**Week 4: Spring REST using Spring Boot 3 Hands-on Summary**

**Student Name: Jitesh Sen**

**Email: jiteshsen2002@gmail.com**

**Superset ID: 6387362**

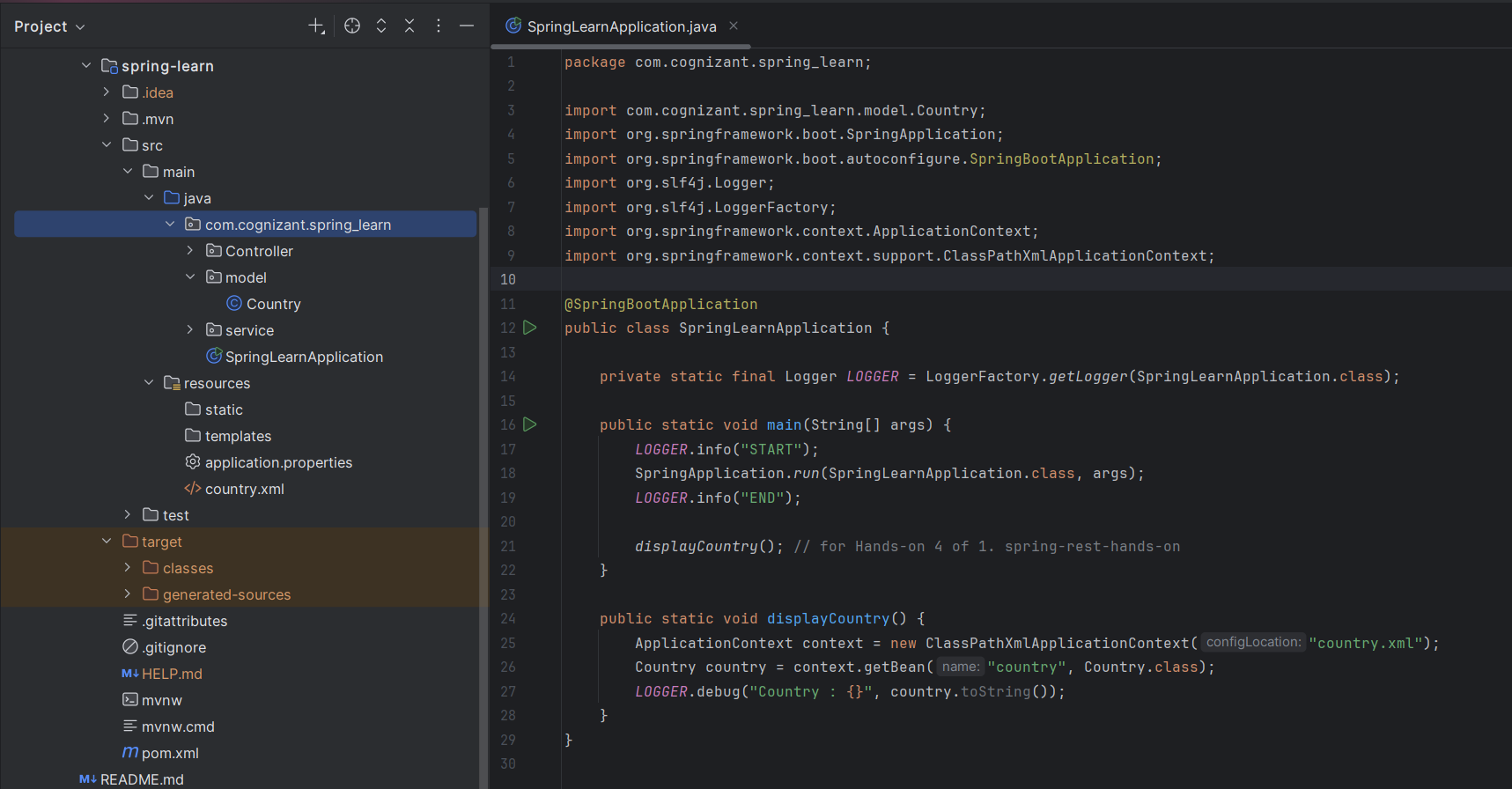
* **Hands-on 1.1: Spring Web Project Setup**

**Steps Covered:**

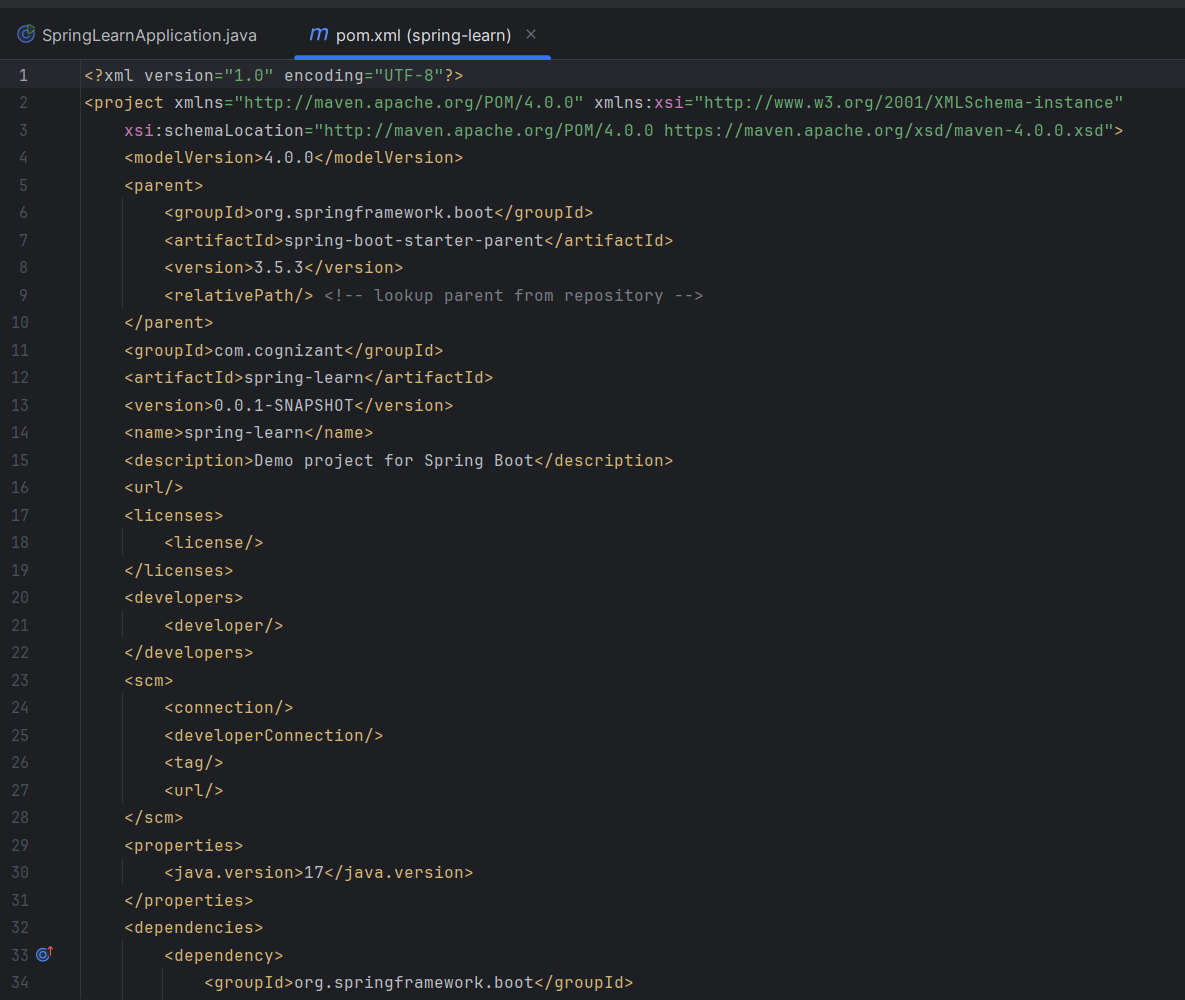
* Created Maven project from [start.spring.io](https://start.spring.io/) with group com.cognizant and artifact spring-learn
* Selected dependencies: Spring Web, Spring Boot DevTools
* Extracted ZIP into Intellij Workspace
* Built using proxy-aware Maven command
* Imported via “Existing Maven Projects” option
* Verified startup logs in SpringLearnApplication.java

**Screenshot Targets:**

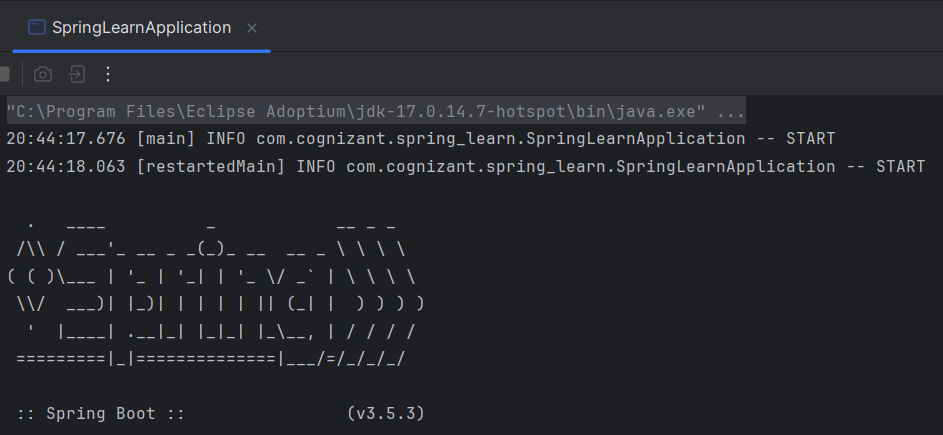
* Eclipse project structure: src/main/java, src/main/resources, src/test/java



* pom.xml Dependency Hierarchy tab



* Console logs showing START and END



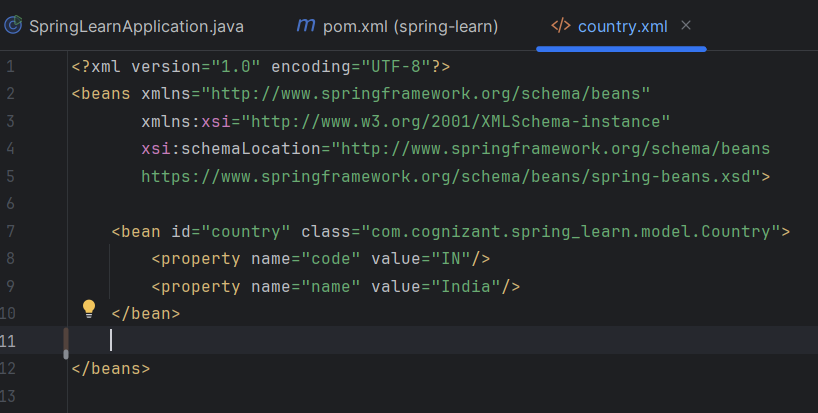
* **Hands-on 1.4: Load Country from Spring XML**

**Objectives Completed:**

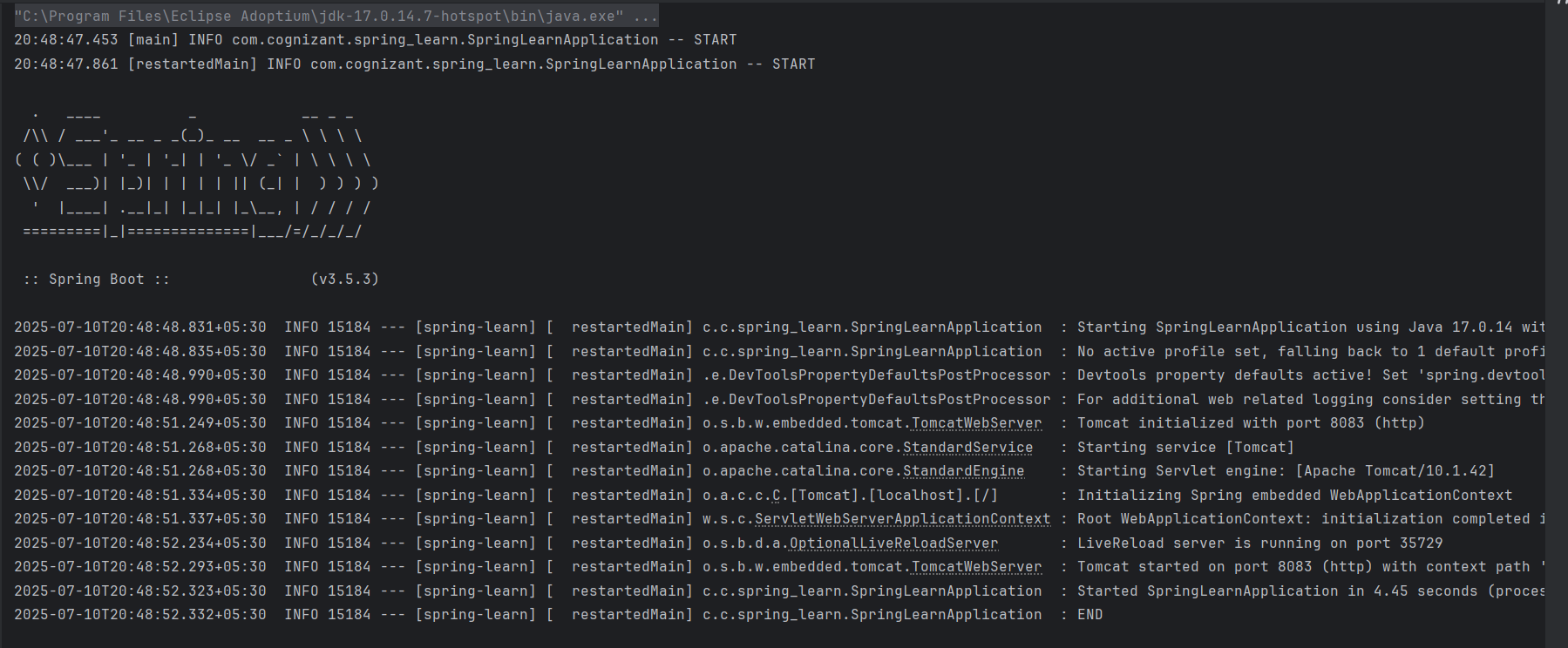
* Created Country bean in country.xml
* Implemented Country class with constructor, logging in setters/getters, and toString()
* Added displayCountry() method in SpringLearnApplication to load the bean and log its details

**Screenshot Targets:**

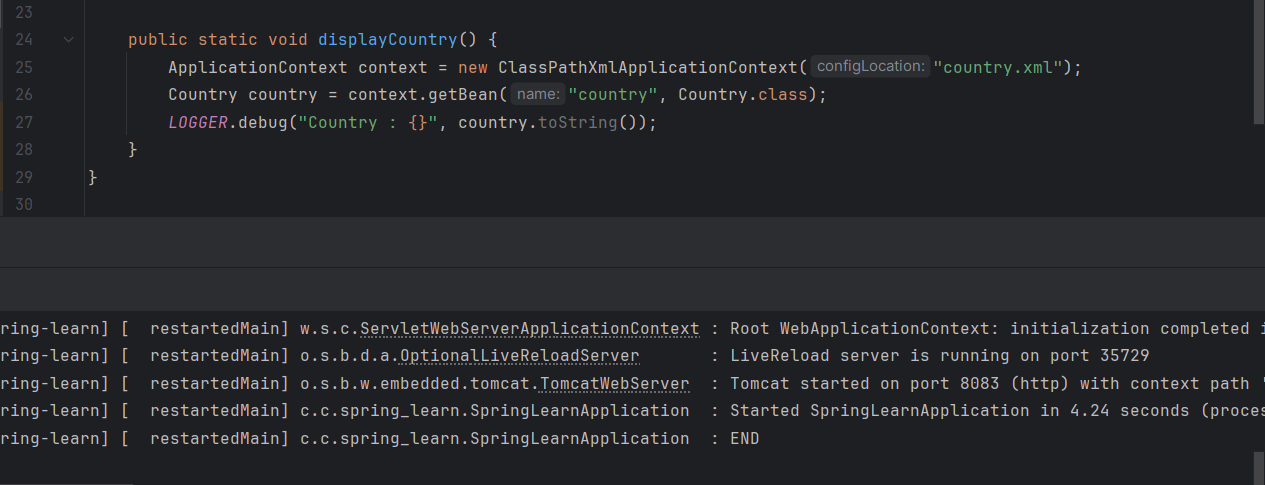
* XML configuration for country bean



* Console output showing bean lifecycle logging



* displayCountry() invocation and debug output



* **Hands-on 2.1: Hello World REST API**

**Endpoint:**   
GET http://localhost:8083/hello

**Controller Method:**

@GetMapping("/hello")

public String sayHello() {

log.info("Start");

String s = "Hello World!!";

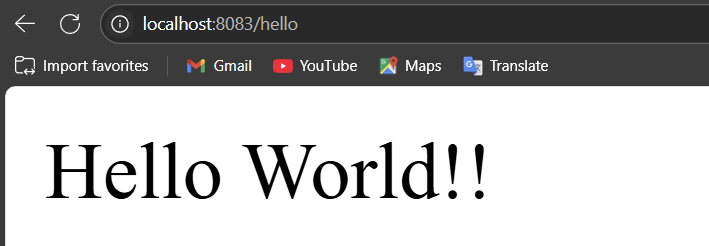
log.info("end");

return s;

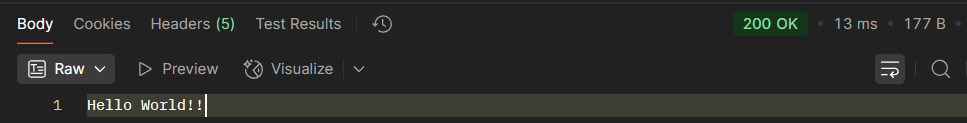
}

**Screenshot Targets:**

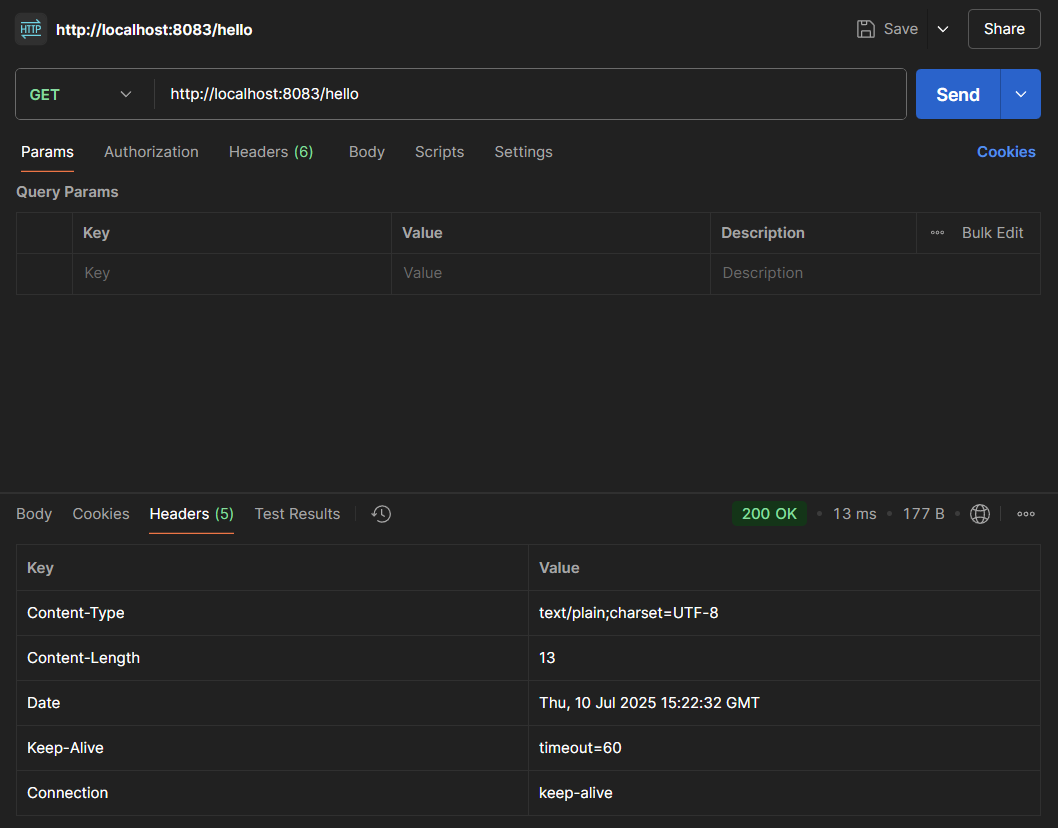
* Browser response: Hello World!!



* Postman response tab



* Postman "Headers" tab



* **Hands-on 2.2: Single Country Endpoint**

**Endpoint:**   
GET http://localhost:8083/country

**Details:**

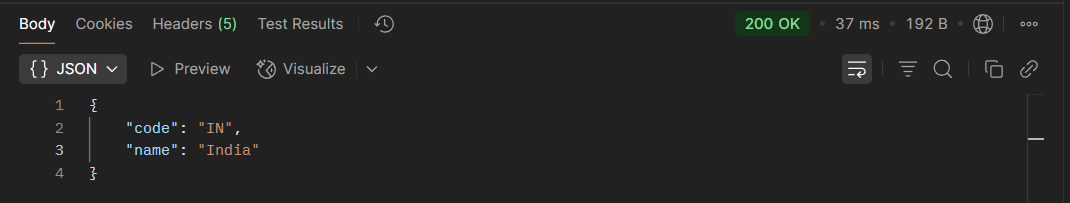
* Loads "country" bean from XML
* Returns India details in JSON

**Sample Response:**

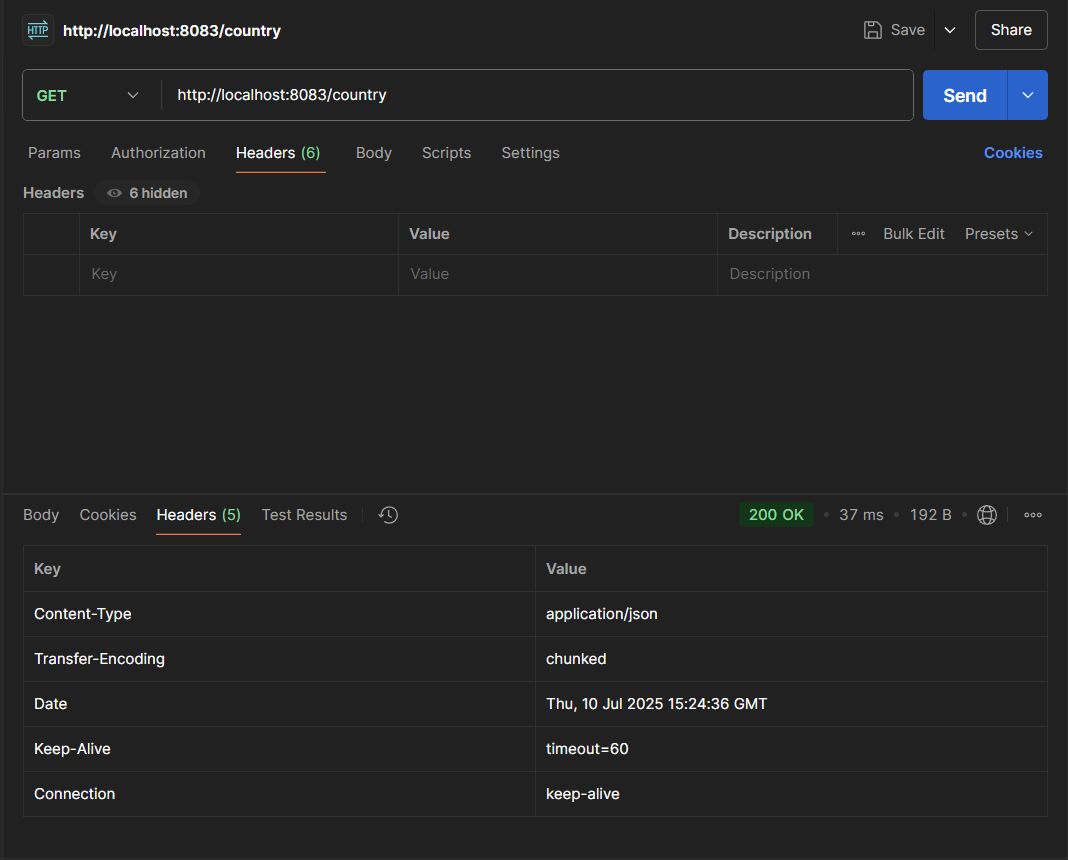
{ "code": "IN", "name": "India" }

**Screenshot Targets:**

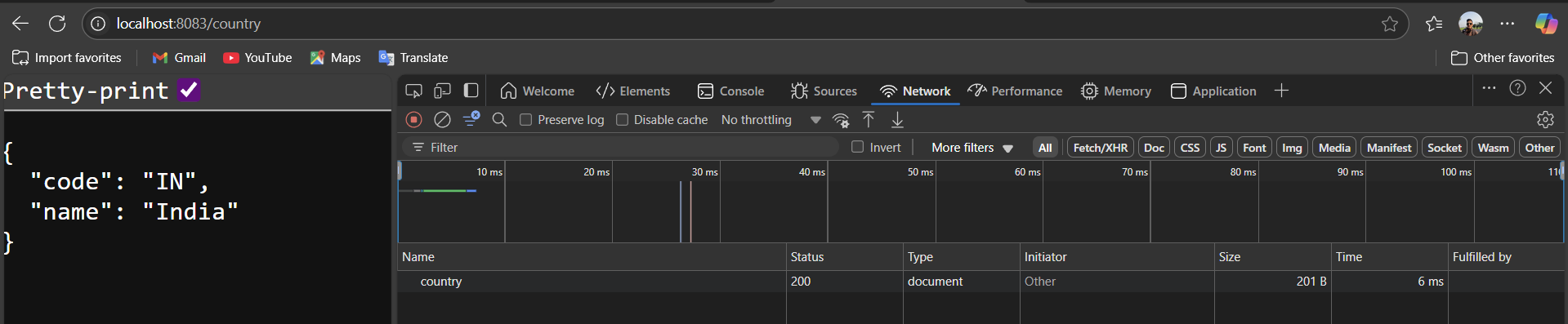
* Postman response with JSON



* Postman headers



* DevTools Network tab



* **Hands-on 2.3: All Countries List**

**Endpoint:**   
GET http://localhost:8083/countries

**Details:**

* Loads bean list named "countryList" from XML
* Returns list in JSON format

**Sample Response:**

[

{ "code": "IN", "name": "India" },

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

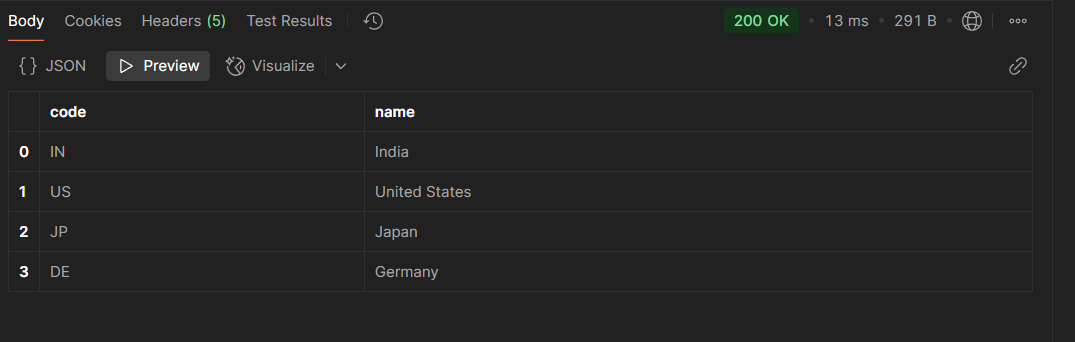
{ "code": "JP", "name": "Japan" },

{ "code": "DE", "name": "Germany" }

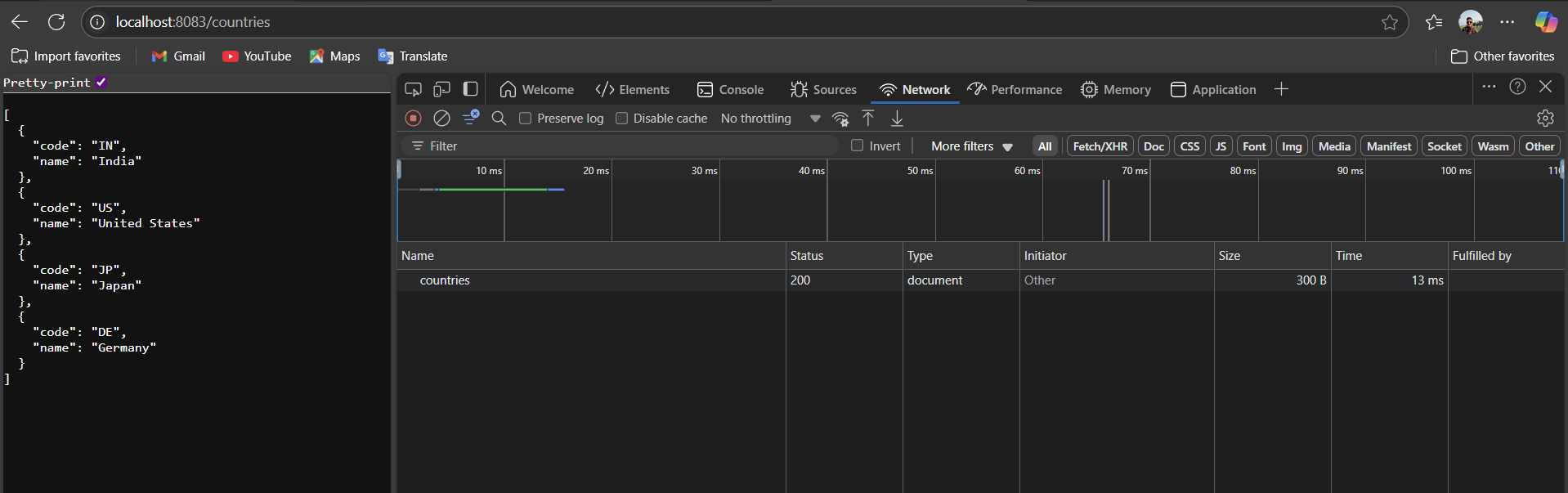
]

**Screenshot Targets:**

* Postman Preview



* DevTools response preview



* **Hands-on 2.4: Country Lookup by Code**

**Endpoint Examples:**

* GET http://localhost:8083/countries/in
* GET http://localhost:8083/countries/US

**Details:**

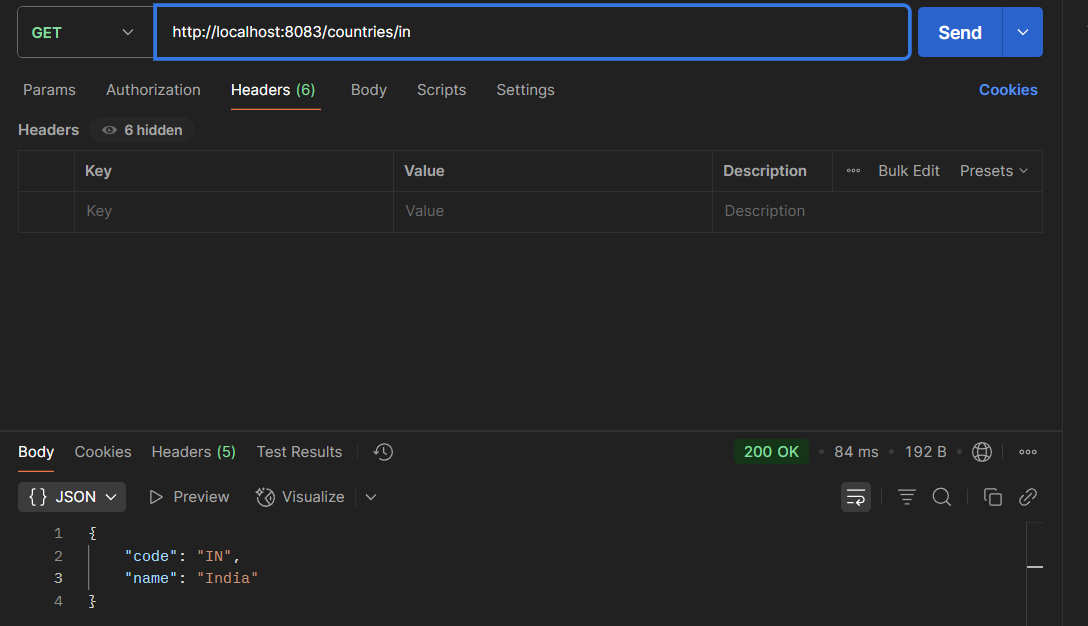
* @PathVariable to accept case-insensitive code
* Searches "countryList" using lambda expression

**Sample Response:**

{ "code": "IN", "name": "India" }

**Screenshot Targets:**

* Multiple requests with varying case (IN/in/Us/US)
* JSON output in Postman





* Header’s tab showing response status & content type

