Spring REST using Spring Boot 3 Mandatory Hands-on Exercise:

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

### Code:

// MainApp.java

@SpringBootApplication

public class MainApp {

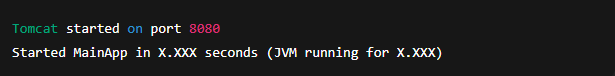
public static void main(String[] args) {

SpringApplication.run(MainApp.class, args);

}

}

### Expected Output:



## Exercise 2: Spring Core-Load Country from Spring Configuration XML

### Country.xml:

<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="country" class="com.example.Country">

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

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

</bean>

</beans>

### Code:

public class Main {

public static void main(String[] args) {

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

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

System.out.println(country.getCode() + " - " + country.getName());

}

}

### Expected Output:



## Exercise 3: Hello World RESTful Web Service

### Code:

// HelloController.java

@RestController

public class HelloController {

@GetMapping("/hello")

public String sayHello() {

return "Hello World";

}

}

### Expected Output:



## Exercise 4: REST- Country Web Service

### Country.java:

public class Country {

private String code;

private String name;

// Constructor, Getters, Setters

public Country(String code, String name) {

this.code = code;

this.name = name;

}

}

### CountryController.java:

@RestController

public class CountryController {

@GetMapping("/country")

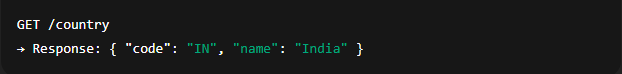
public Country getCountry() {

return new Country("IN", "India");

}

}

### Expected Output:



## Exercise 5: REST- Get country based on country code

## Code:

@RestController

public class CountryController {

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

public Country getCountry(@PathVariable String code) {

if (code.equalsIgnoreCase("IN"))

return new Country("IN", "India");

else if (code.equalsIgnoreCase("US"))

return new Country("US", "United States");

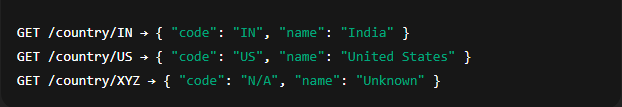
else

return new Country("N/A", "Unknown");

}

}

### Expected Output:



## Exercise 6: Create authentication service that returns JWT

### Dependencies (in pom.xml):

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-api</artifactId>

<version>0.11.5</version>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-impl</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-jackson</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

### Code:

@RestController

public class AuthController {

private String secretKey = "mysecret";

@PostMapping("/authenticate")

public String authenticate(@RequestParam String username, @RequestParam String password) {

if ("admin".equals(username) && "123".equals(password)) {

return Jwts.builder()

.setSubject(username)

.signWith(Keys.hmacShaKeyFor(secretKey.getBytes()), SignatureAlgorithm.HS256)

.compact();

}

return "Invalid credentials";

}

}

### Expected Output:

