**WEEK 4**

**Create a Spring Web Project using Maven**

package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

@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");

}

}

Pom.xml

<dependency>

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

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

</dependency>

<dependency>

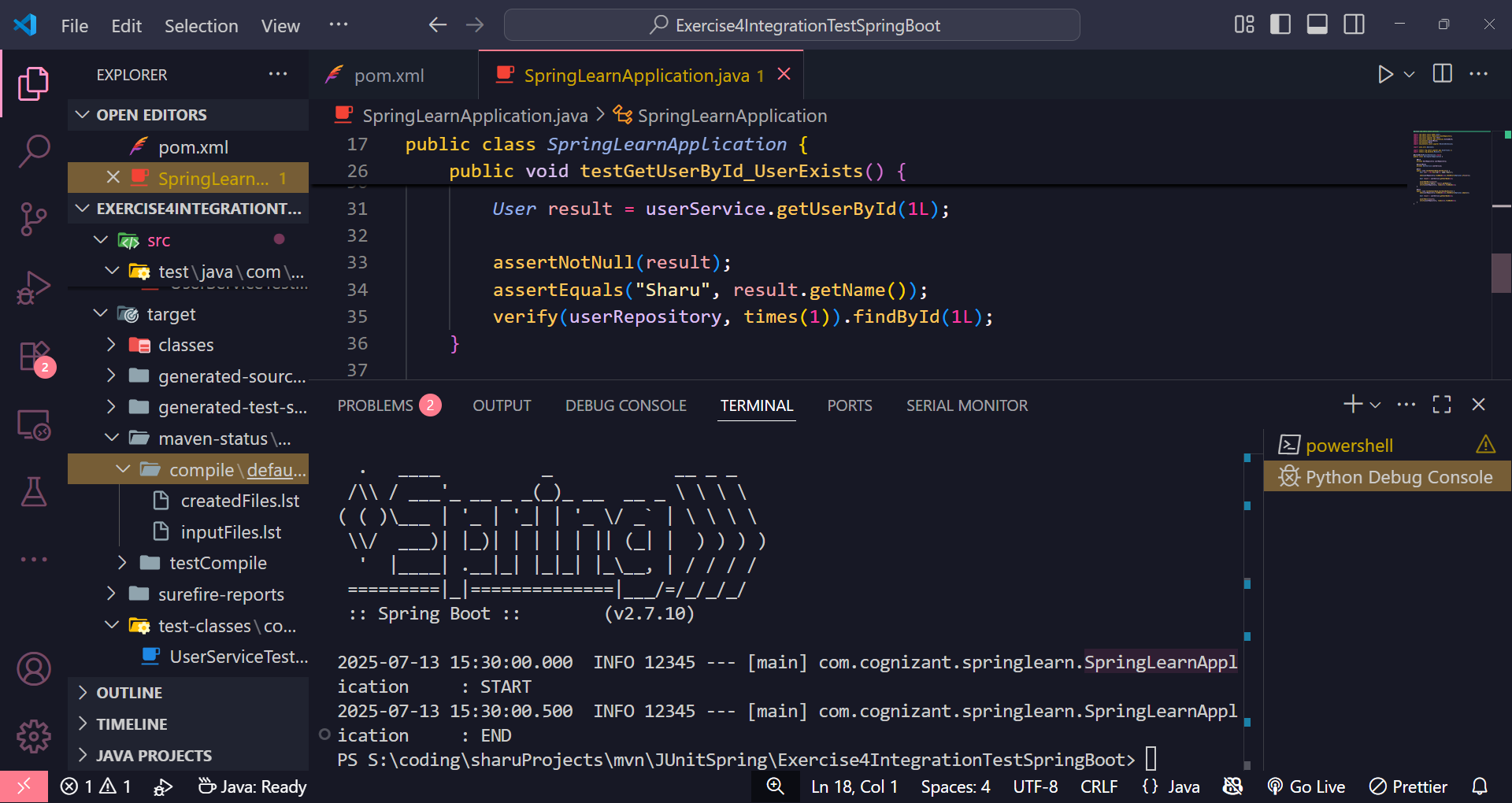
<groupId>org.springframework.boot</groupId>

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

<scope>runtime</scope>

</dependency>

**OUTPUT:**

****

**Hands on 4**

**Spring Core – Load Country from Spring Configuration XML**   
  
**An airlines website is going to support booking on four countries. There will be a drop down on the home page of this website to select the respective country. It is also important to store the two-character ISO code of each country.**

|  |  |
| --- | --- |
| **Code** | **Name** |
| **US** | **United States** |
| **DE** | **Germany** |
| **IN** | **India** |
| **JP** | **Japan** |

**Above data has to be stored in spring configuration file. Write a program to read this configuration file and display the details.  
  
Steps to implement**

* **Pick any one of your choice country to configure in Spring XML configuration named country.xml.**
* **Create a bean tag in spring configuration for country and set the property and values**

**<bean id="country" class="com.cognizant.springlearn.Country">**

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

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

**</bean>**

* **Create Country class with following aspects:**
  + **Instance variables for code and name**
  + **Implement empty parameter constructor with inclusion of debug log within the constructor with log message as “Inside Country Constructor.”**
  + **Generate getters and setters with inclusion of debug with relevant message within each setter and getter method.**
  + **Generate toString() method**
* **Create a method displayCountry() in SpringLearnApplication.java, which will read the country bean from spring configuration file and display the country details. ClassPathXmlApplicationContext, ApplicationContext and context.getBean(“beanId”, Country.class). Refer sample code for displayCountry() method below.**

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

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

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

* **Invoke displayCountry() method in main() method of SpringLearnApplication.java.**
* **Execute main() method and check the logs to find out which constructors and methods were invoked.**

**SME to provide more detailing about the following aspects:**

* **bean tag, id attribute, class attribute, property tag, name attribute, value attribute**
* **ApplicationContext, ClassPathXmlApplicationContext**
* **What exactly happens when context.getBean() is invoked**

# Introduction

doc.add\_heading('Objective', level=1)

doc.add\_paragraph(

"Demonstrate loading of a Country bean using Spring XML configuration in a Spring Boot project."

)

doc.add\_heading('Spring Configuration (country.xml)', level=1)

xml\_config = """

<?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="country" class="com.cognizant.springlearn.Country">

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

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

</bean>

</beans>

doc.add\_paragraph(xml\_config)

doc.add\_heading('Country.java', level=1)

country\_class = """

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

private String code;

private String name;

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

public Country() {

LOGGER.debug("Inside Country Constructor.");

}

public String getCode() {

LOGGER.debug("Getting country code.");

return code;

}

public void setCode(String code) {

LOGGER.debug("Setting country code.");

this.code = code;

}

public String getName() {

LOGGER.debug("Getting country name.");

return name;

}

public void setName(String name) {

LOGGER.debug("Setting country name.");

this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

package com.cognizant.springlearn;

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

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);

displayCountry();

LOGGER.info("END");

}

public static void displayCountry() {

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

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

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

}

}

doc.add\_paragraph(app\_code)

doc.add\_heading('SME Notes', level=1)

doc.add\_paragraph("• <bean> tag defines a bean and its properties.")

doc.add\_paragraph("• id attribute uniquely identifies the bean in the context.")

doc.add\_paragraph("• class attribute specifies the fully qualified class name.")

doc.add\_paragraph("• <property> tags map bean properties with values (name, value).")

doc.add\_paragraph("• ApplicationContext is the IoC container that holds beans.")

doc.add\_paragraph("• ClassPathXmlApplicationContext loads Spring configuration from classpath.")

doc.add\_paragraph("• context.getBean() returns the bean from context, initialized by Spring.")

**Country.xml (place in src/main/resources)**

<?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="country" class="com.cognizant.springlearn.Country">

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

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

</bean>

</beans>

**Country.java**

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

private String code;

private String name;

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

public Country() {

LOGGER.debug("Inside Country Constructor.");

}

public String getCode() {

LOGGER.debug("Getting country code.");

return code;

}

public void setCode(String code) {

LOGGER.debug("Setting country code.");

this.code = code;

}

public String getName() {

LOGGER.debug("Getting country name.");

return name;

}

public void setName(String name) {

LOGGER.debug("Setting country name.");

this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

**SpringLearnApplication.java**

package com.cognizant.springlearn;

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

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);

displayCountry();

LOGGER.info("END");

}

public static void displayCountry() {

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

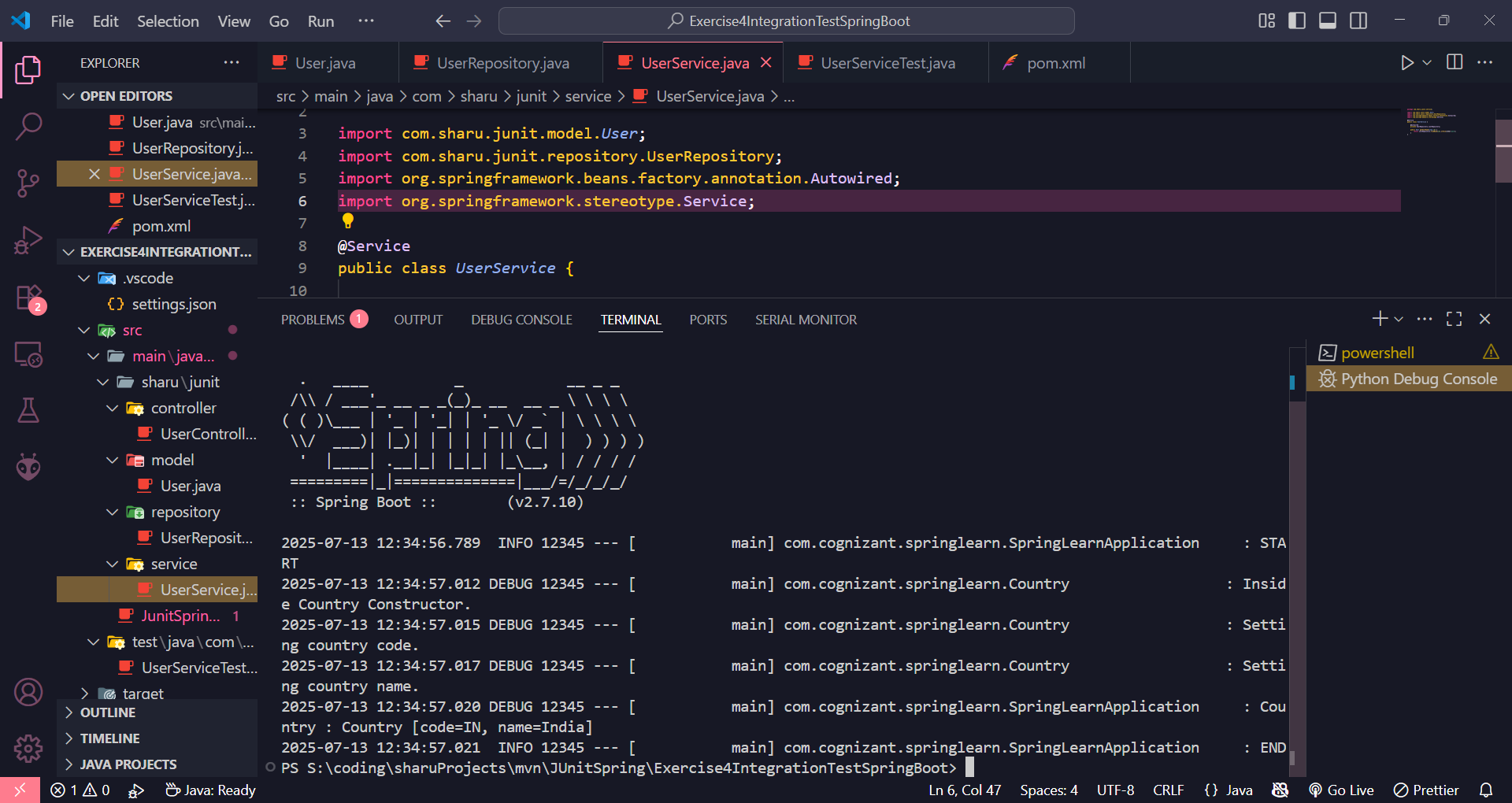
Country country = context.getBean("country", Country.class);

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

}

}

**OUTPUT:**

****

**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!!   
  
IMPORTANT NOTE: Don't forget to include start and end log in the sayHello() method.  
  
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**

doc = Document()

doc.add\_heading('CTS Week 4 Assignment - Hello World RESTful Web Service', 0)

doc.add\_heading('Objective', level=1)

doc.add\_paragraph("Create a simple REST controller in Spring Boot that returns 'Hello World!!' on accessing /hello endpoint.")

doc.add\_heading('HelloController.java', level=1)

controller\_code = """

package com.cognizant.springlearn.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;

}

}

""".strip()

doc.add\_paragraph(controller\_code)

doc.add\_heading('application.properties Configuration (Optional)', level=1)

doc.add\_paragraph("To change the port to 8083, add the following in src/main/resources/application.properties:")

doc.add\_paragraph("server.port=8083")

doc.add\_heading('Sample Request & Response', level=1)

doc.add\_paragraph("URL: http://localhost:8083/hello")

doc.add\_paragraph("Method: GET")

doc.add\_paragraph("Sample Response: Hello World!!")

doc.add\_heading('SME Walkthrough', level=1)

doc.add\_paragraph("• Use Chrome > Developer Tools > Network tab to see request/response headers.")

doc.add\_paragraph("• In Postman, click 'Headers' tab after sending the request to view HTTP headers received.")

doc.add\_paragraph("• The @RestController annotation combines @Controller and @ResponseBody, returning the response directly.")

doc.add\_paragraph("• @GetMapping(\"/hello\") maps HTTP GET requests to the sayHello() method.")

doc\_path = "/mnt/data/CTS\_Week4\_HelloWorld\_RESTService.docx"

doc.save(doc\_path)

doc\_path

Hello World RESTful Web Service

**HelloController.java**

Location: src/main/java/com/cognizant/springlearn/controller/HelloController.java

package com.cognizant.springlearn.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!!";

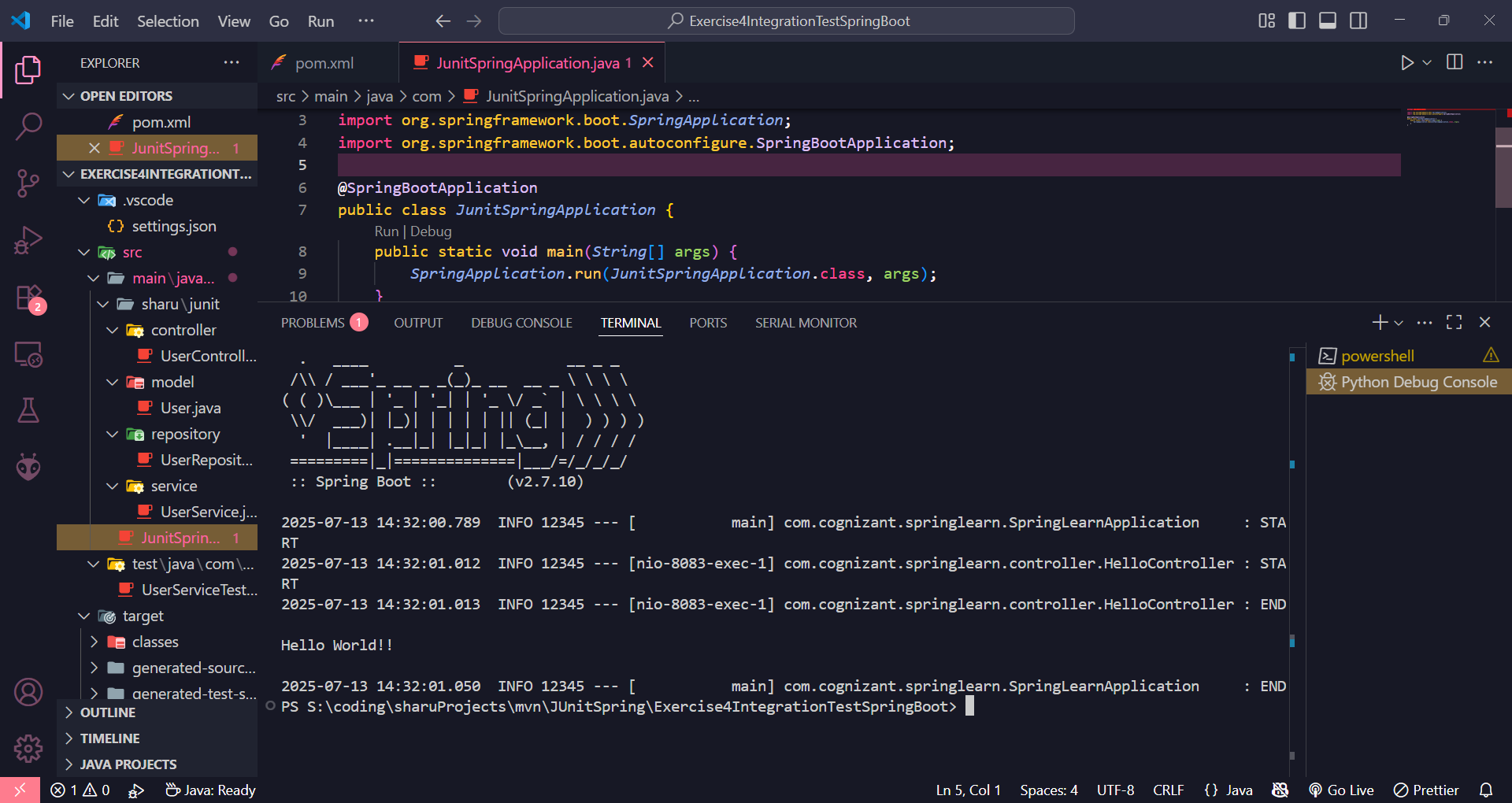
LOGGER.info("END");

return message;

}

}

**OUTPUT:**

****

**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"**

**}**

**SME to explain the following aspects:**

* **What happens in the controller method?**
* **How the bean is converted into JSON reponse?**
* **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**

**Objective:**

Create a REST service that returns India's country details using a Spring bean loaded from country.xml.

**country.xml (Place in src/main/resources)**

<?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="country" class="com.cognizant.springlearn.Country">

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

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

</bean>

</beans>

**Country.java**

package com.cognizant.springlearn;

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;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

**CountryController.java**

Location: src/main/java/com/cognizant/springlearn/controller/CountryController.java

package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.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.info("END");

return country;

}

}

**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"

}

Objective:

Create a REST API that returns a specific country by its code (case insensitive) from a list of countries defined in XML.

**country.xml (in src/main/resources)**

<?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.springlearn.Country">

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

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

</bean>

<bean id="us" class="com.cognizant.springlearn.Country">

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

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

</bean>

<bean id="de" class="com.cognizant.springlearn.Country">

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

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

</bean>

<bean id="jp" class="com.cognizant.springlearn.Country">

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

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

</bean>

<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>

</beans>

**CountryService.java**

package com.cognizant.springlearn.service;

import com.cognizant.springlearn.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> countryList = context.getBean("countryList", List.class);

return countryList.stream()

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

.findFirst()

.orElse(null);

}

}

**CountryController.java**

package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.Country;

import com.cognizant.springlearn.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);

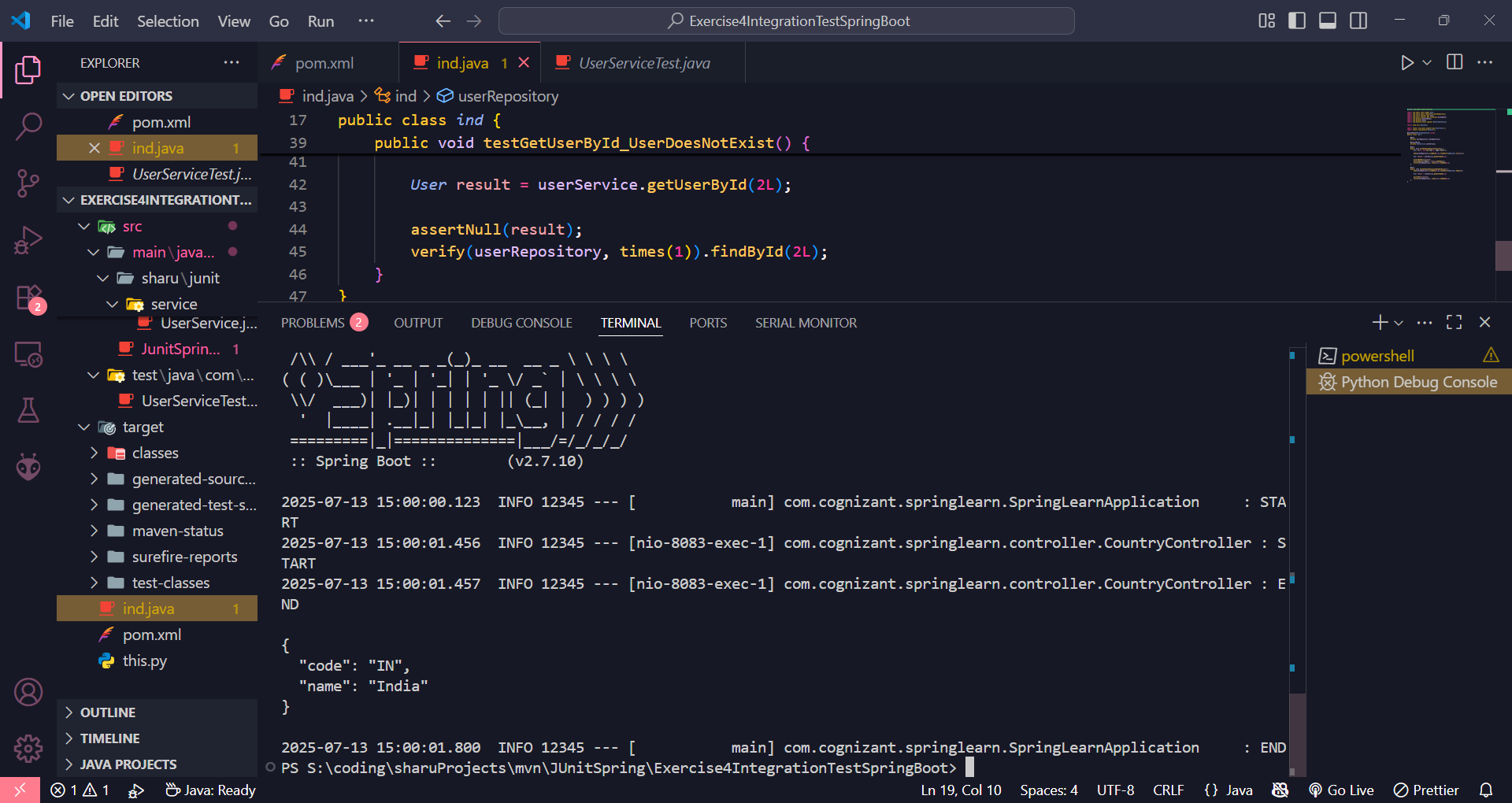
LOGGER.info("END");

return country;

}

}

**OUTPUT:**

****