**WEEK – 4 : HANDS-ON EXERCISE**

**Spring REST using SpringBoot3**

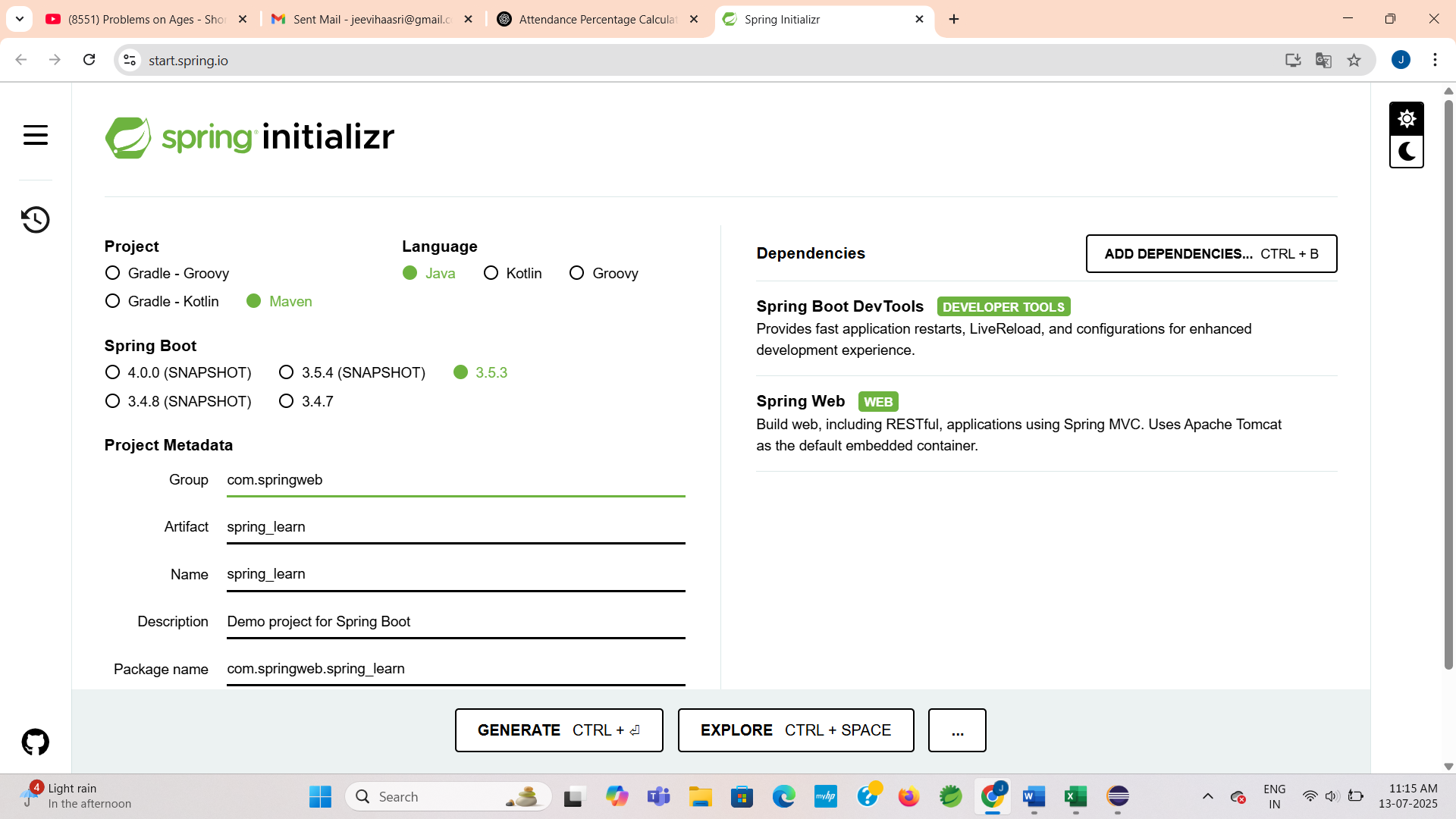
**Exercise 1: Create a Spring Web Project using Maven**

**Scenario:**

Create a Spring Web project using Maven and verify it runs successfully.

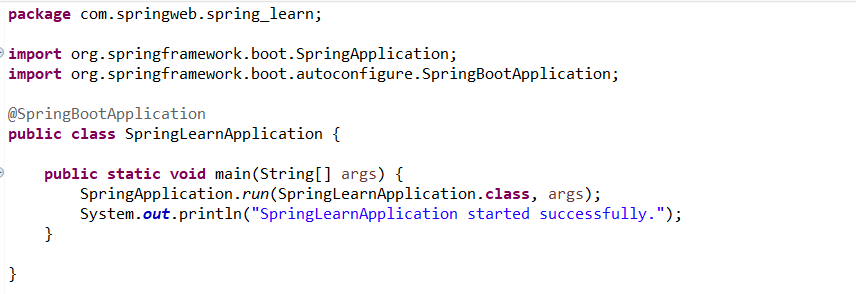
**Step 1: Project Setup**

1. Go to <https://start.spring.io/>
2. Change Group to ‘com.cognizant’
3. Change Artifact to ‘spring-learn’
4. Select the following dependencies:
   * Spring Boot DevTools
   * Spring Web
5. Click Generate to download the project as a ZIP file.
6. Extract the ZIP file into your Eclipse workspace root folder.



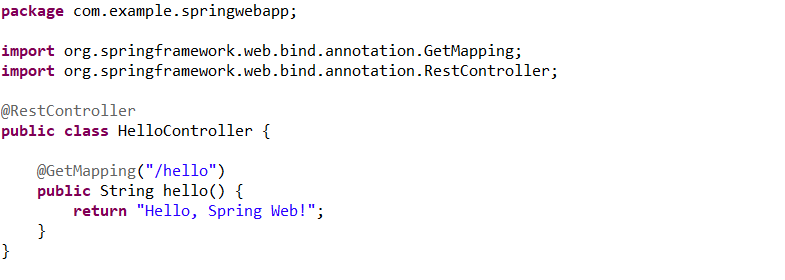
**SpringLearnApplication.java**

**CODE:**



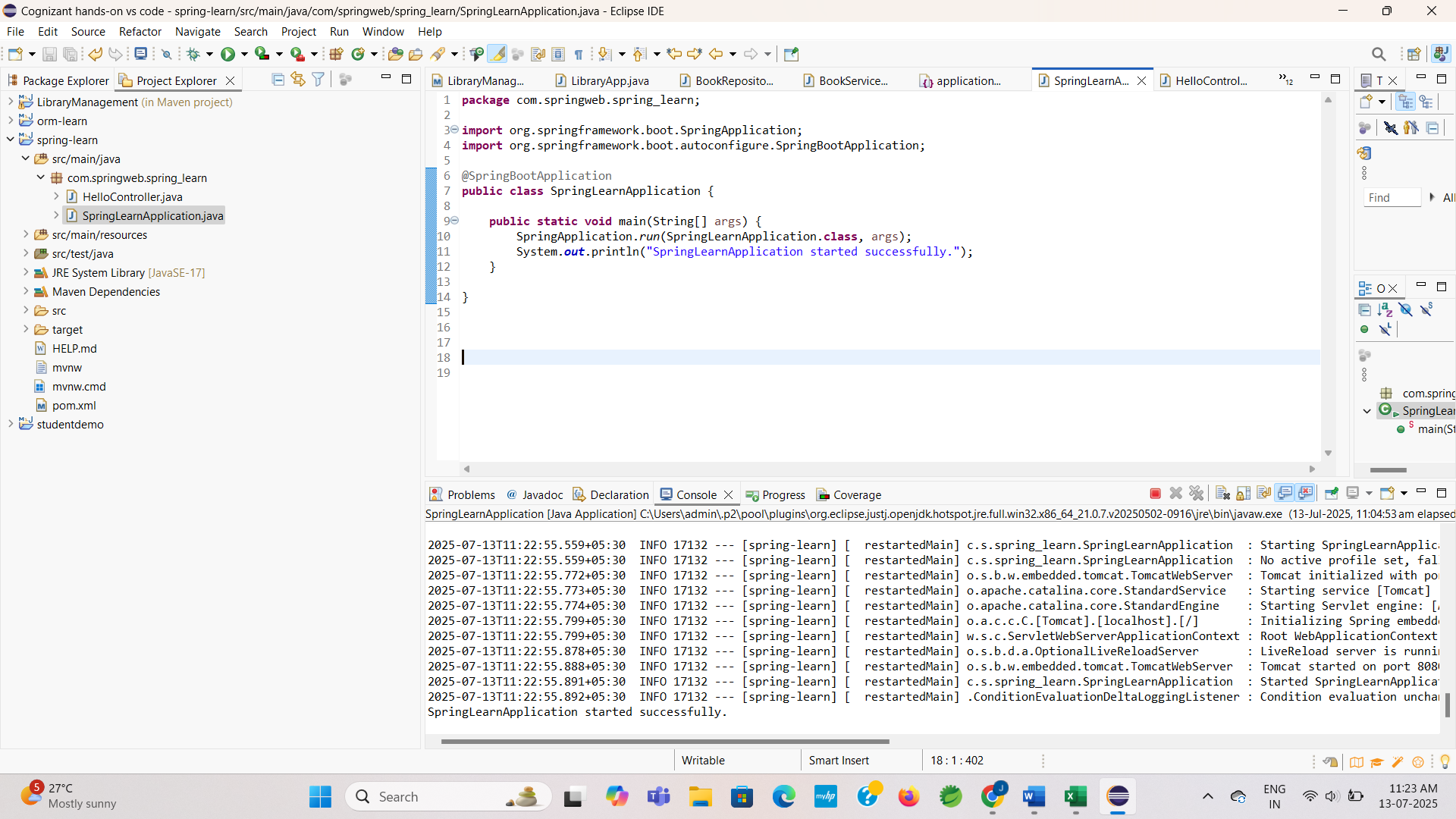
**HelloController.java**

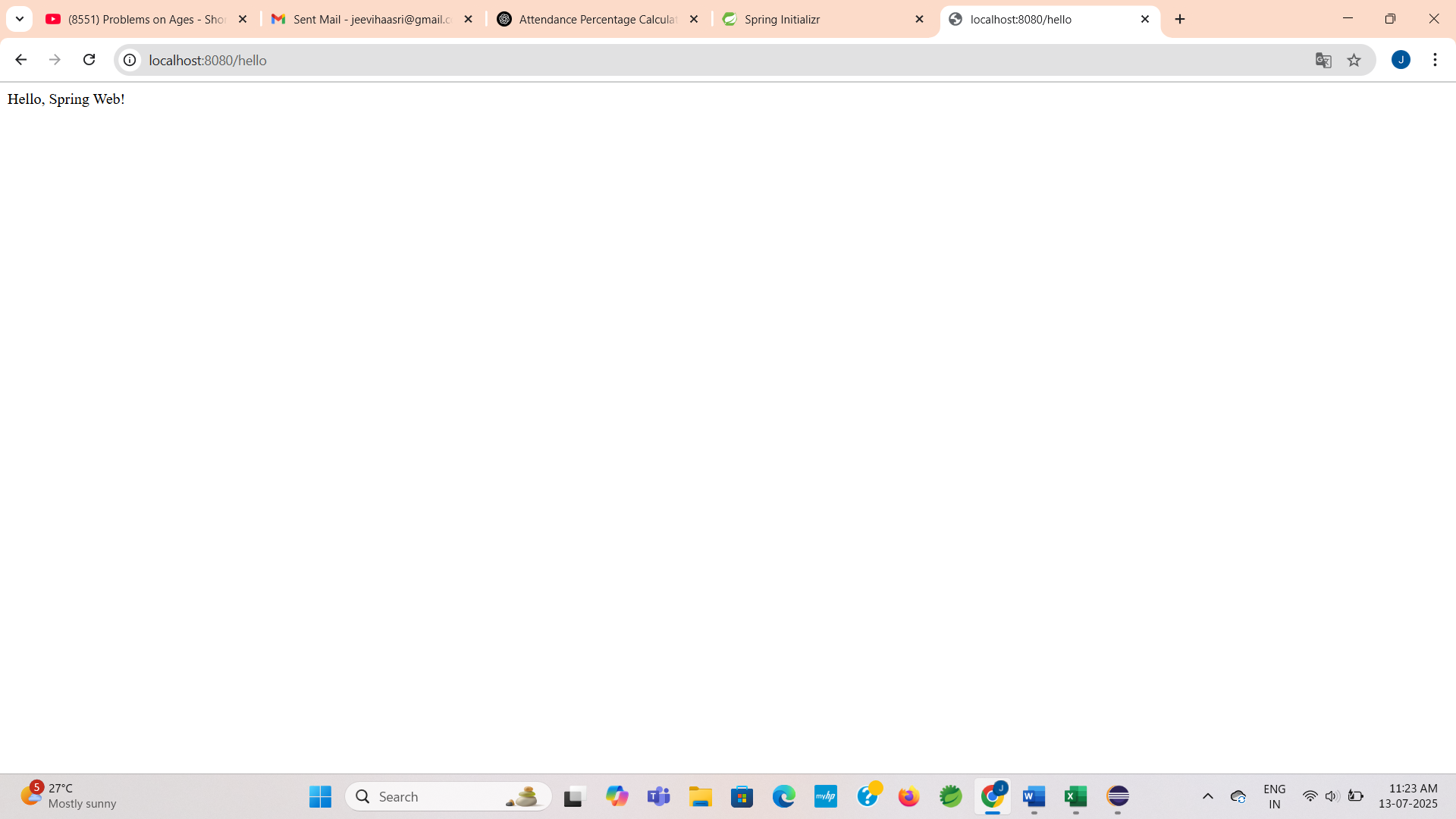
**CODE:**



**Step 4: Running the Application**

**OUTPUT:**





**Exercise 2: Spring Core – Load Country from Spring Configuration XML**

**Scenario:**

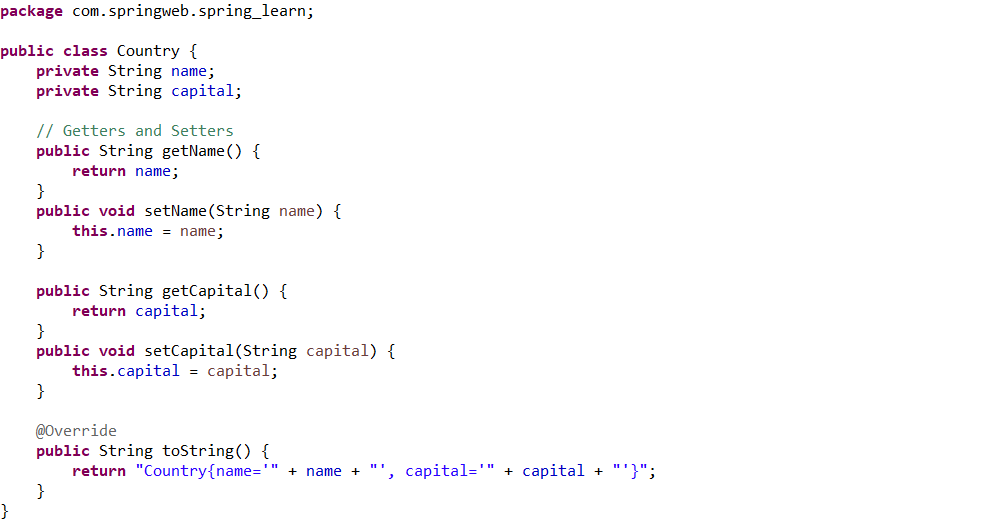
Centralize the creation of a Country object using Spring XML configuration and load it through the Spring application context.

**Step 1: Create a New Maven Project**

This step was already done in Exercise 1. We are continuing the same in this project.

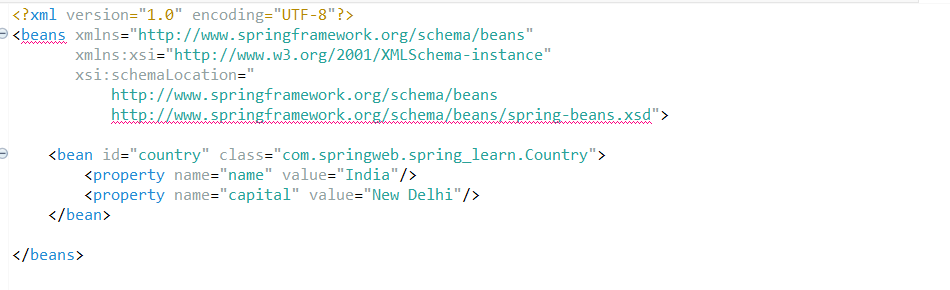
**Country.java:**

**CODE:**



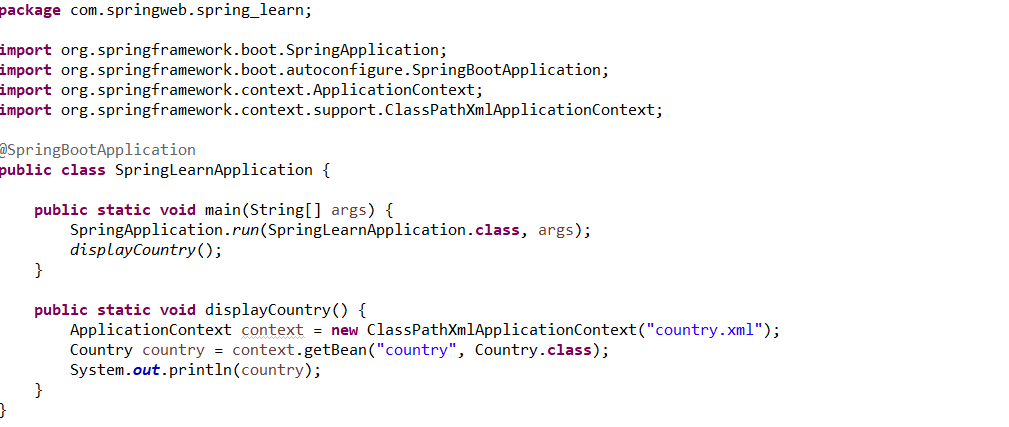
**Country.xml – XML Bean Configuration:**

**CODE:**



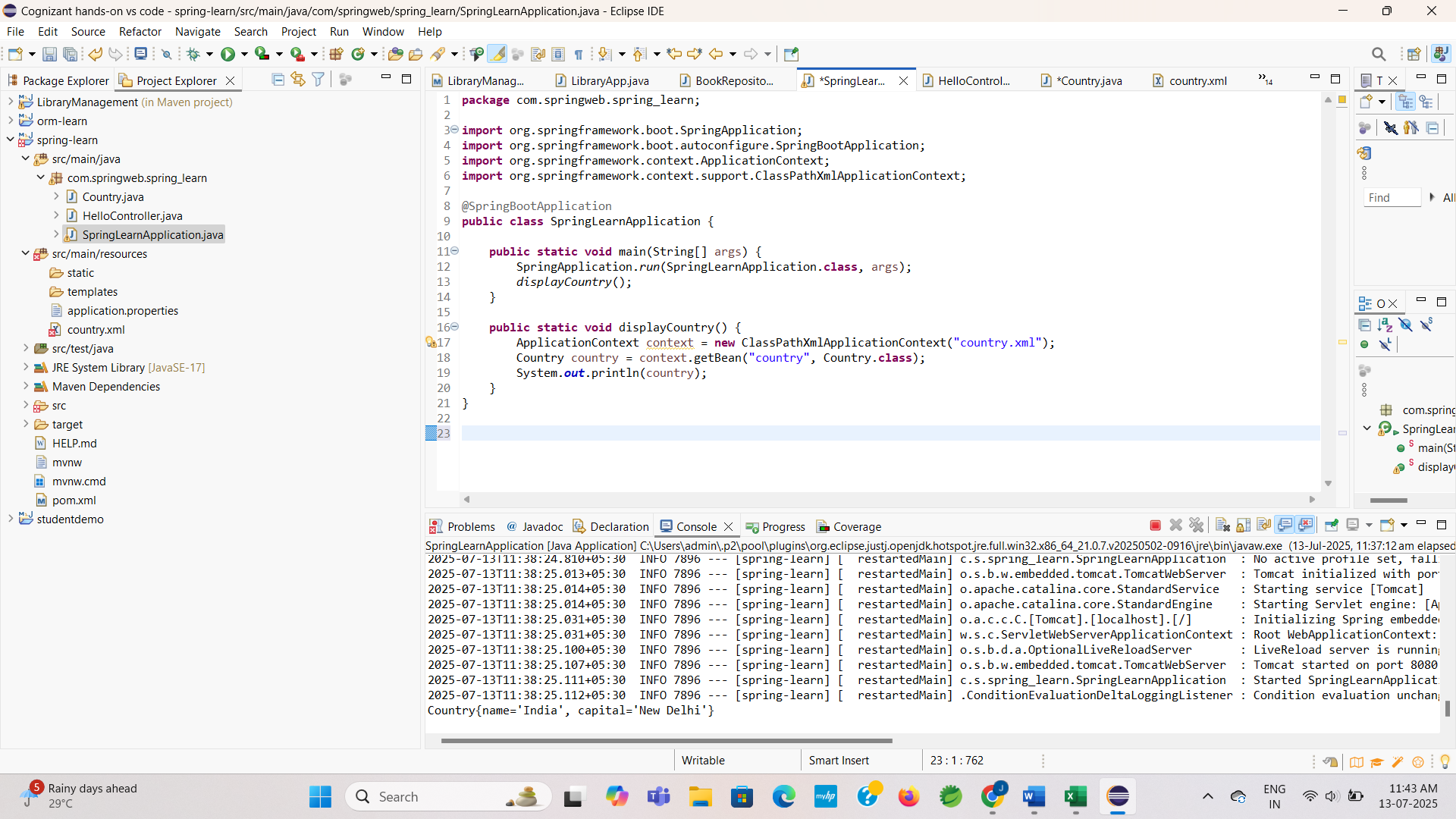
**SpringLearnApplication.java:**

**CODE:**



**Step 3: Run the Application**

**OUTPUT:**



**Exercise 3: Hello World RESTful Web Service using Spring Web**

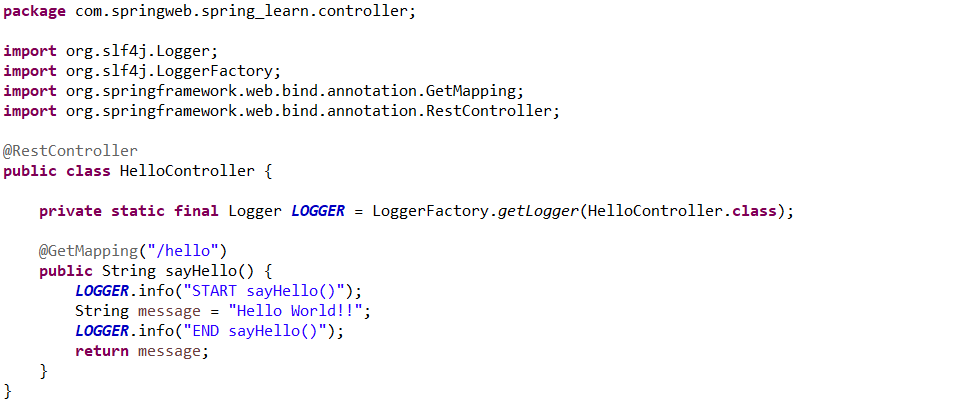
**Scenario:**

Create a RESTful API that returns "Hello World!!" when accessed at /hello. This helps in understanding how to build a simple REST controller in Spring using the Web module.

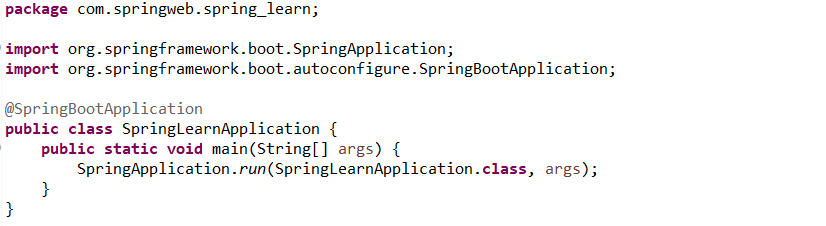
**Step 1: Create a New Maven Project**

* This step was already done in Exercise 1. We are continuing the same in this project.

**HelloController.java:**

****

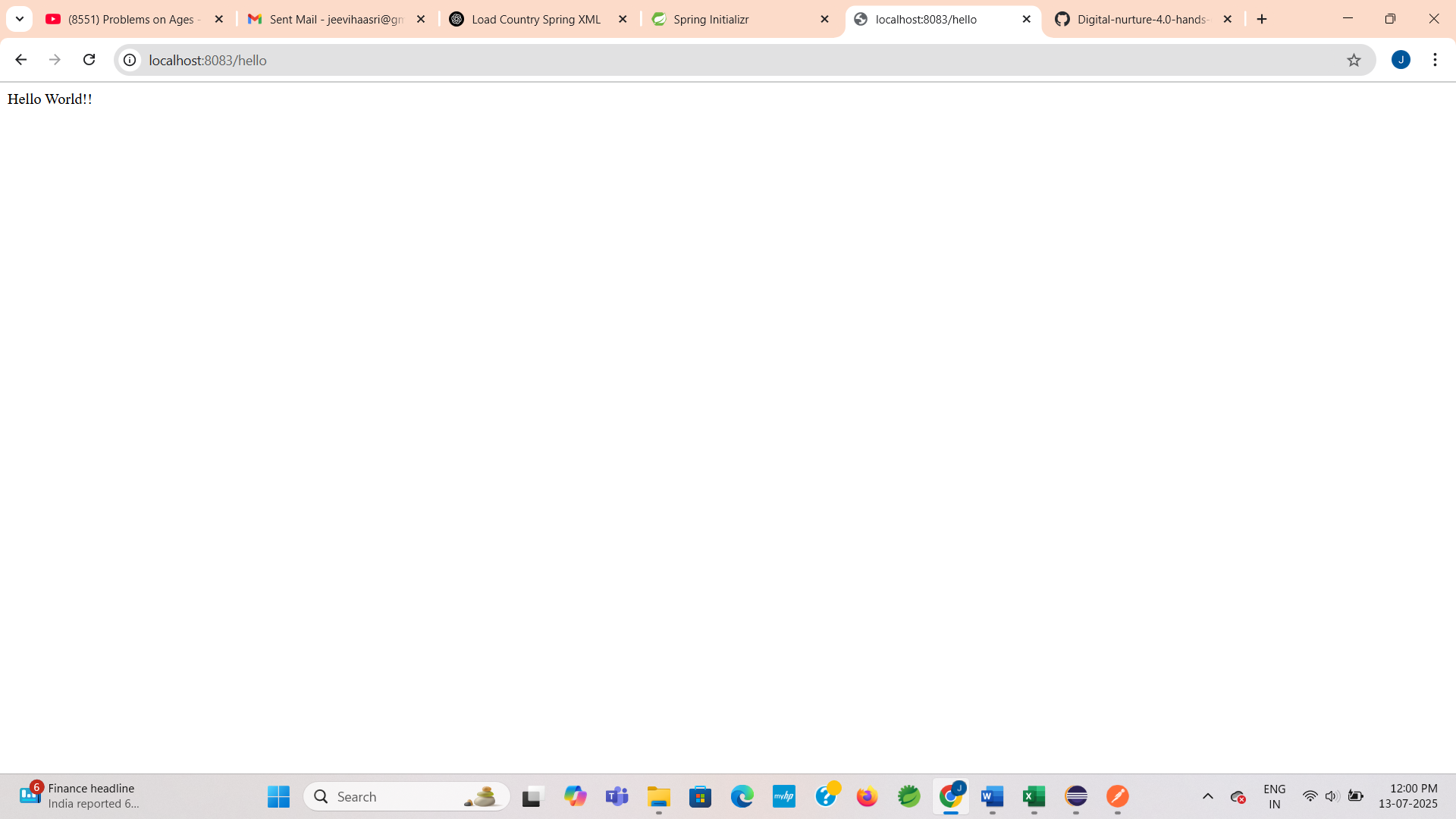
**SpringLearnApplication.java:**

****

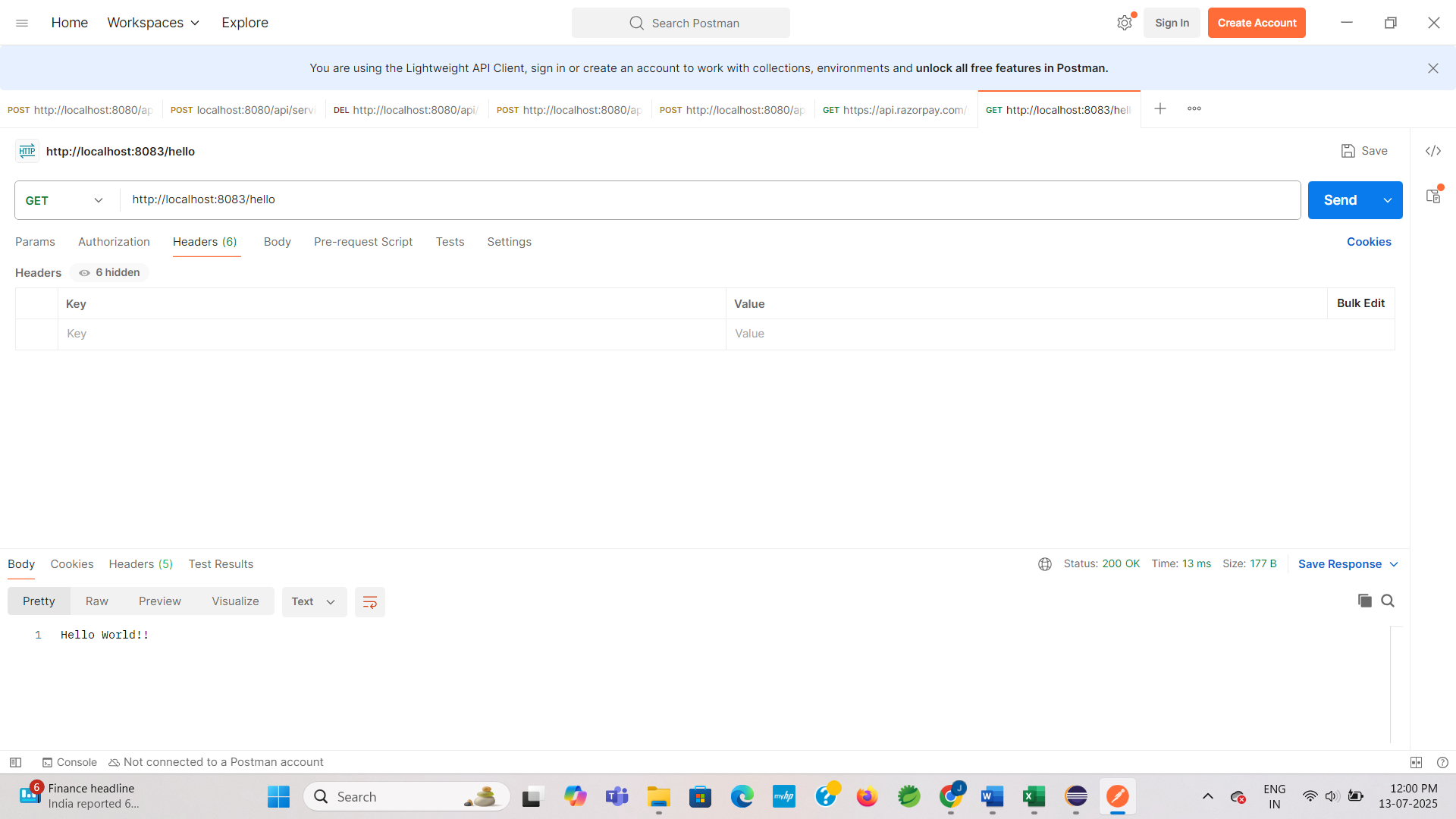
**Step 4: Run the Application**

**OUTPUT:**

**In Chrome browser:**



**In Postman:**



**Exercise 4: REST – Country Web Service using Spring Web**

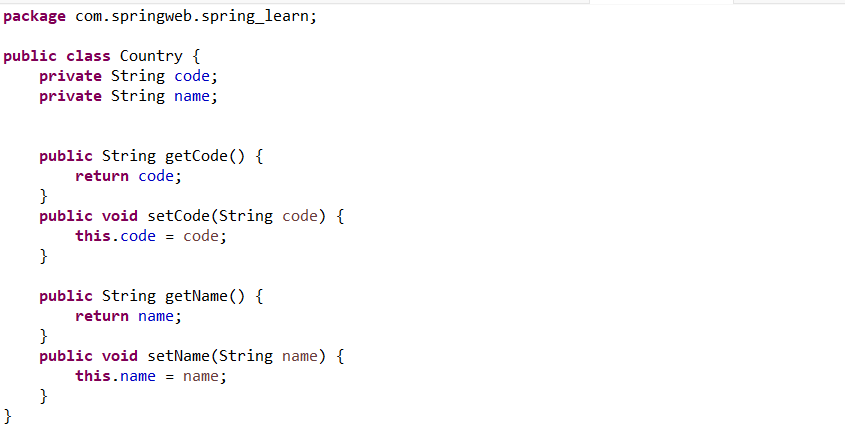
**Scenario:**

Create a REST API in the existing Spring Learn application to return country details for India by loading the Country bean from the Spring XML configuration.

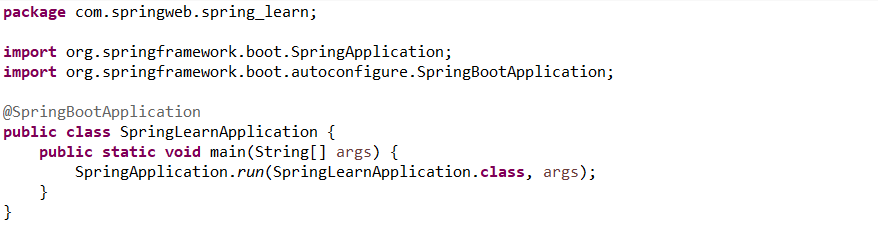
**Step 1: Create a New Maven Project**

* This step was already done in Exercise 1. We are continuing the same in this project.

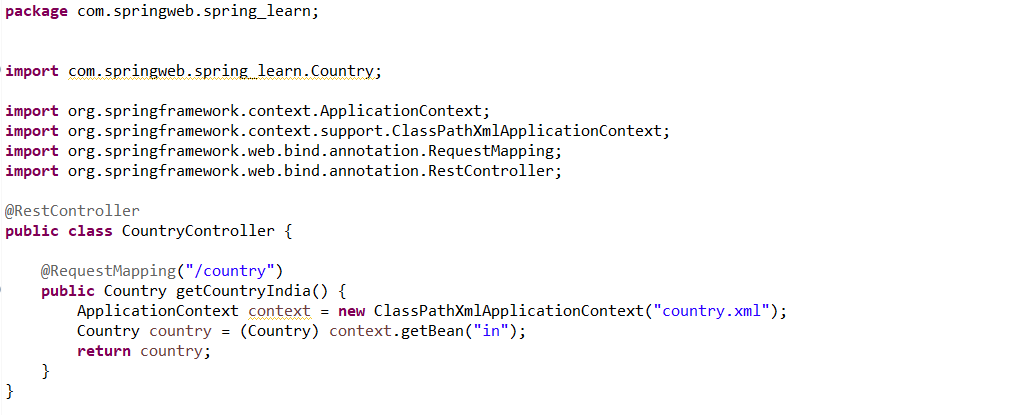
**Country.java:**

****

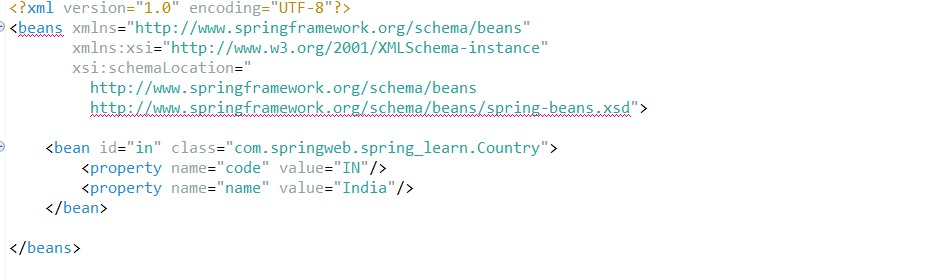
**SpringLearnApplication.java:**

****

**CountryController.java:**

****

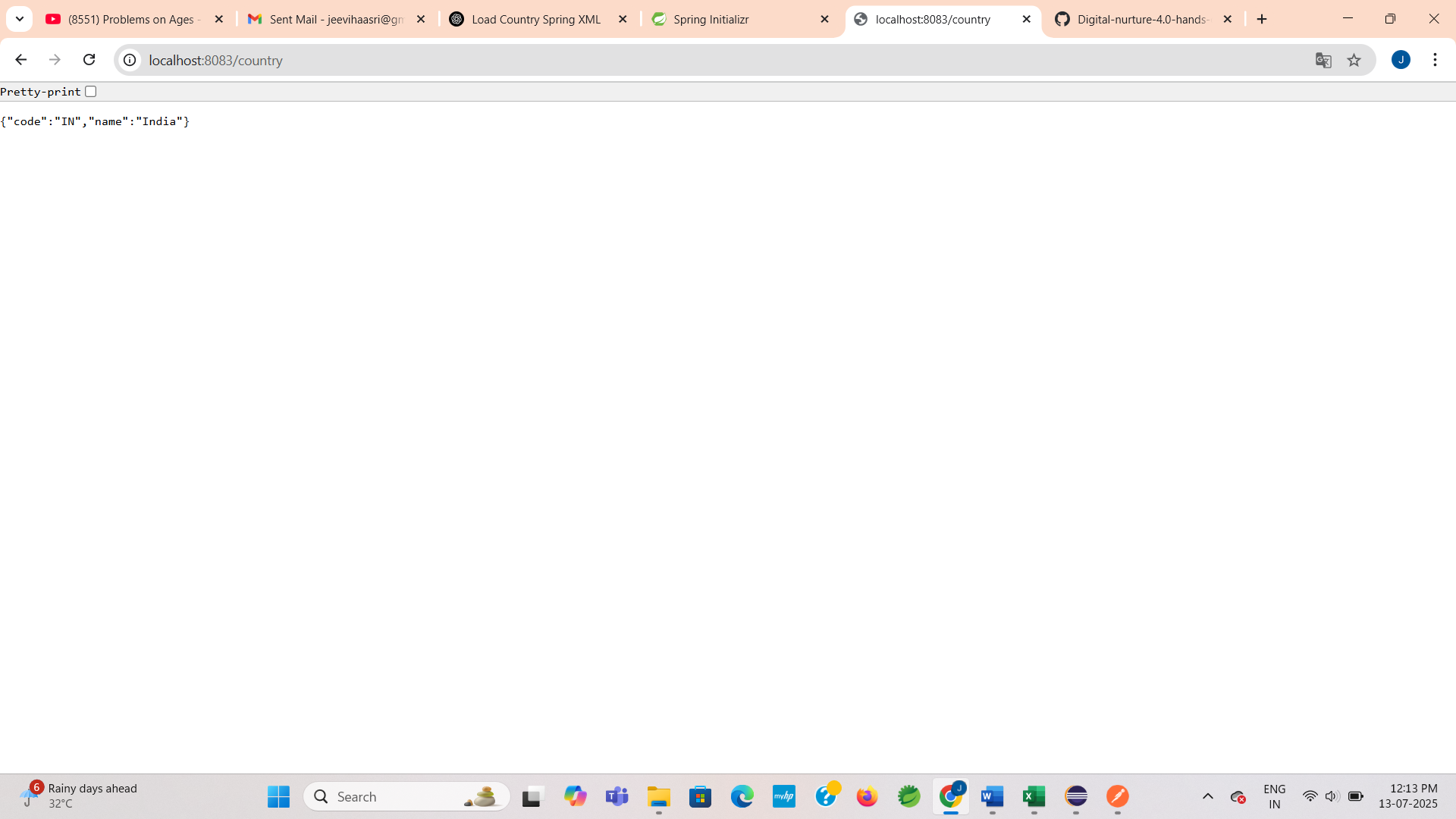
**Country.xml:**

****

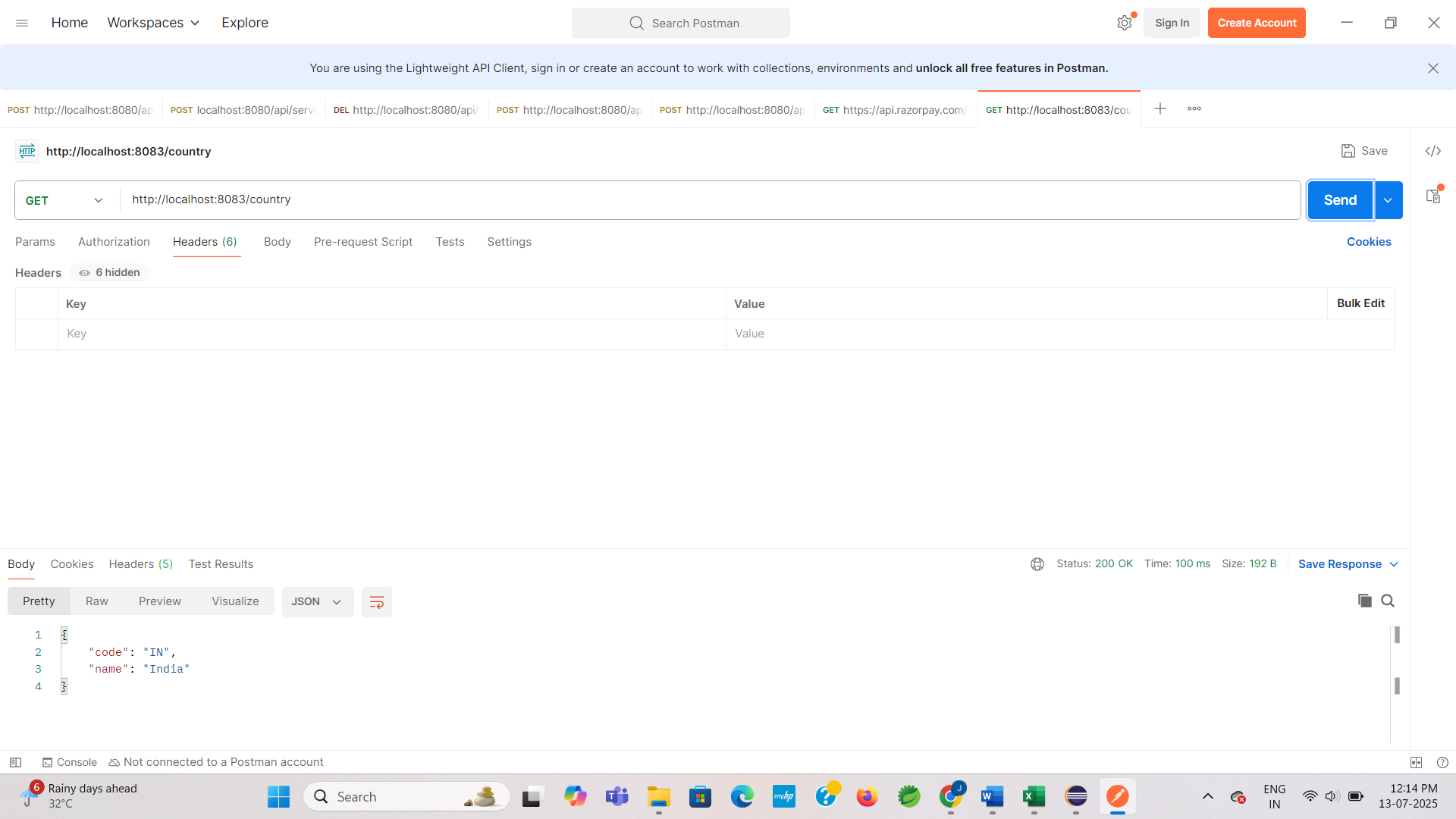
**Step 4: Run the Application**

**OUTPUT:**

**In Chrome Browser:**



**In Postman:**



**Exercise 5: REST – Get Country by Code**

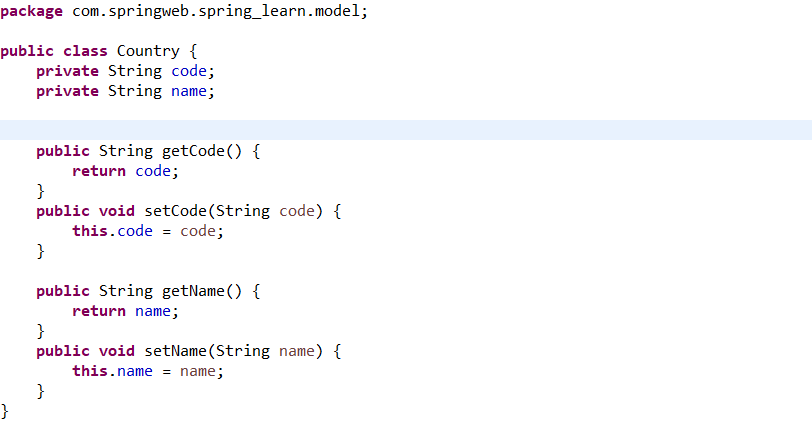
**Scenario:**

Create a Spring REST API that returns the details of a country based on the country code (e.g., IN, in, In). The country data will be loaded from country.xml.

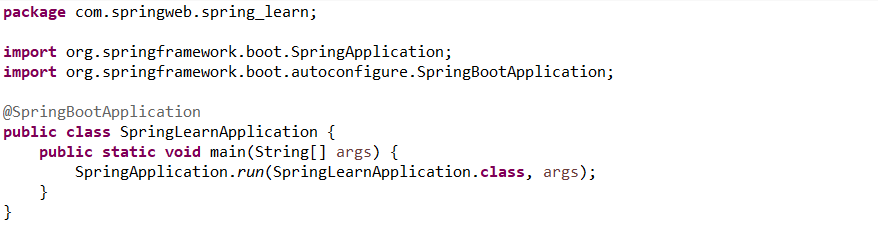
**Step 1: Create a New Maven Project**

* This step was already done in Exercise 1. We are continuing the same in this project.

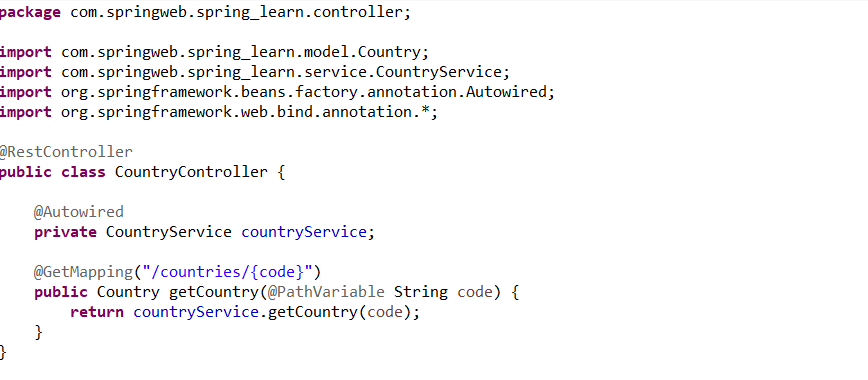
**Country.java:**

****

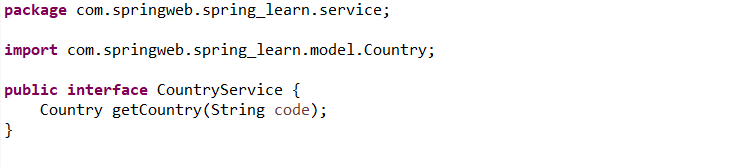
**SpringLearnApplication.java:**

****

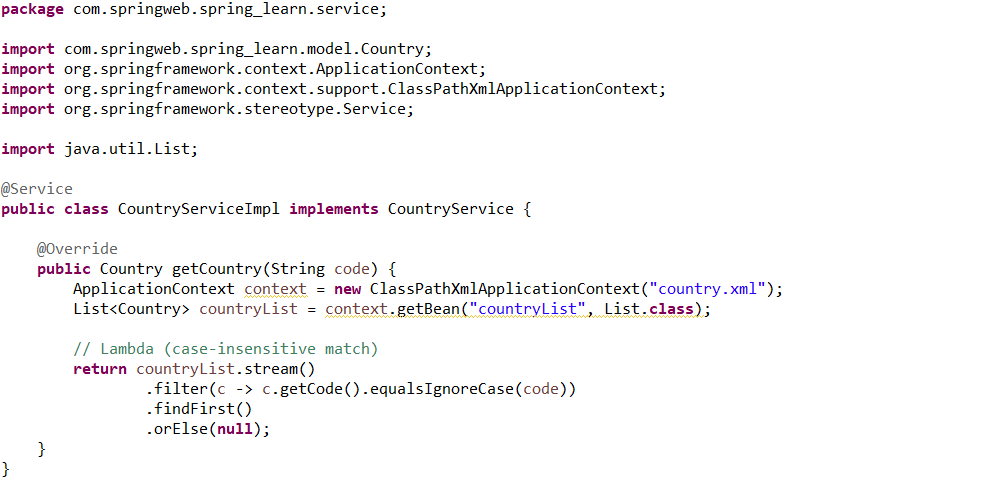
**CountryController.java:**

****

**CountryService.java:**

****

**CountryServiceImpl.java:**

****

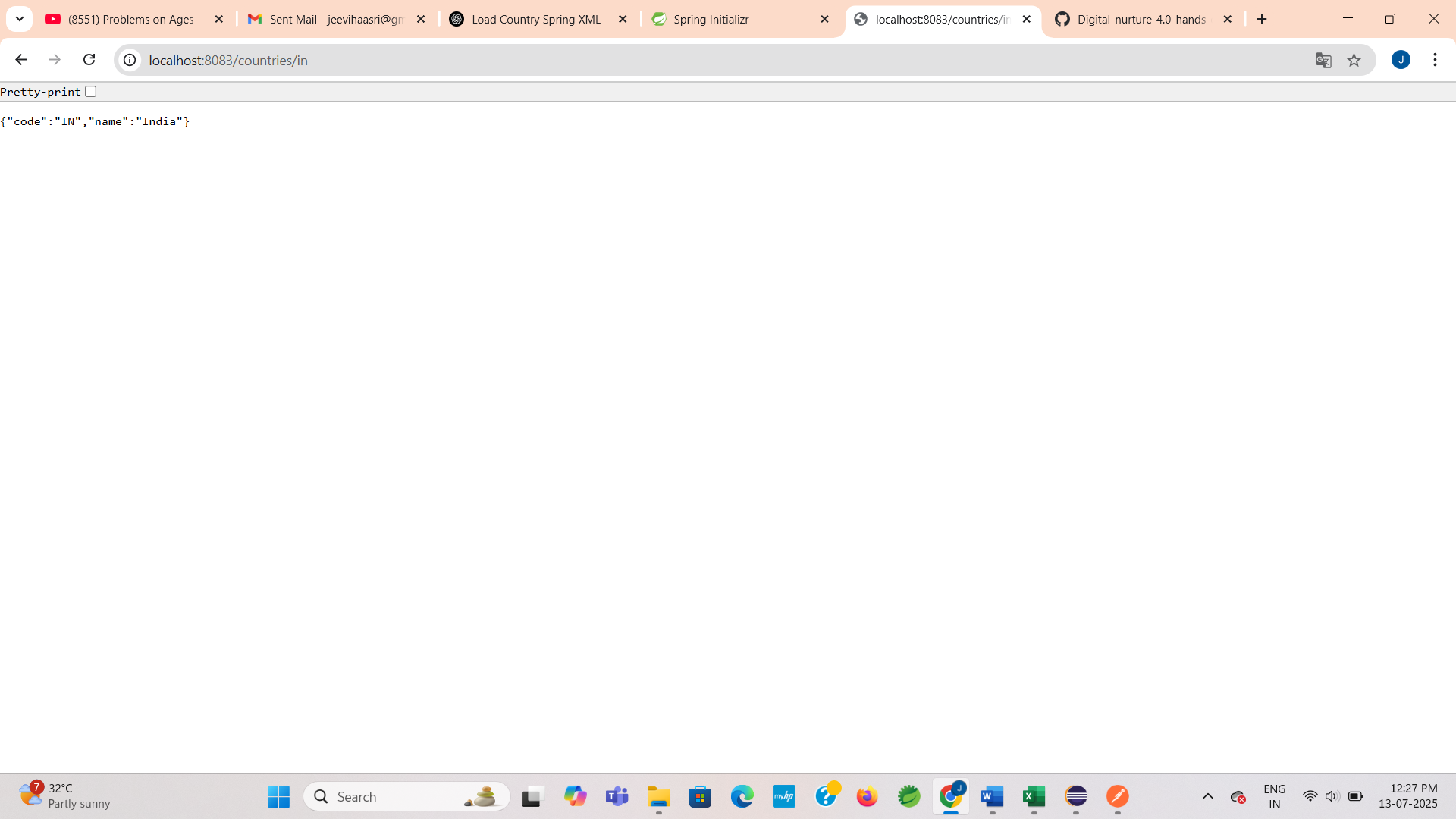
**Country.xml:**

****

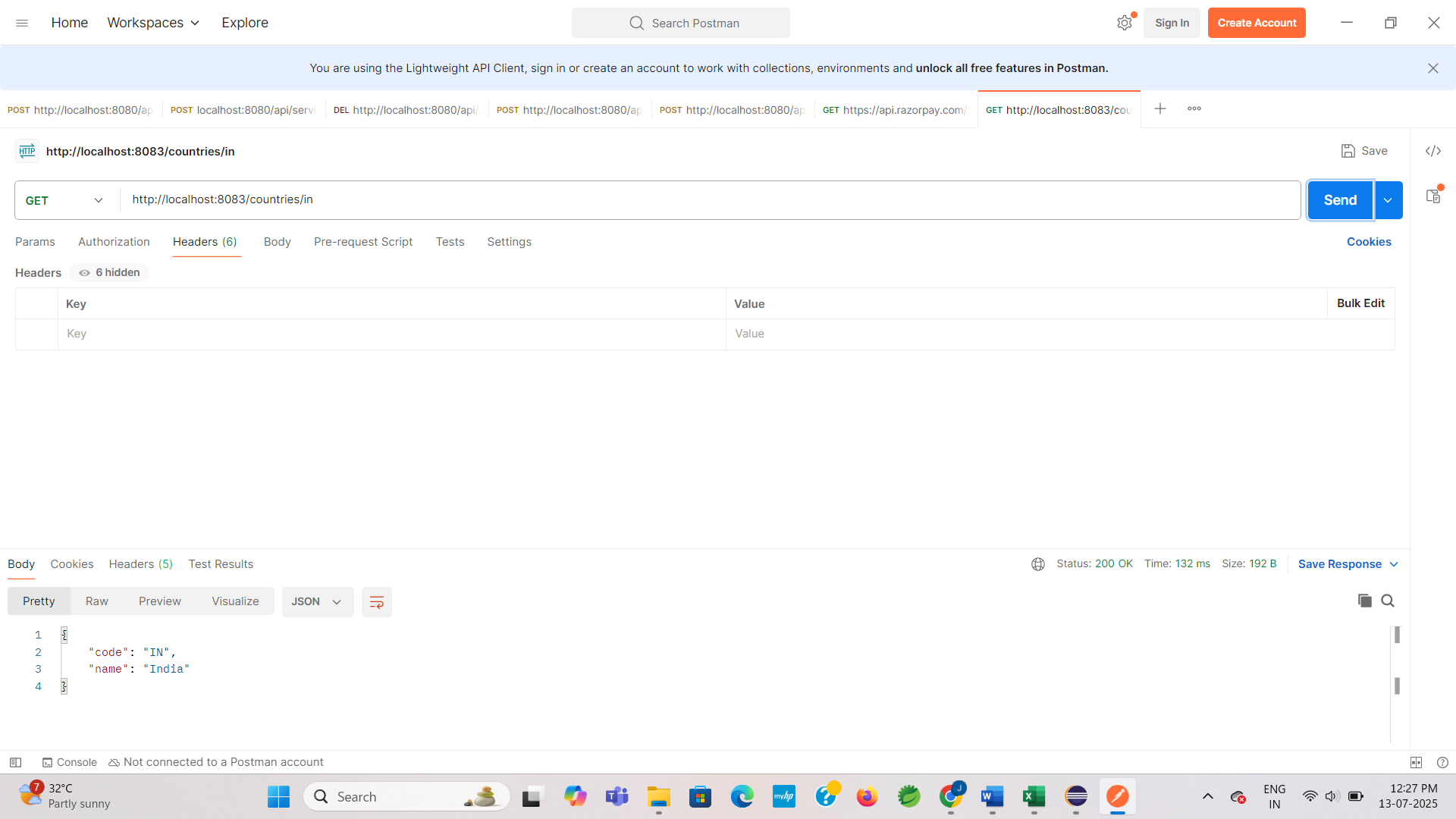
**Step 4: Run the Application**

**OUTPUT:**

**In Chrome Browser:**



**In Postman:**



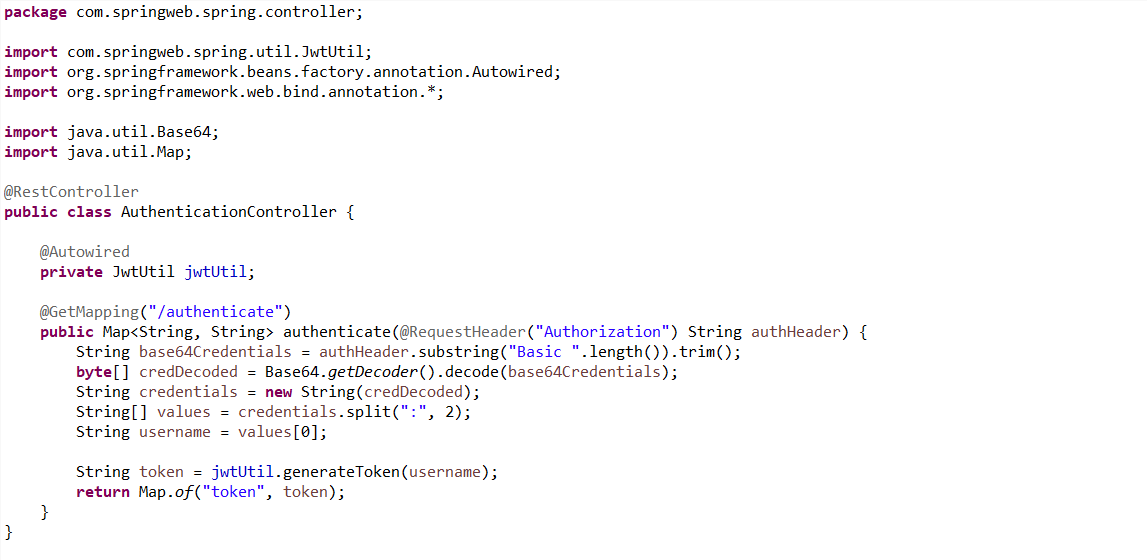
**Exercise 6: Create authentication service that returns JWT**

**Scenario:** Secure the /hello endpoint so only users with a valid JWT token can access it.

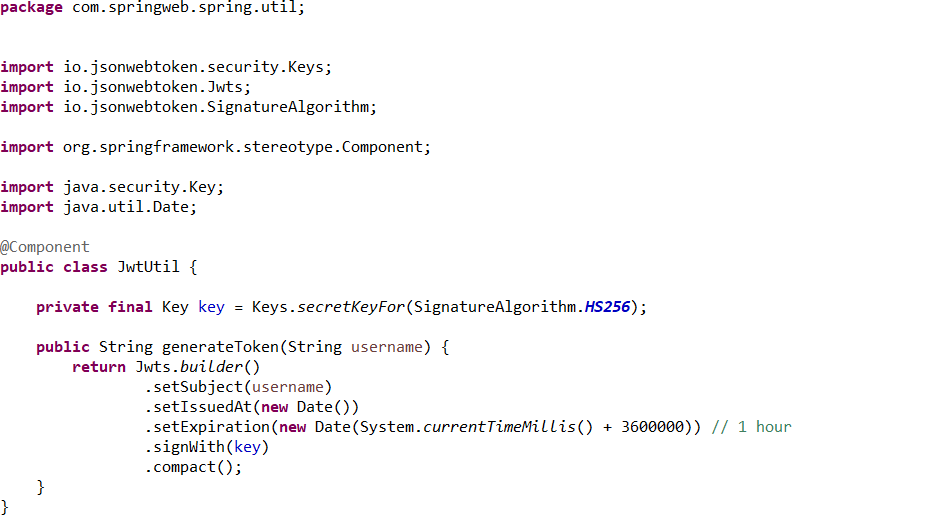
**Step 1: Create a New Maven Project**

* This step was already done in Exercise 1. We are continuing the same in this project.

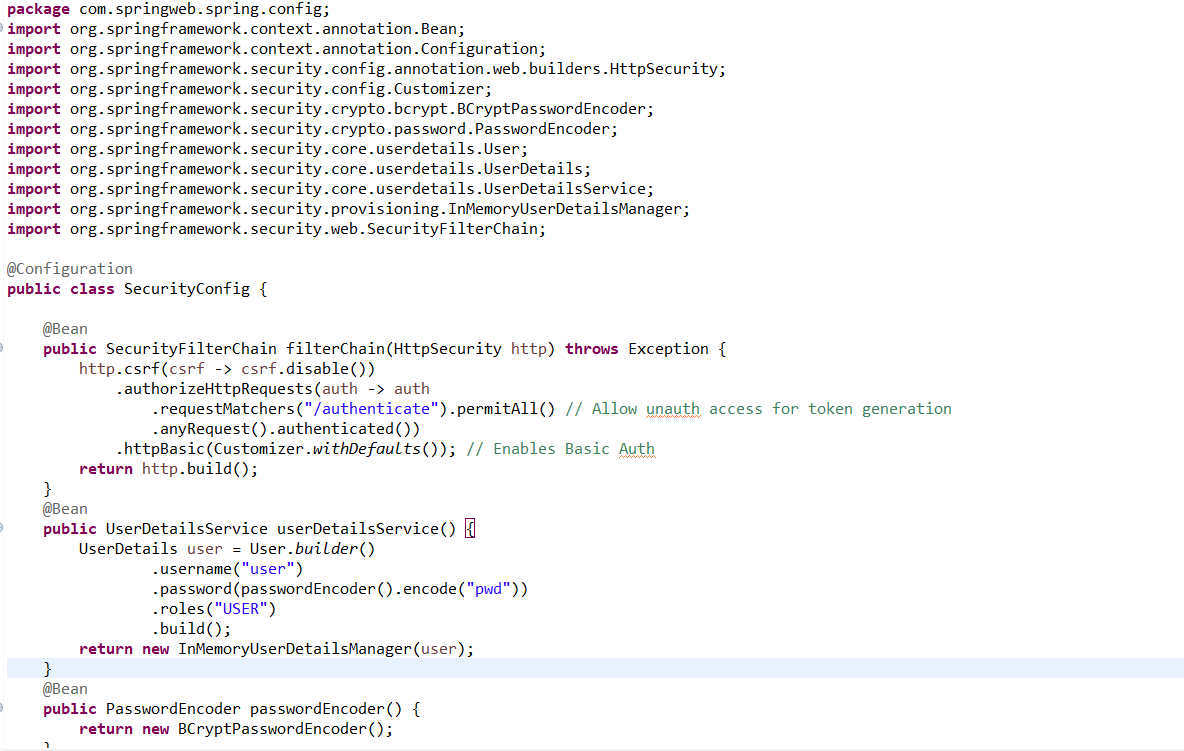
**AuthenticationController.java:**

****

**JwtUtil.java:**

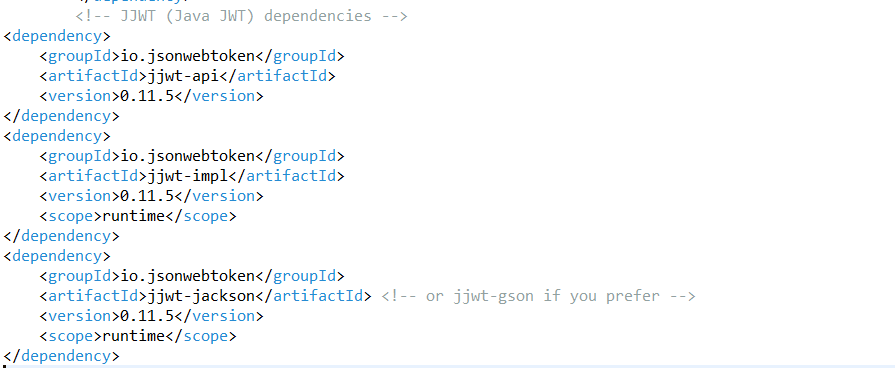
****

**SecurityConfig.java:**

****

**Pom.xml:**

* Add the JWT dependencies in pom.xml

****

**Step 4: Run the Application**

**OUTPUT:**

**In Postman:**

