**Spring REST using Spring Boot 3**

**EXERCISE: REST - Get country based on country code**

**Objective:**

The objective of this hands-on is to develop a RESTful Web Service using Spring Boot that retrieves country information based on a country code passed as a path variable. This demonstrates how to use `@RestController`, `@GetMapping`, and `@PathVariable` annotations in a Spring Boot 3.x application.

**Tools & Technologies Used:**

\* Java 17

\* Spring Boot 3.x

\* Spring Web (REST)

\* Maven

\* Eclipse IDE

\* Spring Initializr ([<https://start.spring.io](https://start.spring.io))>

Project Metadata:

Project - Maven Project

Language - Java

Spring Boot - 3.5.x

Group - `com.example`

Artifact - `countrycodeapi`

Name - `countrycodeapi`

Description - Hands-on Spring Boot REST API that returns country details based on country code using `@PathVariable`

Packaging - Jar

Java Version - 17

Dependency - Spring Web

**Package Structure:**

com.example.countrycodeapi

├── CountrycodeapiApplication.java

├── controller

│ └── CountryCodeController.java

└── model

└── Country.java

**Implementation Overview:**

1. Main Application Class

The `CountrycodeapiApplication.java` class contains the Spring Boot entry point using `@SpringBootApplication`.

2. Model Class – Country.java

A simple POJO representing country data with fields such as `code`, `name`, and `capital`.

```java

public class Country {

private String code;

private String name;

private String capital;

// Getters and Setters

}

3. Controller – CountryCodeController.java

This REST controller uses the `@GetMapping("/country/{code}")` endpoint. It handles HTTP GET requests and returns a hardcoded `Country` object based on the provided country code.

```java

@RestController

public class CountryCodeController {

@GetMapping("/country/{code}")

public Country getCountryByCode(@PathVariable String code) {

Country country = new Country();

country.setCode(code);

switch (code.toUpperCase()) {

case "IN":

country.setName("India");

country.setCapital("New Delhi");

break;

case "US":

country.setName("United States");

country.setCapital("Washington, D.C.");

break;

case "FR":

country.setName("France");

country.setCapital("Paris");

break;

default:

country.setName("Unknown");

country.setCapital("Unknown");

}

return country;

}

}

**API Endpoint:**

\* URL Pattern: `GET /country/{code}`

\* Example Request:

`http://localhost:8081/country/IN`

\* Example Response:

{

"code": "IN",

"name": "India",

"capital": "New Delhi"

}

**Output Verification:**

Tested using browser and Postman with different country codes like `/country/IN`, `/country/US`, and `/country/FR`. The application returned accurate and expected responses.

**OUTPUT:**

