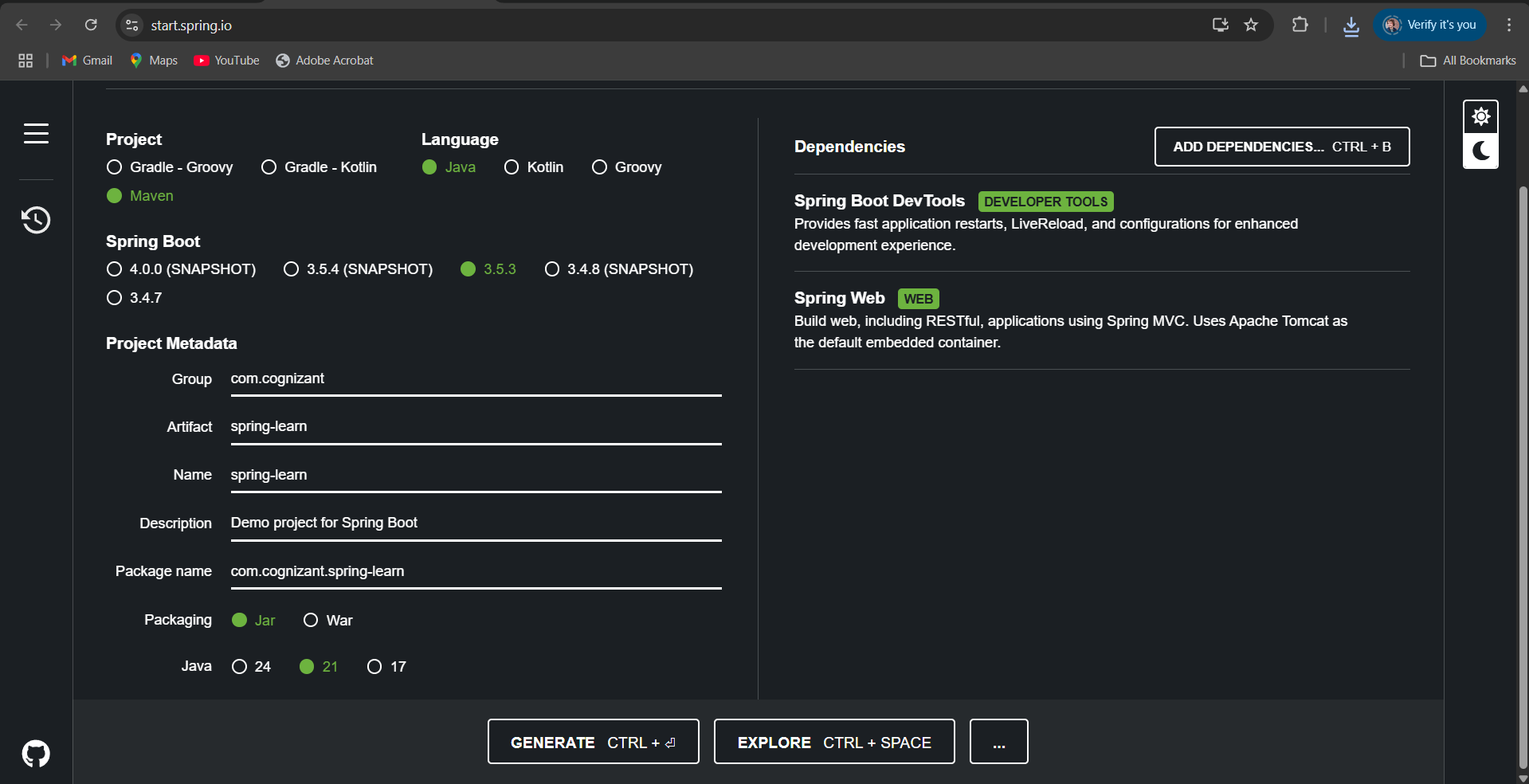
1.Spring-rest-handson

# 1. Create a Spring Web Project using Maven :



**Student.java :**

package com.cognizant.spring\_learn;

public class Student {

private int id;

private String name;

private String department;

public Student() {}

public Student(int id, String name, String department) {

this.id = id;

this.name = name;

this.department = department;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}

}

**StudentController.java :**

package com.cognizant.spring\_learn;

import org.springframework.web.bind.annotation.\*;

import java.util.\*;

@RestController

@RequestMapping("/students")

public class StudentController {

private List<Student> students = new ArrayList<>();

public StudentController() {

students.add(new Student(1, "Harish", "CSE"));

students.add(new Student(2, "Aishwarya", "ECE"));

students.add(new Student(3, "Karthik", "MECH"));

}

@GetMapping

public List<Student> getAllStudents() {

return students;

}

@GetMapping("/{id}")

public Student getStudentById(@PathVariable int id) {

return students.stream()

.filter(s -> s.getId() == id)

.findFirst()

.orElseThrow(() -> new RuntimeException("Student not found"));

}

@PostMapping

public String addStudent(@RequestBody Student student) {

students.add(student);

return "Student added successfully!";

}

@PutMapping("/{id}")

public String updateStudent(@PathVariable int id, @RequestBody Student updated) {

for (Student s : students) {

if (s.getId() == id) {

s.setName(updated.getName());

s.setDepartment(updated.getDepartment());

return "Student updated!";

}

}

return "Student not found";

}

@DeleteMapping("/{id}")

public String deleteStudent(@PathVariable int id) {

boolean removed = students.removeIf(s -> s.getId() == id);

return removed ? "Student deleted" : "Student not found";

}

}

**SpringLearnApplication.java :**

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) {

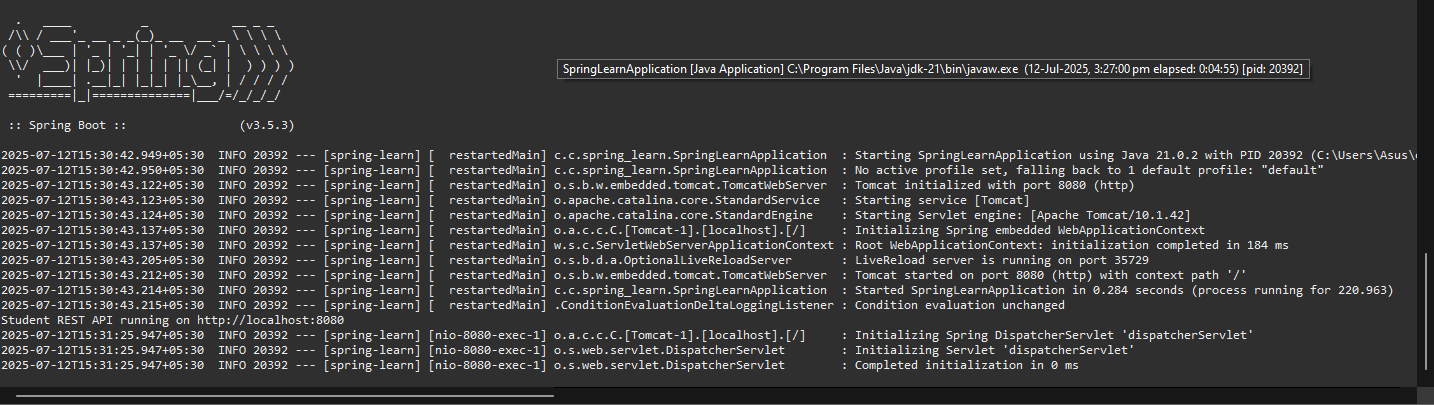
SpringApplication.run(SpringLearnApplication.class, args);

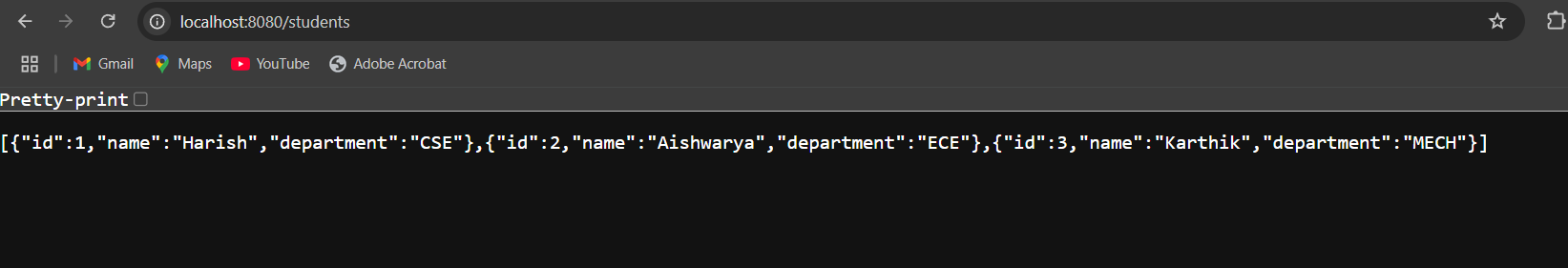
System.out.println("Student REST API running on http://localhost:8080");

}

}

**OUTPUT :**





# 4. Spring Core – Load Country from Spring Configuration XML

**Country.xml :**

<?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.spring\_learn.Country">

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

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

</bean>

</beans>

**Country.java :**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

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

private String code;

private String name;

public Country() {

LOGGER.debug("Inside Country Constructor");

}

public String getCode() {

LOGGER.debug("Inside getCode");

return code;

}

public void setCode(String code) {

LOGGER.debug("Inside setCode");

this.code = code;

}

public String getName() {

LOGGER.debug("Inside getName");

return name;

}

public void setName(String name) {

LOGGER.debug("Inside setName");

this.name = name;

}

@Override

public String toString() {

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

}

}

**SpringLearnApplication :**

package com.cognizant.spring\_learn;

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) {

SpringApplication.run(SpringLearnApplication.class, args);

LOGGER.info("START");

displayCountry();

LOGGER.info("END");

}

public static void displayCountry() {

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

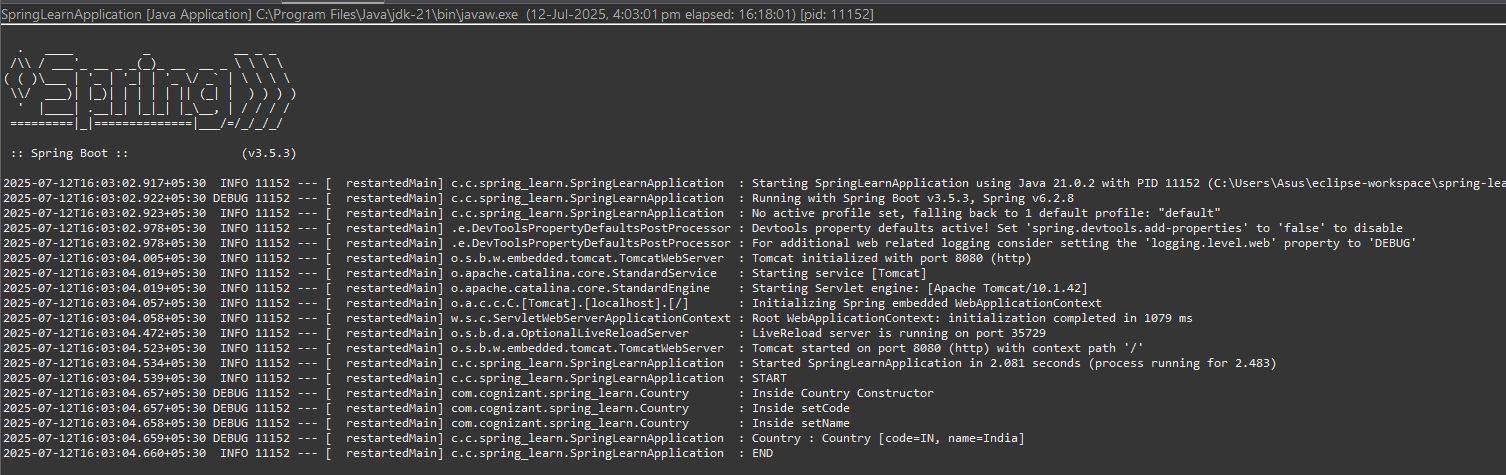
Country country = context.getBean("country", Country.class);

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

}

}

**OUTPUT :**



2. Spring-rest-handson

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**

**HelloController.java :**

package com.cognizant.spring\_learn.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");

LOGGER.info("END");

return "Hello World!!";

}

}

**SpringLearnApplication :**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@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);

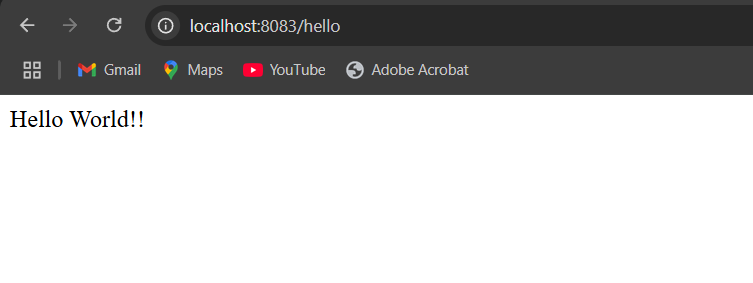
LOGGER.info("END");

}

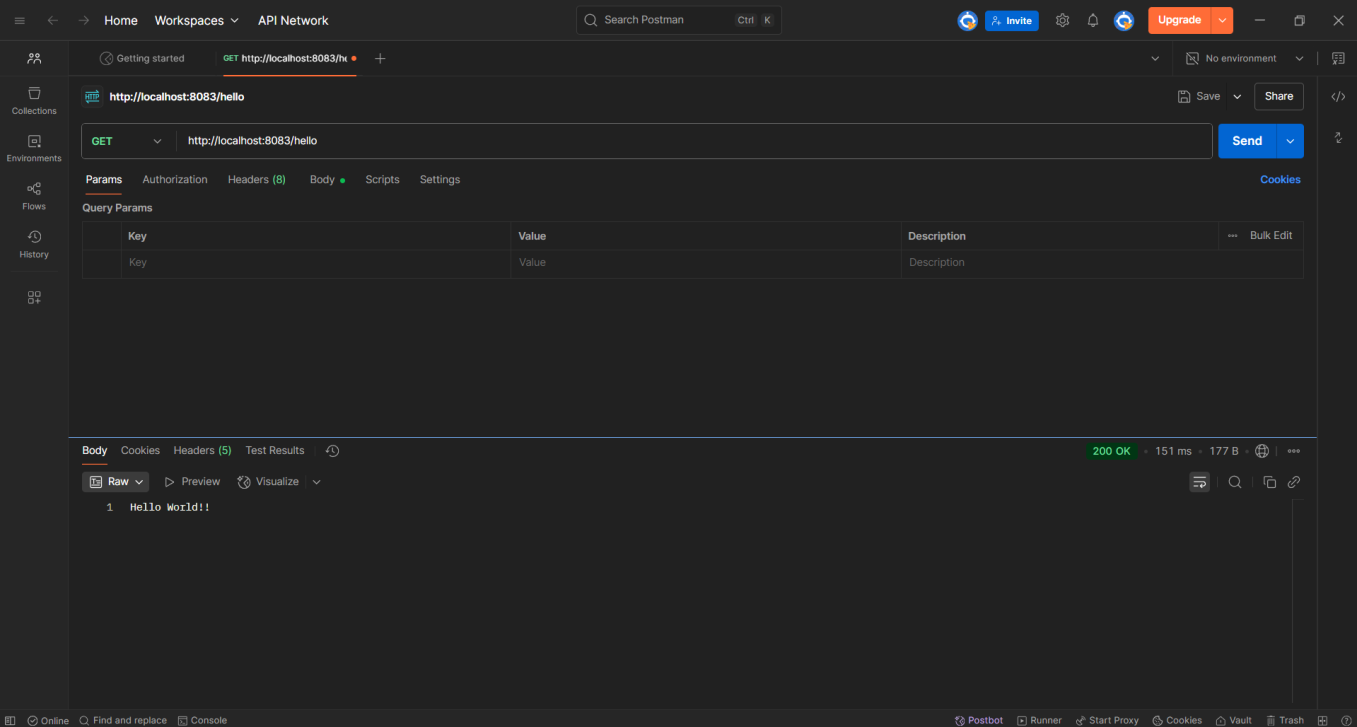
}

**OUTPUT :**

**IN CHROME BROWSER :**



**IN POST MAN :**



REST - Country Web Service   
  
**Write a REST service that returns India country details in the earlier created spring learn application.**

**Country.java :**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

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

private String code;

private String name;

public Country() {

LOGGER.debug("Inside Country Constructor");

}

public String getCode() {

LOGGER.debug("Inside getCode()");

return code;

}

public void setCode(String code) {

LOGGER.debug("Inside setCode()");

this.code = code;

}

public String getName() {

LOGGER.debug("Inside getName()");

return name;

}

public void setName(String name) {

LOGGER.debug("Inside setName()");

this.name = name;

}

@Override

public String toString() {

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

}

}

**CountryController.java :**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.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.debug("Country: {}", country.toString());

LOGGER.info("END");

return country;

}

}

**SpringLearnApplication :**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@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);

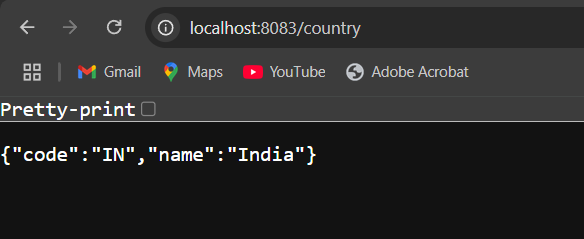
LOGGER.info("END");

}

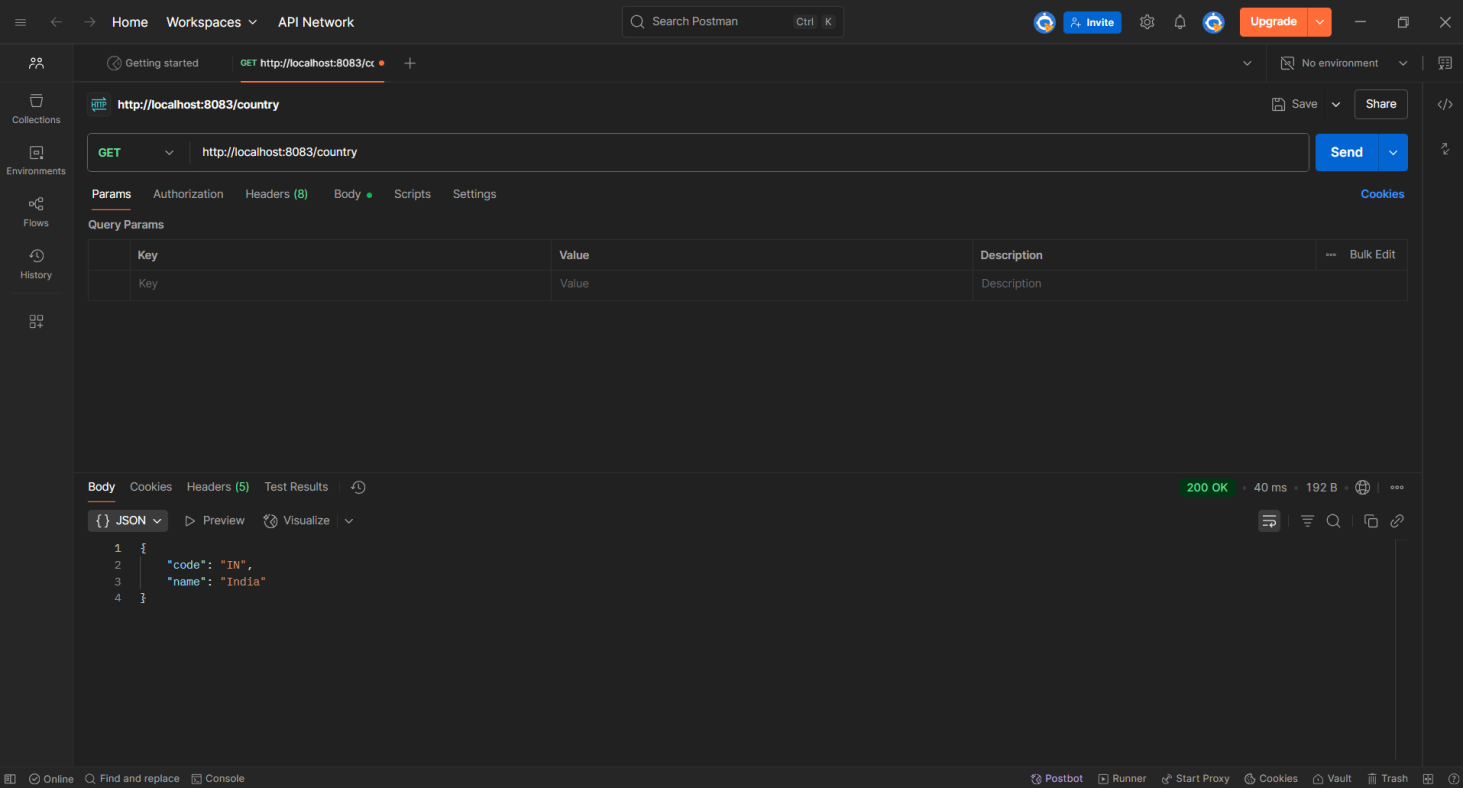
}

**OUTPUT :**

**IN CHROME BROWSER :**



**IN POST MAN :**



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.**

**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

https://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="countryIN" class="com.cognizant.spring\_learn.model.Country">

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

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

</bean>

<bean id="countryUS" class="com.cognizant.spring\_learn.model.Country">

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

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

</bean>

<bean id="countryDE" class="com.cognizant.spring\_learn.model.Country">

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

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

</bean>

<bean id="countryJP" class="com.cognizant.spring\_learn.model.Country">

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

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

</bean>

<bean id="countryList" class="java.util.ArrayList">

<constructor-arg>

<list>

<ref bean="countryIN"/>

<ref bean="countryUS"/>

<ref bean="countryGE"/>

<ref bean="countryJP"/>

</list>

</constructor-arg>

</bean>

</beans>

**Country.java :**

package com.cognizant.spring\_learn.model;

public class Country {

private String code;

private String name;

public Country() {

System.out.println("Inside Country Constructor");

}

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 + "]";

}

}

**CountryService.java :**

package com.cognizant.spring\_learn.service;

import com.cognizant.spring\_learn.model.Country;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

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

public Country getCountry(String code) {

LOGGER.debug("Start getCountry()");

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

List<Country> countries = context.getBean("countryList", List.class);

return countries.stream()

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

.findFirst()

.orElse(null);

}

}

**CountryController.java :**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.model.Country;

import com.cognizant.spring\_learn.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("/country/{code}")

public Country getCountry(@PathVariable String code) {

LOGGER.info("START getCountry()");

Country country = countryService.getCountry(code);

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

LOGGER.info("END getCountry()");

return country;

}

}

**SpringLearnApplication.java :**

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

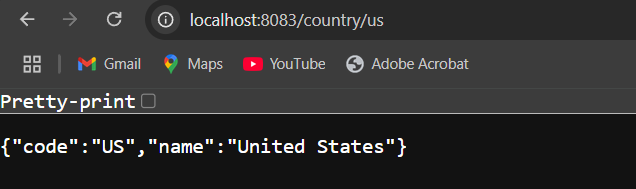
public static void main(String[] args) {

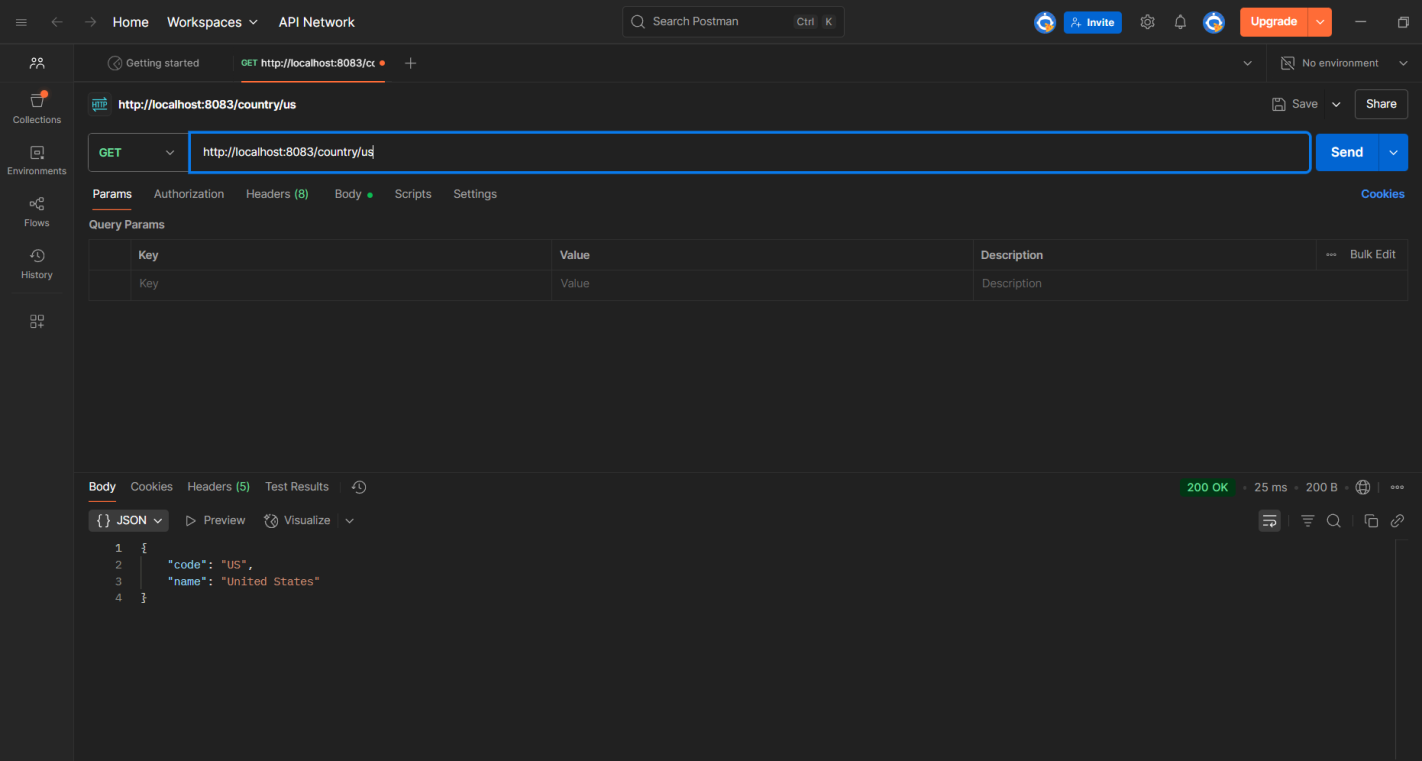
SpringApplication.run(SpringLearnApplication.class, args);

}

}

**OUTPUT :**

**IN CHROME BROWSER :**  


**IN POST MAN :**  


5. JWT-handson

# Create authentication service that returns JWT

**Pom.xml :**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

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

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

<version>3.5.3</version>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<properties>

<java.version>21</java.version>

</properties>

<dependencies>

<!-- Spring Boot Web and Security -->

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<!-- JWT dependencies -->

<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>

</dependencies>

<build>

<plugins>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**Application.properties :**

server.port=8090

**JwtService.java :**

package com.cognizant.spring\_learn.service;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.security.core.Authentication;

import org.springframework.stereotype.Service;

import java.util.Date;

@Service

public class JwtService {

private final String SECRET\_KEY = "secret-key";

public String generateToken(Authentication authentication) {

return Jwts.builder()

.setSubject(authentication.getName())

.setIssuedAt(new Date(System.currentTimeMillis()))

.setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 60)) // 1 hour

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

}

**AuthenticationController.java :**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.service.JwtService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.core.Authentication;

import org.springframework.web.bind.annotation.\*;

@RestController

public class AuthenticationController {

@Autowired

private JwtService jwtService;

@PostMapping("/authenticate")

public String authenticate(Authentication authentication) {

return jwtService.generateToken(authentication);

}

@GetMapping("/hello")

public String hello() {

return "Hello, you are authenticated!";

}

}

**SecurityConfig.java :**

package com.cognizant.spring\_learn.security;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.Customizer;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

import org.springframework.security.web.SecurityFilterChain;

@Configuration

public class SecurityConfig {

@Bean

public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {

http

.csrf(csrf -> csrf.disable())

.authorizeHttpRequests(auth -> auth

.requestMatchers("/authenticate").authenticated()

.anyRequest().permitAll()

)

.httpBasic(Customizer.withDefaults());

return http.build();

}

@Bean

public InMemoryUserDetailsManager userDetailsService() {

UserDetails user = User.builder()

.username("user")

.password("{noop}pwd")

.roles("USER")

.build();

return new InMemoryUserDetailsManager(user);

}

}

**SpringLearnApplication.java :**

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

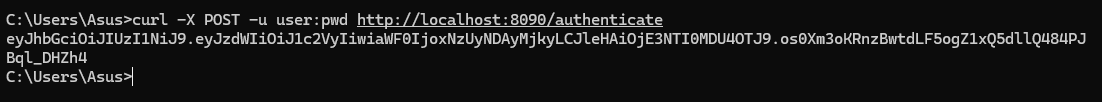
public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

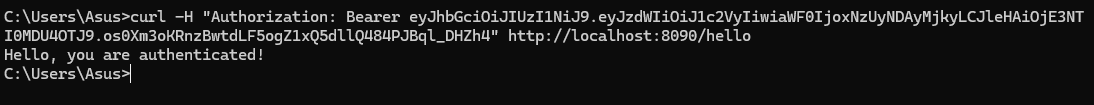
}

**REQUEST AND RESPONSE :**

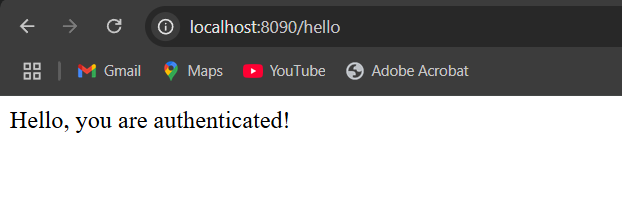


**OUTPUT :**

**IN TERMINAL :**



**IN CHROME BROWSER :**



**IN POST MAN :**

