**REST - Country Web Service**

**REST - Get all countries** 

* **Step 1:**

Open spring initiallizr

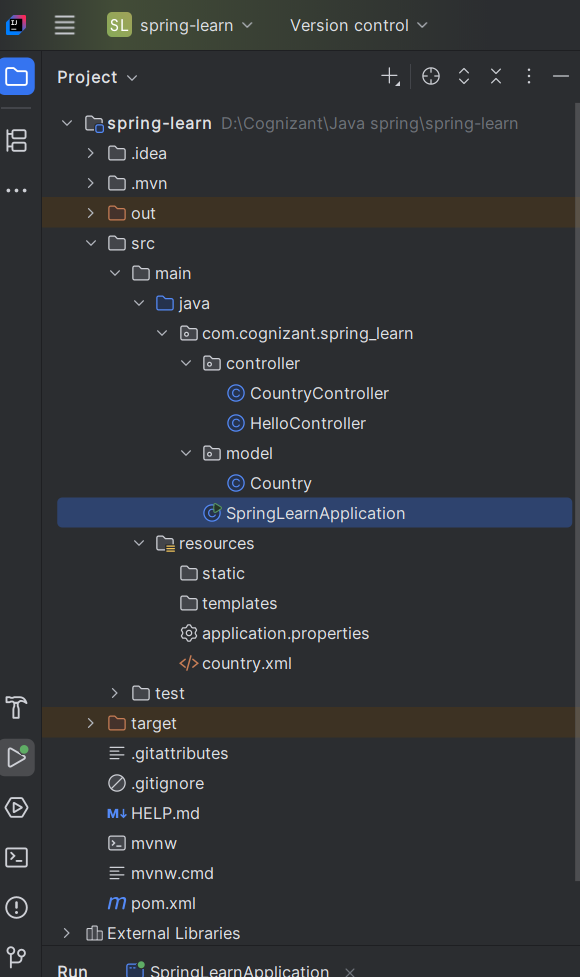
A screenshot of a computer

AI-generated content may be incorrect.

* Do the following things as shown in the picture and then generate it.
* And then extract the generated folder and open it in IntelliJ.

OR

Open intelliJ and in the existing project SPRING\_LEARN add the following :



* Step 2: Insert the codes given below in the classes mentioned:
* **com.cognizant.spring\_learn → New → Package → name it: model**
* **Create a new Java class: 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;

}

}

* **resources folder → New → File → 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

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

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

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

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

</bean>

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

<constructor-arg>

<list>

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

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

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

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

</bean>

</list>

</constructor-arg>

</bean>

</beans>

* **com.cognizant.spring\_learn.controller, create a file CountryController.java**

package com.cognizant.spring\_learn.controller;

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

import java.util.List;

@RestController

public class CountryController {

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

@RequestMapping("/country")

public Country getCountryIndia() {

LOGGER.info("START getCountryIndia");

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

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

LOGGER.info("END getCountryIndia");

return country;

}

@GetMapping("/countries")

public List<Country> getAllCountries() {

LOGGER.info("START getAllCountries");

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

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

LOGGER.info("END getAllCountries");

return countries;

}

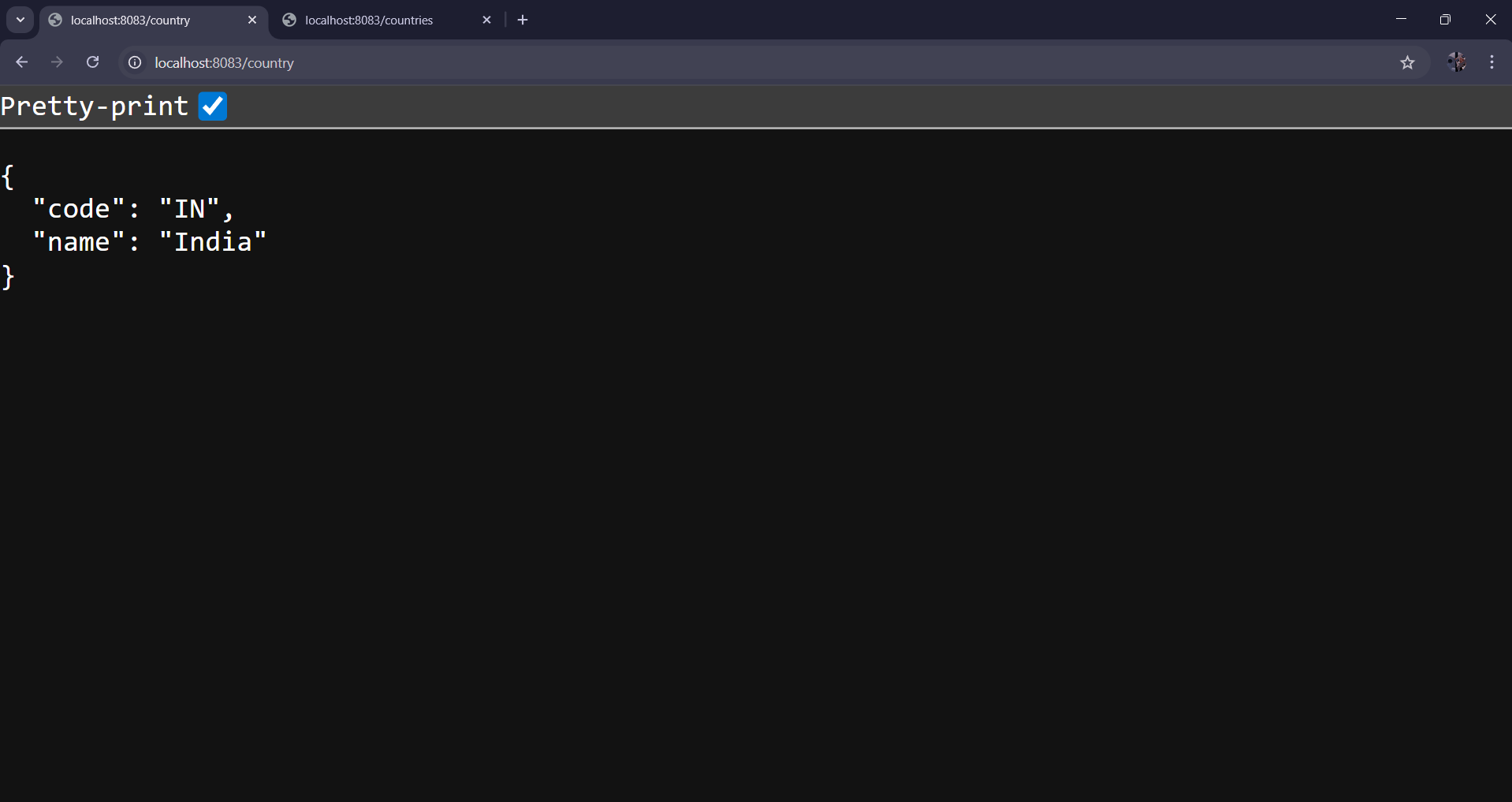
}

**Output of the browser:**

A screenshot of a computer

AI-generated content may be incorrect.

**REST - Country Web Service :**

****

**REST - Get all countries:**

**A computer screen with a black screen

AI-generated content may be incorrect.**

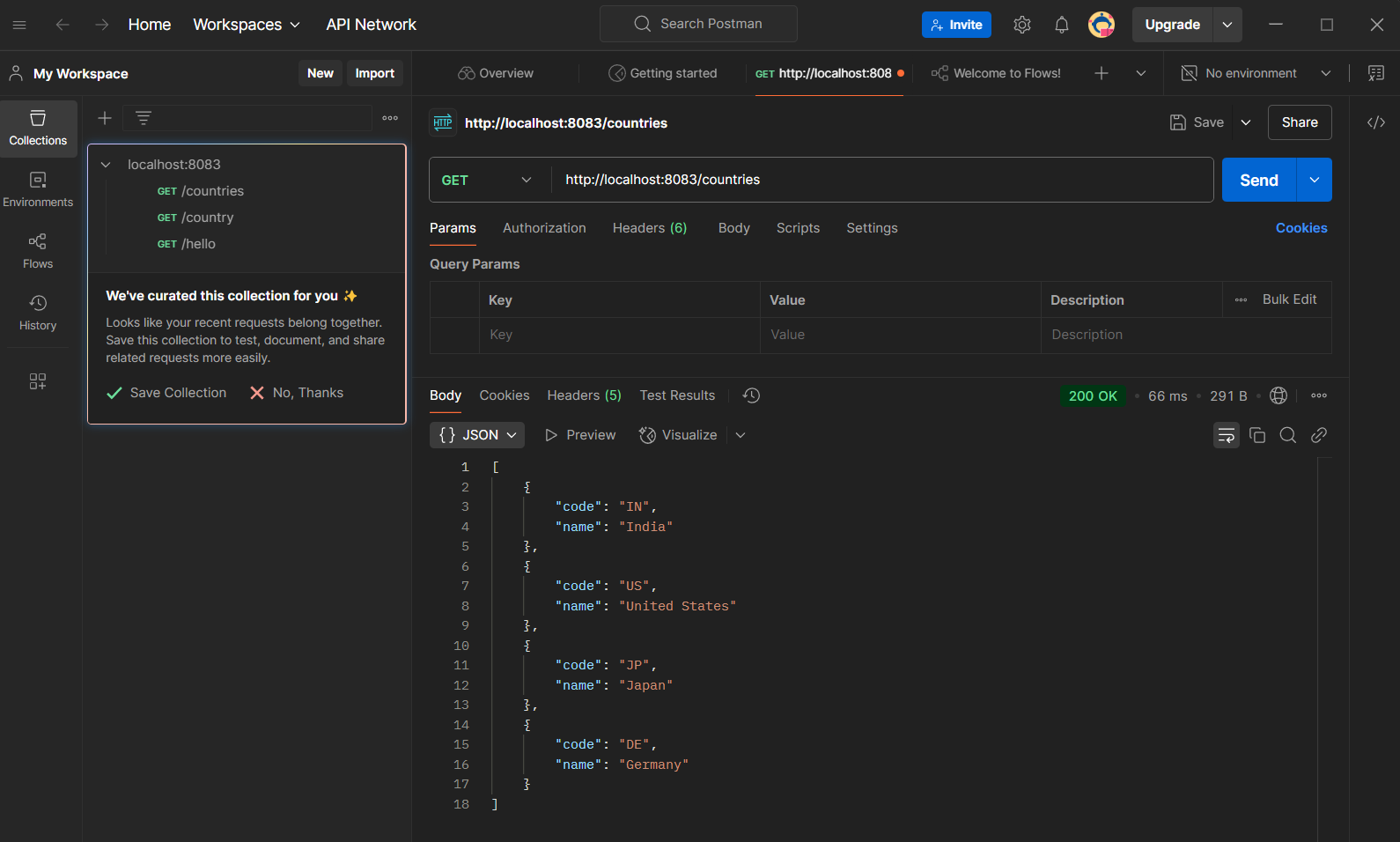
**Postman Outcomes:**

**REST - Country Web Service :**

**A screenshot of a computer

AI-generated content may be incorrect.**

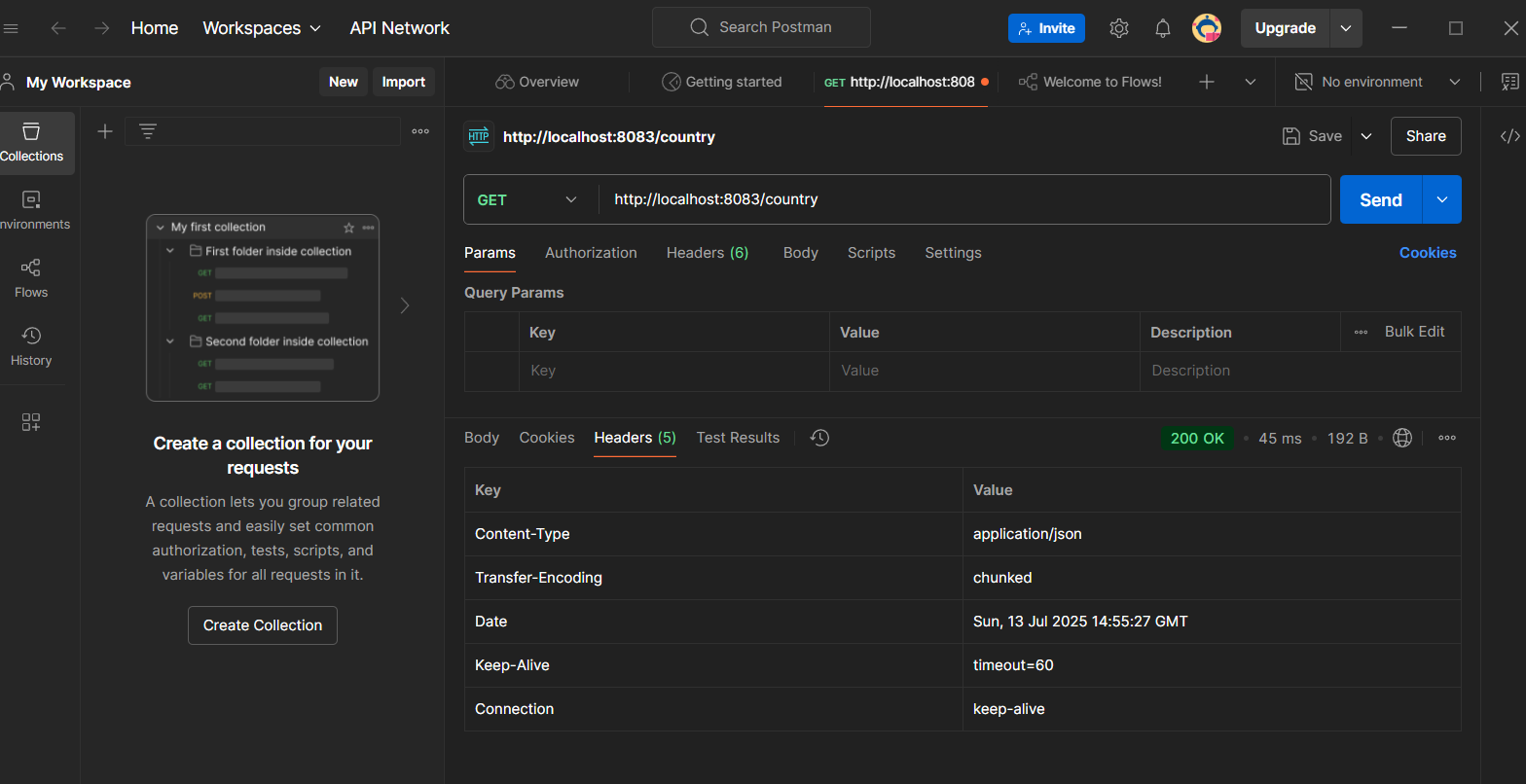
**REST - Get all countries:**

****

**HEADER DETAILS:**

**A screenshot of a computer

AI-generated content may be incorrect.**

****

**Ques 1) What happens in the controller method?**

**Ans) The controller method loads the bean from the Spring XML configuration (country.xml) using ApplicationContext and returns it as a response.**

**Ques 2) How is the bean converted into JSON response?**

**Ans ) Spring Boot automatically uses the Jackson library (included in spring-boot-starter-web) to convert the returned Java object (bean) into a JSON response.**