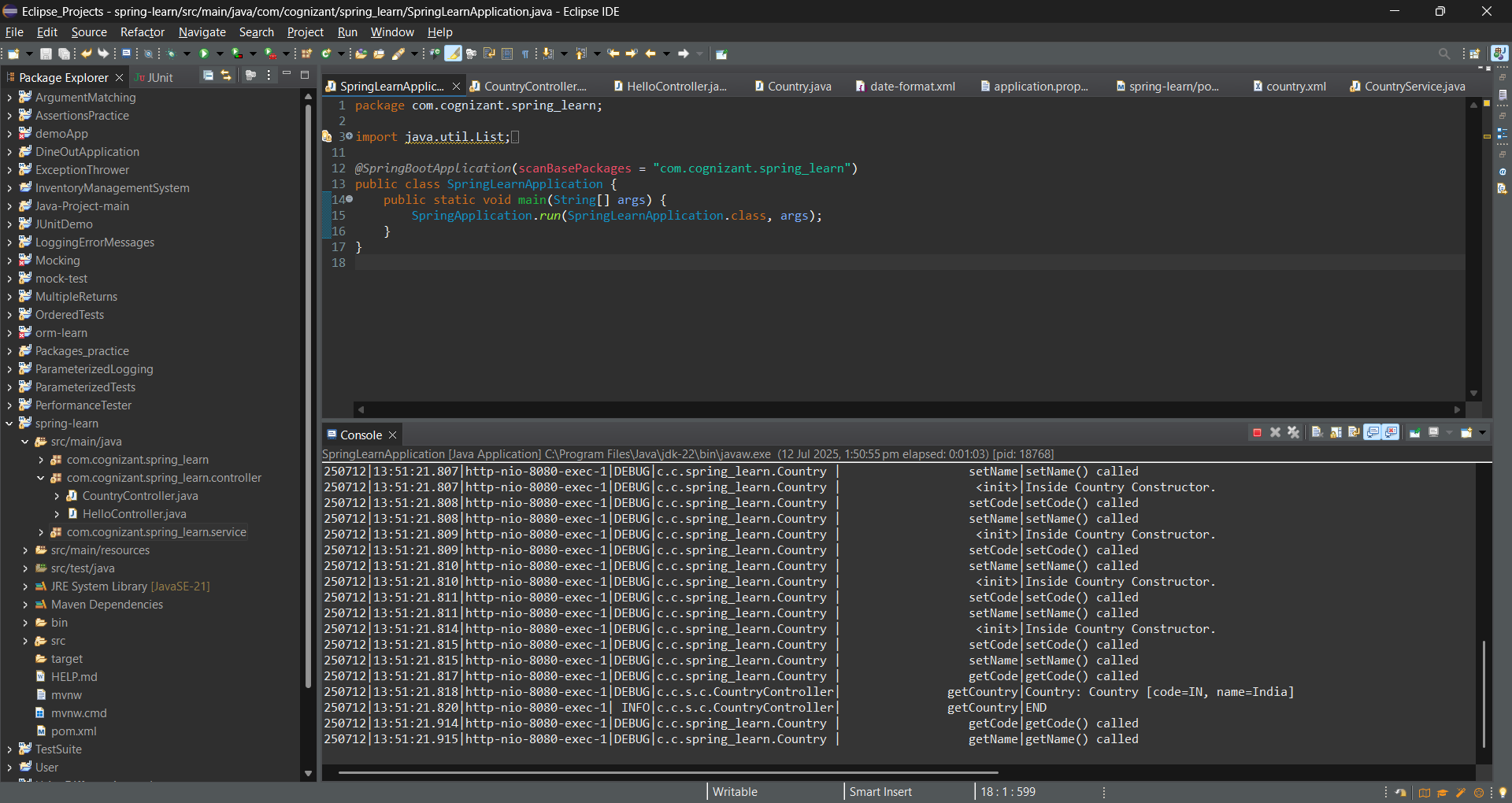
public static void main(String[] args) {

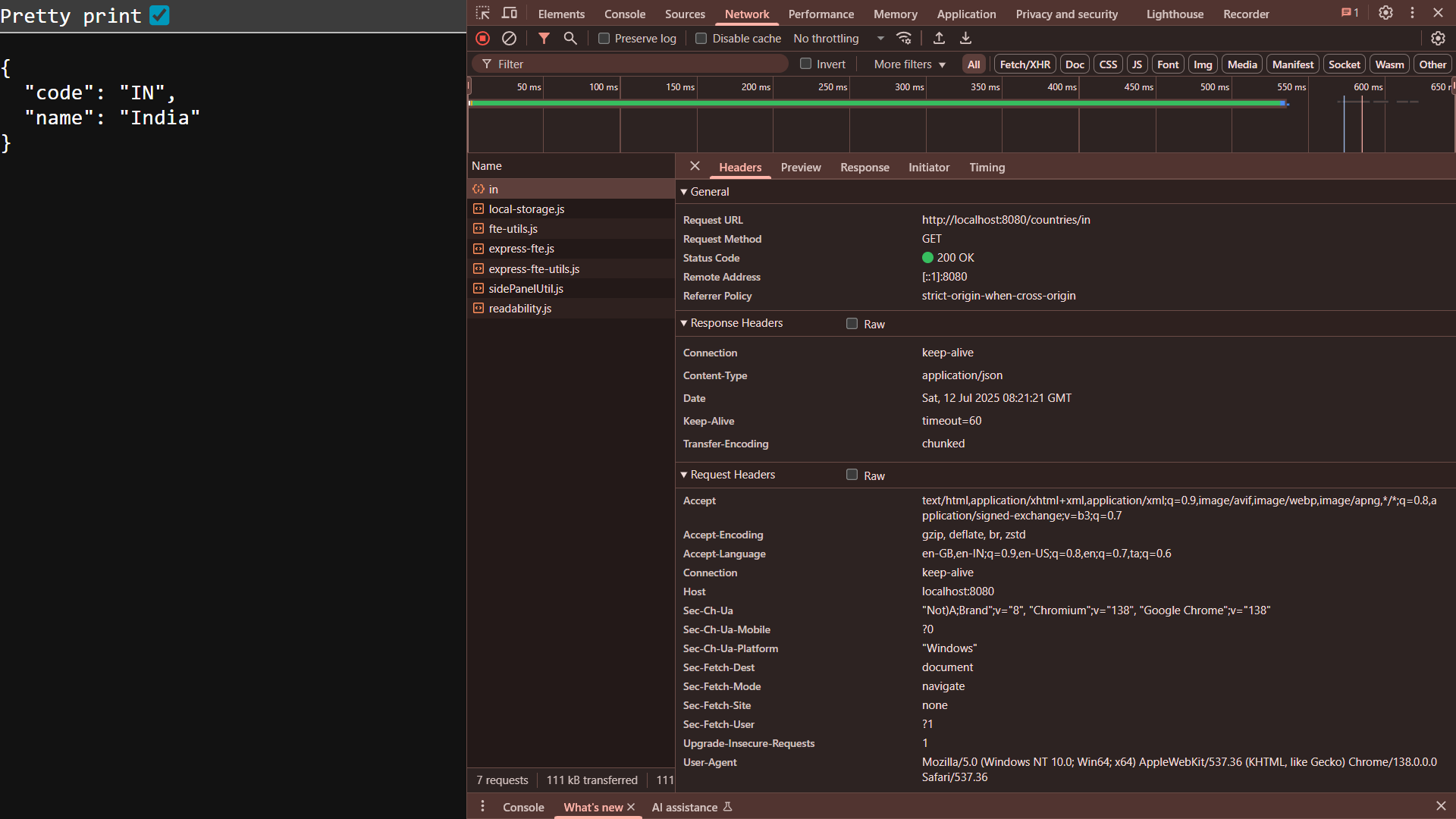
SpringApplication.run(SpringLearnApplication.class, args);

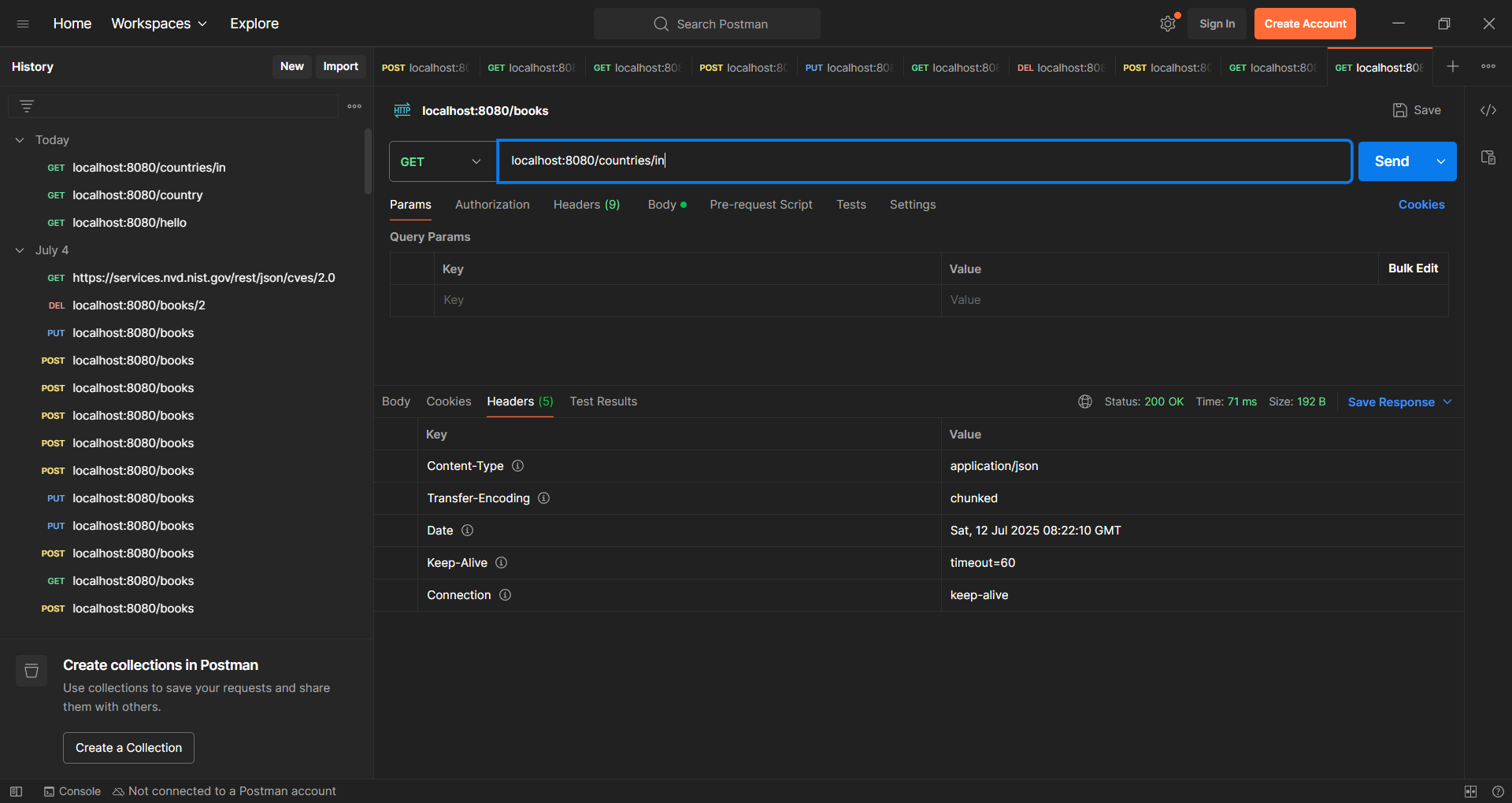
}

}

Output:







**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.  
  
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");

}

public CountryNotFoundException(String message) {

super(message);

}

}

**CountryService.java**

package com.cognizant.spring\_learn.service;

import com.cognizant.spring\_learn.Country;

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

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

*@Service*

public class CountryService {

public Country getCountry(String code) throws CountryNotFoundException {

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

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

return countries.stream()

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

.findFirst()

.orElseThrow(CountryNotFoundException::new);

}

}

**CountryController.java**

package com.cognizant.spring\_learn.controller;

import java.util.List;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

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

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

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

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

import com.cognizant.spring\_learn.Country;

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

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

*@RestController*

public class CountryController {

*@Autowired*

private CountryService countryService;

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("country", Country.class);

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

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

return country;

}

*@GetMapping*("/countries")

public List<Country> getAllCountries() {

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

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

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

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

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

return countryList;

}

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

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

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

Country country = countryService.getCountry(code);

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

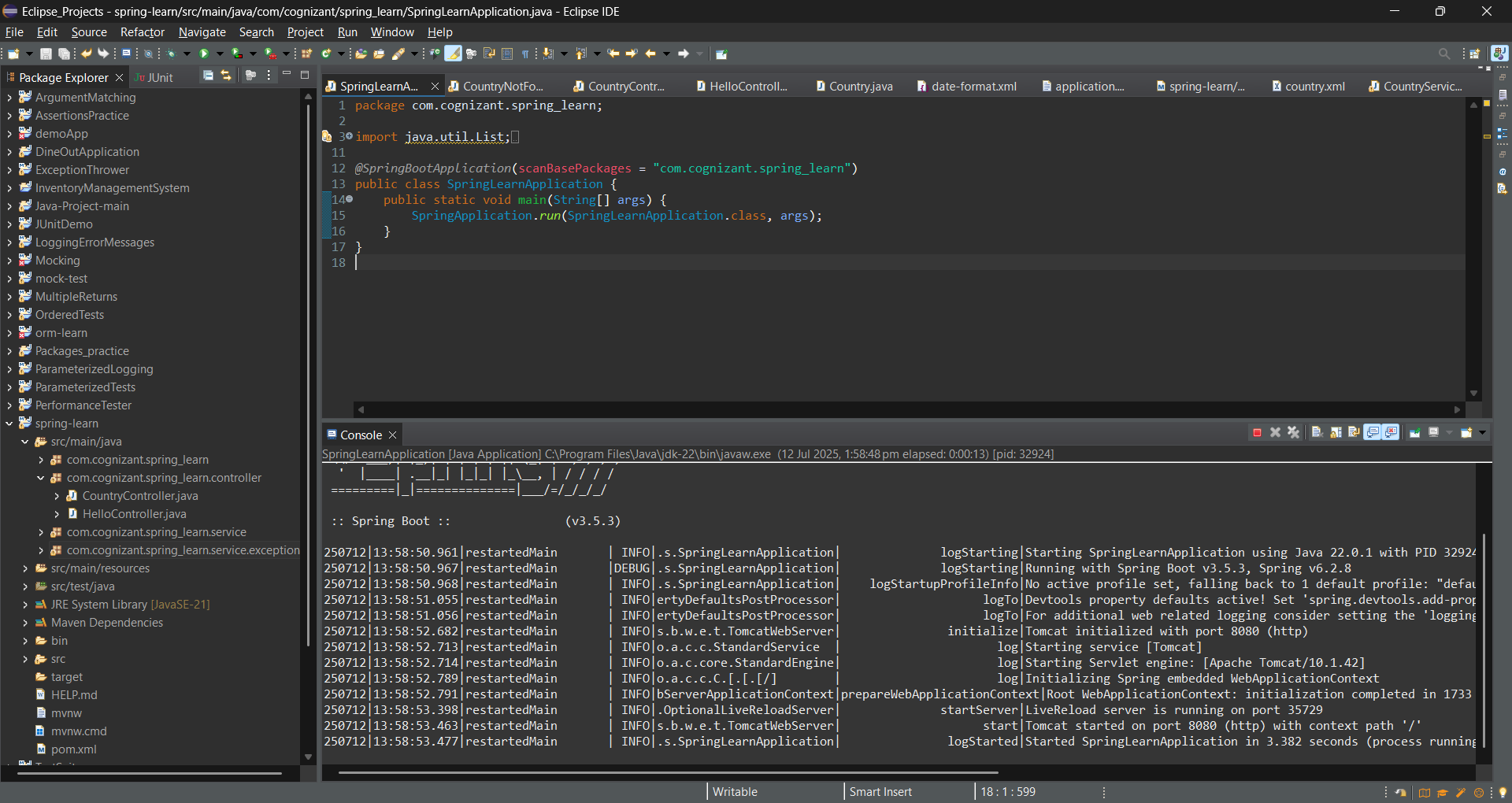
***LOGGER***.info("END");

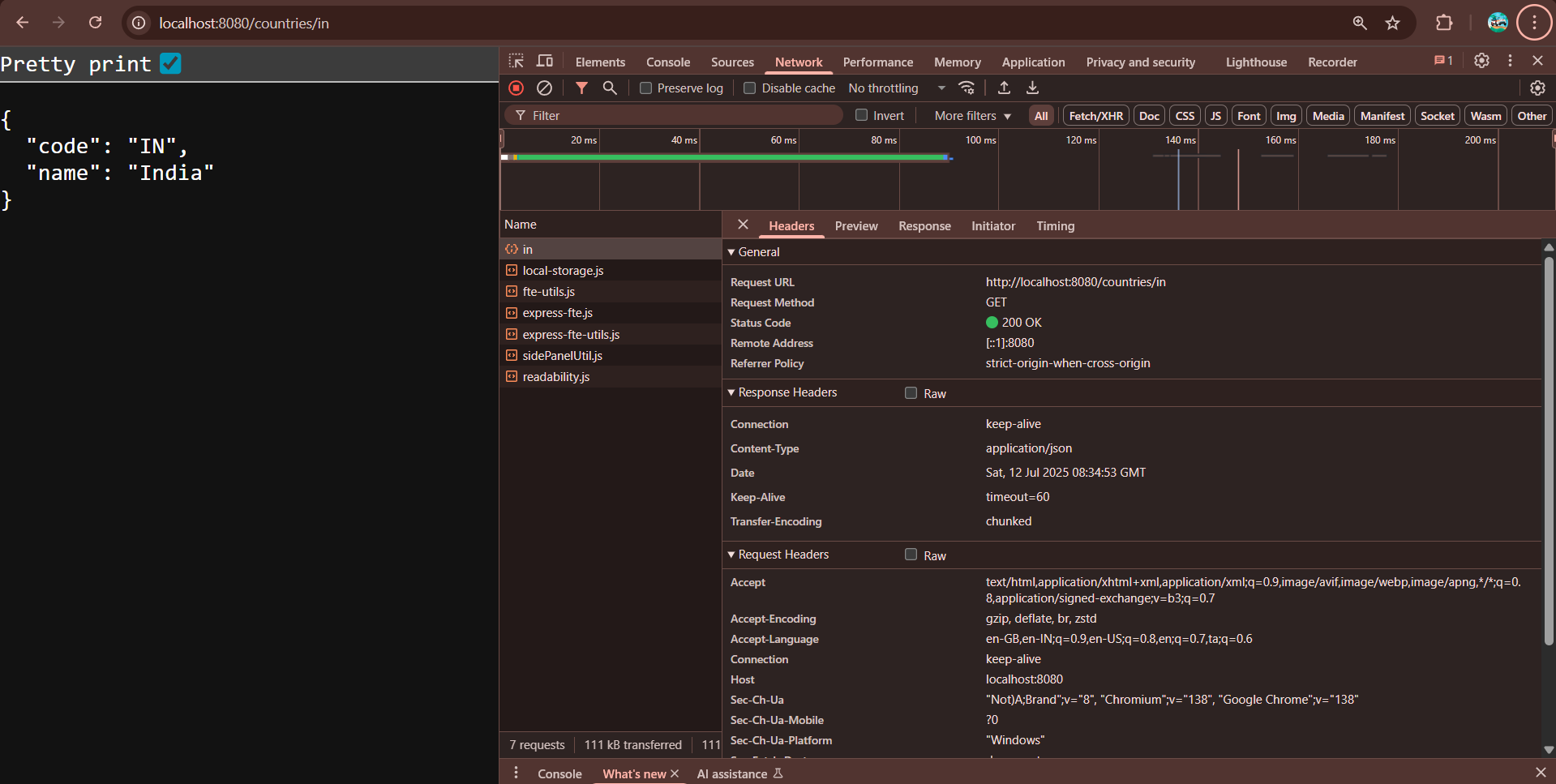
return country;

}

}

**Output:**





**MockMVC - Test get country service**   
  
**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*

public void contextLoads() {

*assertNotNull*(countryController);

}

*@Test*

public void testGetCountry() throws Exception {

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

// Check HTTP 200 OK

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

// Check JSON keys

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

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

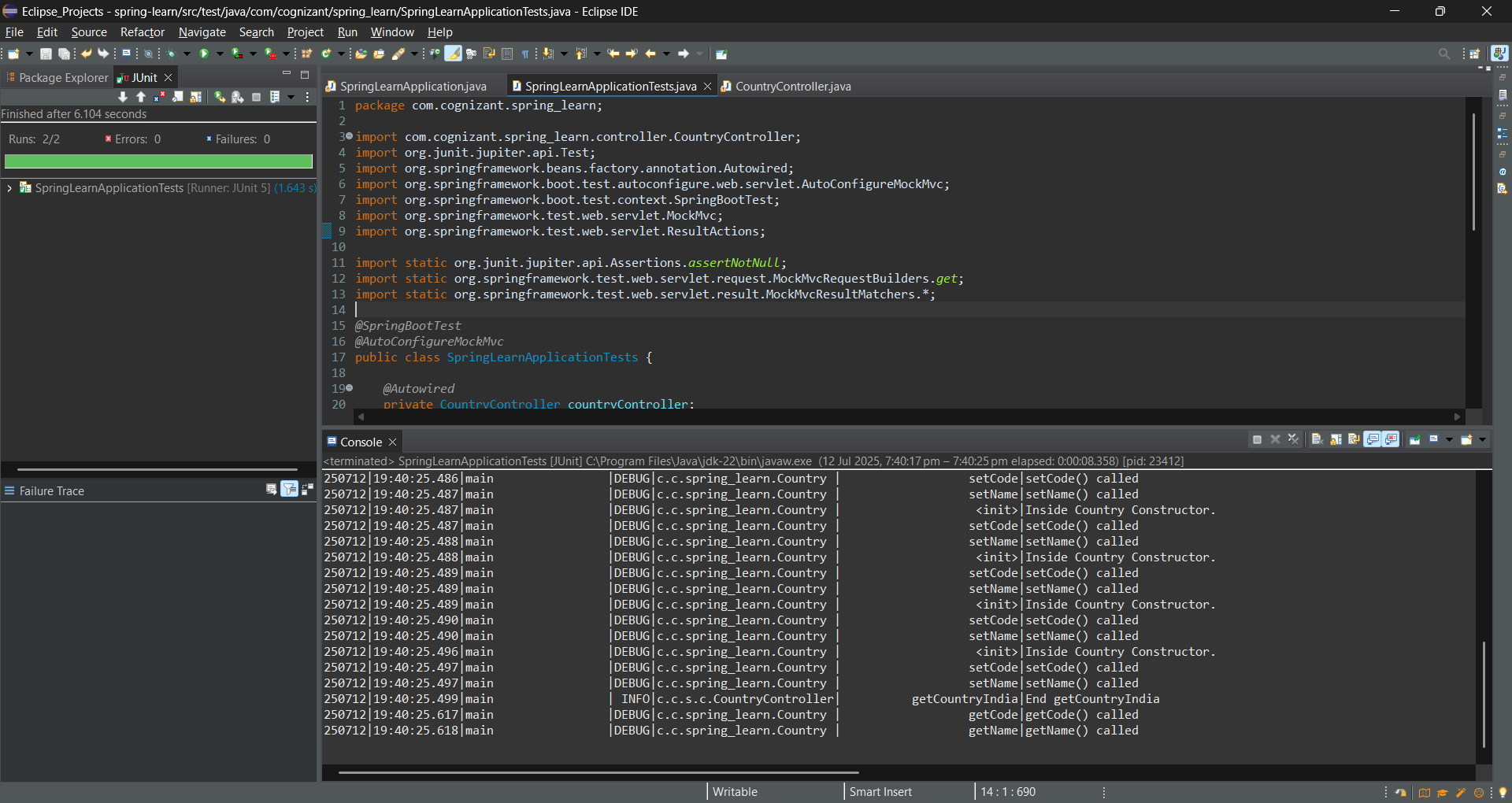
actions.andExpect(*jsonPath*("$.name").exists());

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

}

}

**Output:**



**MockMVC - Test get country service for exceptional scenario**   
  
**SpringLearnApplicationTest.java**

package com.cognizant.spring\_learn;

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

import org.junit.jupiter.api.Test;

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

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

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

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

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

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

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*

public void contextLoads() {

*assertNotNull*(countryController);

}

*@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"));

}

*@Test*

public void testGetCountryException() throws Exception {

ResultActions actions = mvc.perform(*get*("/countries/az"));

actions.andExpect(*status*().isNotFound())

.andExpect(*status*().reason("Country not found"));

}

}

**Output**:

