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

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; }
}
```

country.xml (Spring Configuration in src/main/resources)

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:context="http://www.springframework.org/schema/context"
       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>

</beans>
```

CountryController.java (REST Controller)

```
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.support.ClassPathXmlApplicationContext;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class CountryController {

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

   @GetMapping("/country")
   public Country getCountryIndia() {
      LOGGER.info("Start");

      ClassPathXmlApplicationContext context = new
ClassPathXmlApplicationContext("country.xml");
      Country country = (Country) context.getBean("in");
      context.close();

      LOGGER.info("End");
      return country;
```

```
    }
}
```

Main Class

```
package com.cognizant.spring_learn;


import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication
public class SpringLearnApplication {
    public static void main(String[] args) {
        SpringApplication.run(SpringLearnApplication.class, args);
    }
}
```

COMMAND:

```
curl -s http://localhost:8083/country
```

OUTPUT:

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

SME to explain the following aspects:

- **What happens in the controller method?**

  In a Spring Boot application using Spring MVC:

- A controller method annotated with @GetMapping, @PostMapping, etc., handles HTTP requests.
- It receives data (e.g., from query params, path variables, or request body) and processes it.
- It may call a **service layer** or interact with the **repository** layer (e.g., via JPA).
- It returns an object or a ResponseEntity.

- **How the bean is converted into JSON reponse?**

- Spring Boot uses Jackson (a JSON library) automatically if it's on the classpath (spring-boot-starter-web includes it).
- The @RestController or @ResponseBody annotation tells Spring to serialize the returned Java object to JSON.
- This process is called message conversion, handled by HttpMessageConverter.

- **In network tab of developer tools show the HTTP header details received**

```
Content-Type: application/json
Content-Length: 123
Date: Mon, 15 Jul 2025 10:00:00 GMT
Server: Apache-Coyote/1.1
```

- **In postman click on "Headers" tab to view the HTTP header details received**

```
Content-Type: application/json
Date: Mon, 15 Jul 2025 10:00:00 GMT
Transfer-Encoding: chunked
```

SUPERSET ID: 6407636

KANMANI MURUGHAIYAN