**Week-4 Hands-On Solutions**

**Spring REST using Spring Boot 3**

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

**Project Structure Overview:**

spring-learn/

│

├── src/

│ ├── main/

│ │ ├── java/

│ │ │ └── com/cognizant/spring\_learn/

│ │ │ ├── SpringLearnApplication.java

│ │ │ └── HelloController.java

│ │ └── resources/

│ │ └── application.properties

│ └── test/

│ └── java/

│ └── com/cognizant/spring\_learn/

│ └── SpringLearnApplicationTests.java

├── pom.xml

**Code Files:**

**1.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) {

System.out.println("SpringLearnApplication started working ...");

SpringApplication.run(SpringLearnApplication.class, args);

}

}

**2.HelloController.java**

package com.cognizant.spring\_learn;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HelloController {

@GetMapping("/hello")

public String sayHello() {

return "Hello from Spring Boot! and executed successfully";

}

}

**3.SpringLearnApplicationTests.java**

package com.cognizant.spring\_learn;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

class SpringLearnApplicationTests {

@Test

void contextLoads() {

}

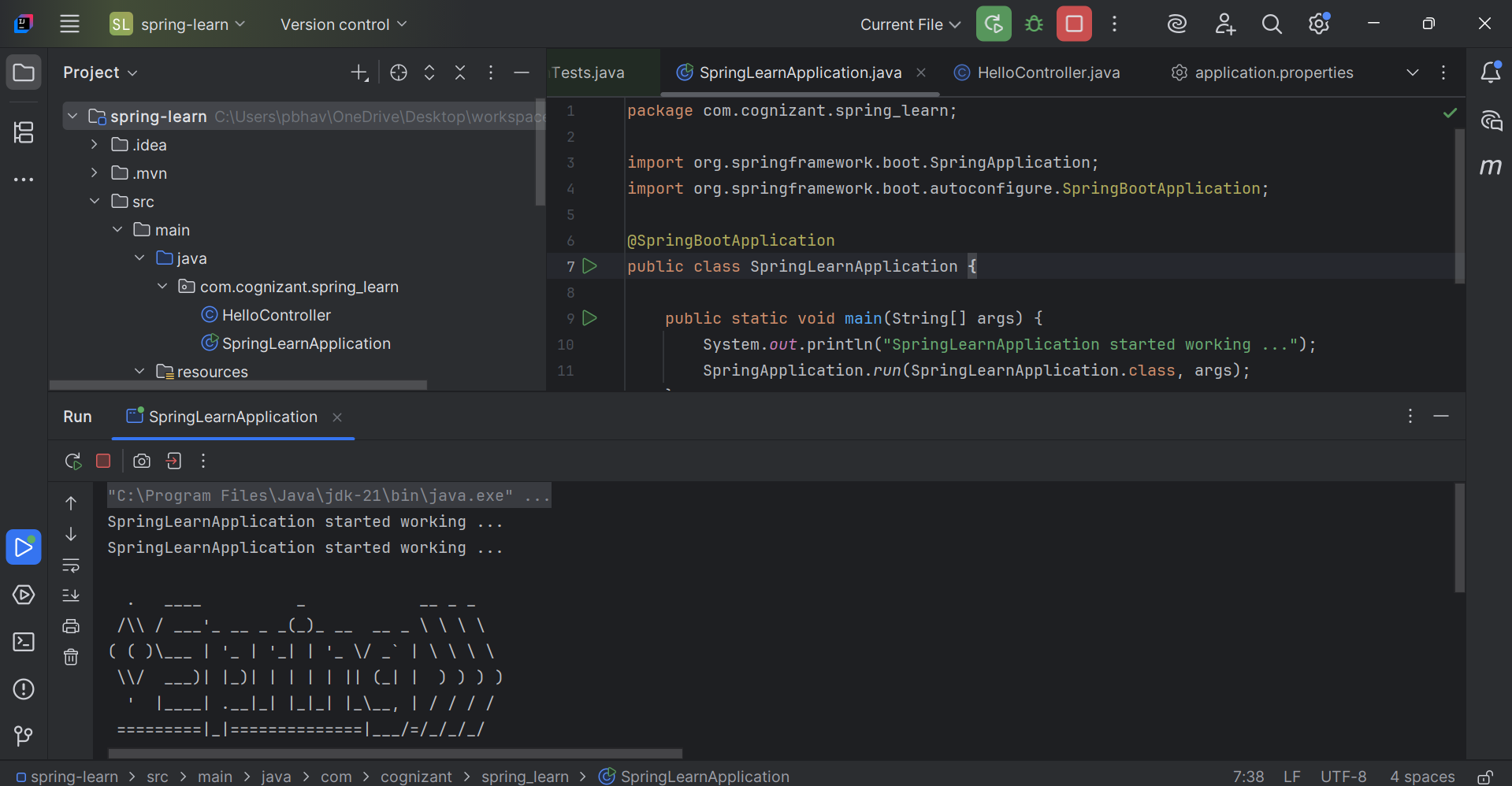
}

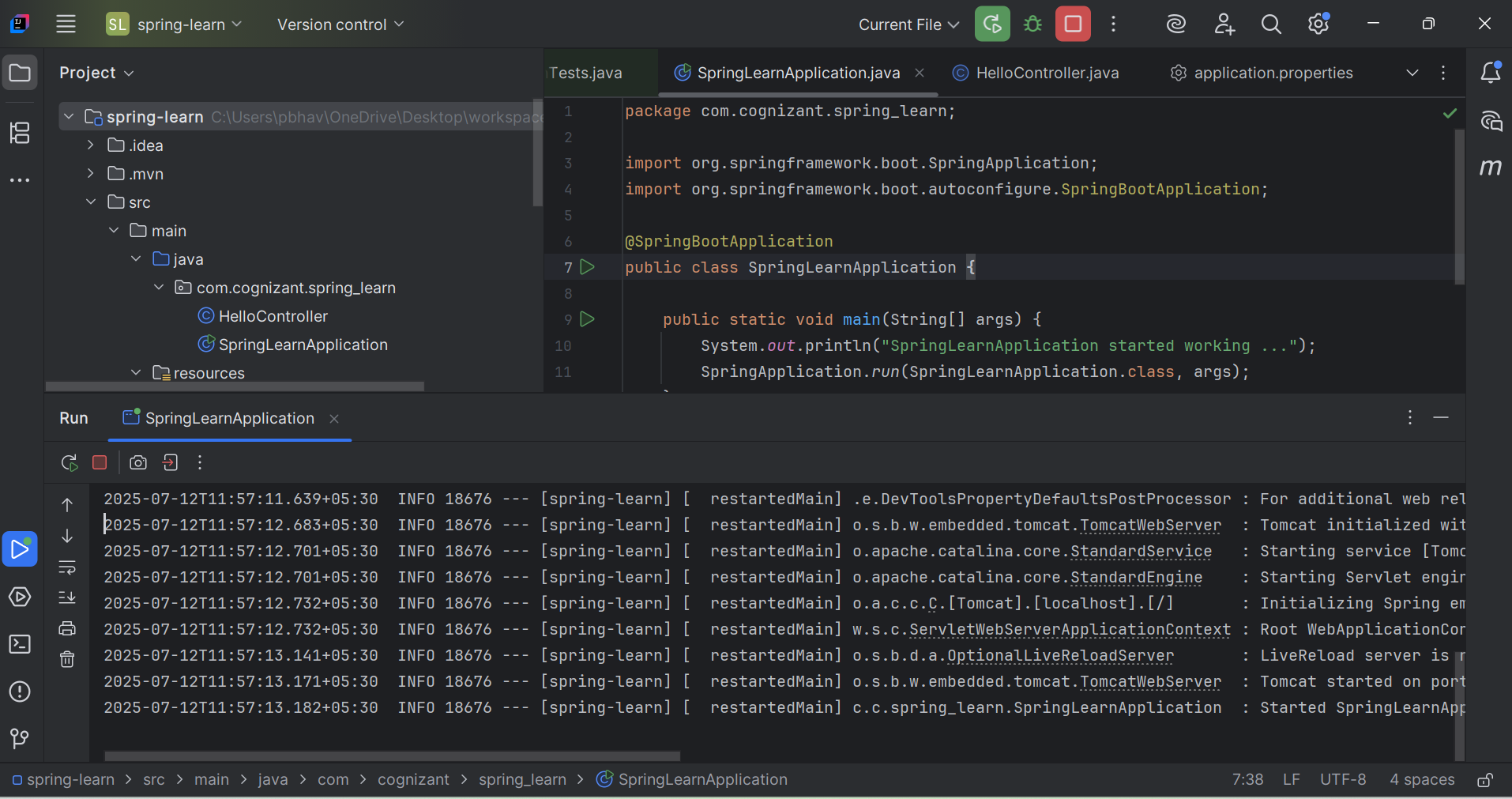
**Configuration File:**

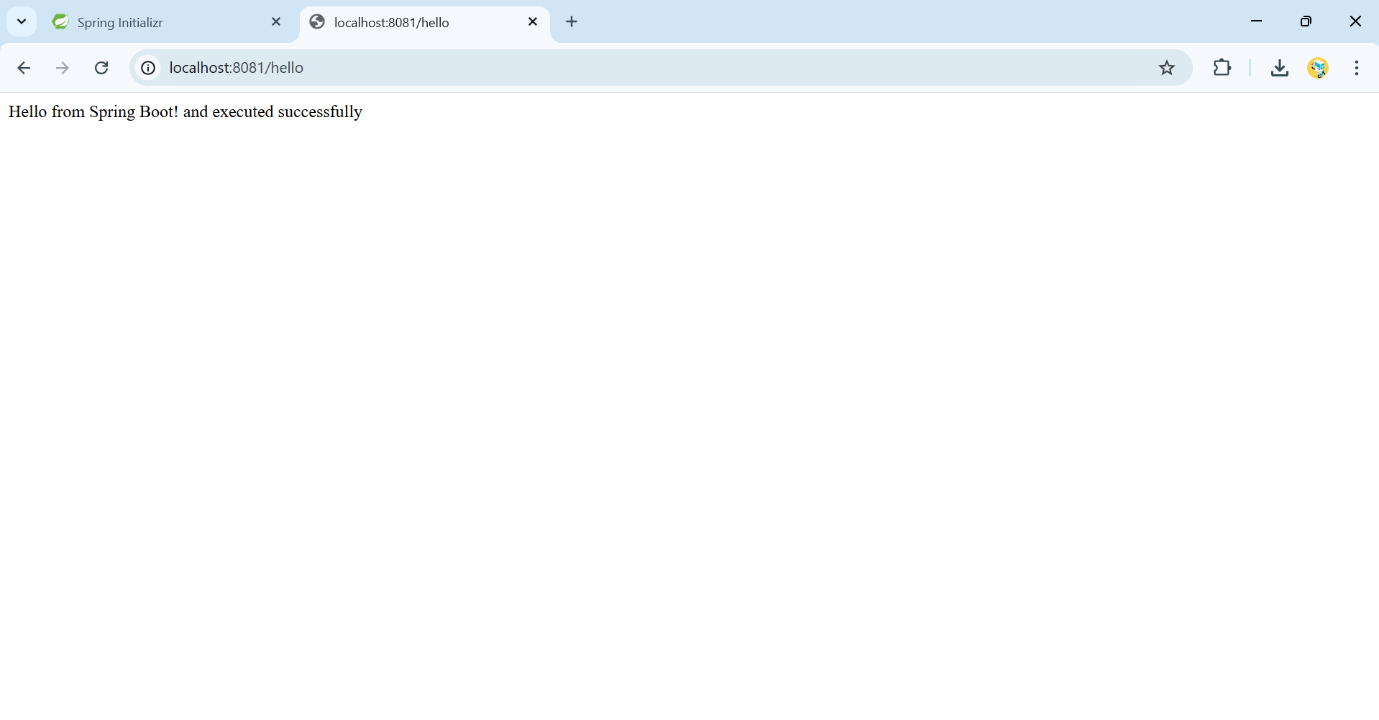
**4.application.properties**

spring.application.name=spring-learn  
server.port=8081

**Output:**

****

****

****

**Exercise2: Spring Core – Load Country from Spring Configuration XML**

**Project Structure:**

spring-learn/

├── src/

│ └── main/

│ ├── java/

│ │ └── com.cognizant.spring\_learn/

│ │ ├── Country.java

│ │ └── SpringLearnApplication.java

│ └── resources/

│ ├── application.properties

│ └── country.xml

├── pom.xml

**Java Classes:**

**1.Country.java**

package com.cognizant.spring\_learn;

public class Country {

private String code;

private String name;

public Country() {

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

}

public void setCode(String code) {

System.out.println("Setter called for code");

this.code = code;

}

public void setName(String name) {

System.out.println("Setter called for name");

this.name = name;

}

@Override

public String toString() {

return "Country{code='" + code + "', name='" + name + "'}";

}

}

**2.SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SpringLearnApplication {

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

public static void main(String[] args) {

displayCountry();

}

public static void displayCountry() {

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

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

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

}

}

**3.XML Configuration (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

http://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>

**4.POM Dependencies (pom.xml):**

<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

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<!-- Spring Core Context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

<!-- SLF4J Logging -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.36</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-simple</artifactId>

<version>1.7.36</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.11.0</version>

<configuration>

<source>17</source>

<target>17</target>

</configuration>

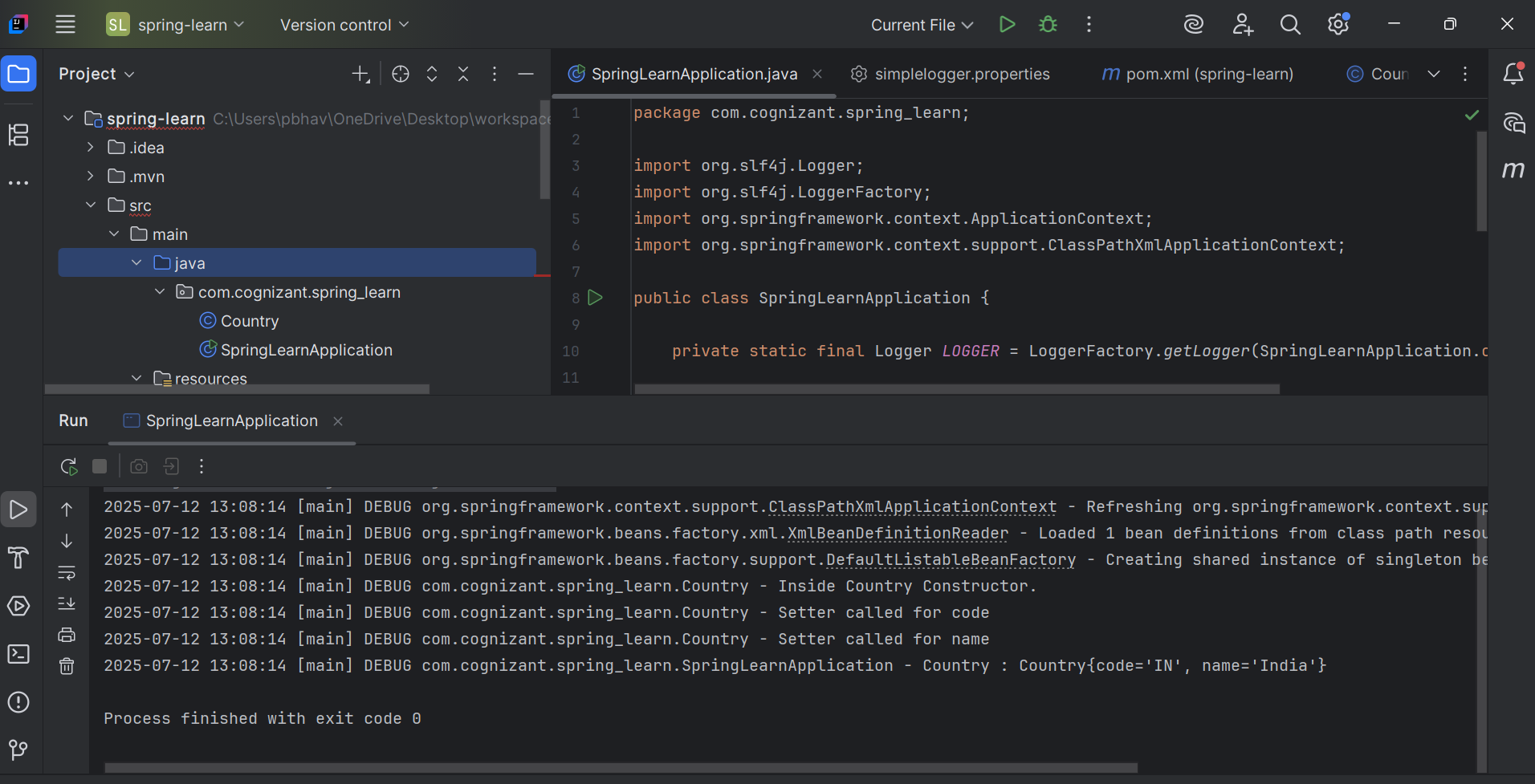
</plugin>

</plugins>

</build>

</project>

**5.Output:**

****

**Exercise3: Hello World RESTful Web Service**

**1.Project Structure (Spring Boot Maven)**

spring-learn/

├── src/

│ ├── main/

│ │ ├── java/

│ │ │ └── com/

│ │ │ └── cognizant/

│ │ │ └── springlearn/

│ │ │ ├── SpringLearnApplication.java

│ │ │ └── controller/

│ │ │ └── HelloController.java

│ │ └── resources/

│ │ └── application.properties

├── pom.xml

**1.Main Application Class**

**SpringLearnApplication.java**

package com.cognizant.springlearn;

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);

}

}

**2.HelloController Class**

**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 - sayHello()");

String response = "Hello World!!";

LOGGER.info("END - sayHello()");

return response;

}

}

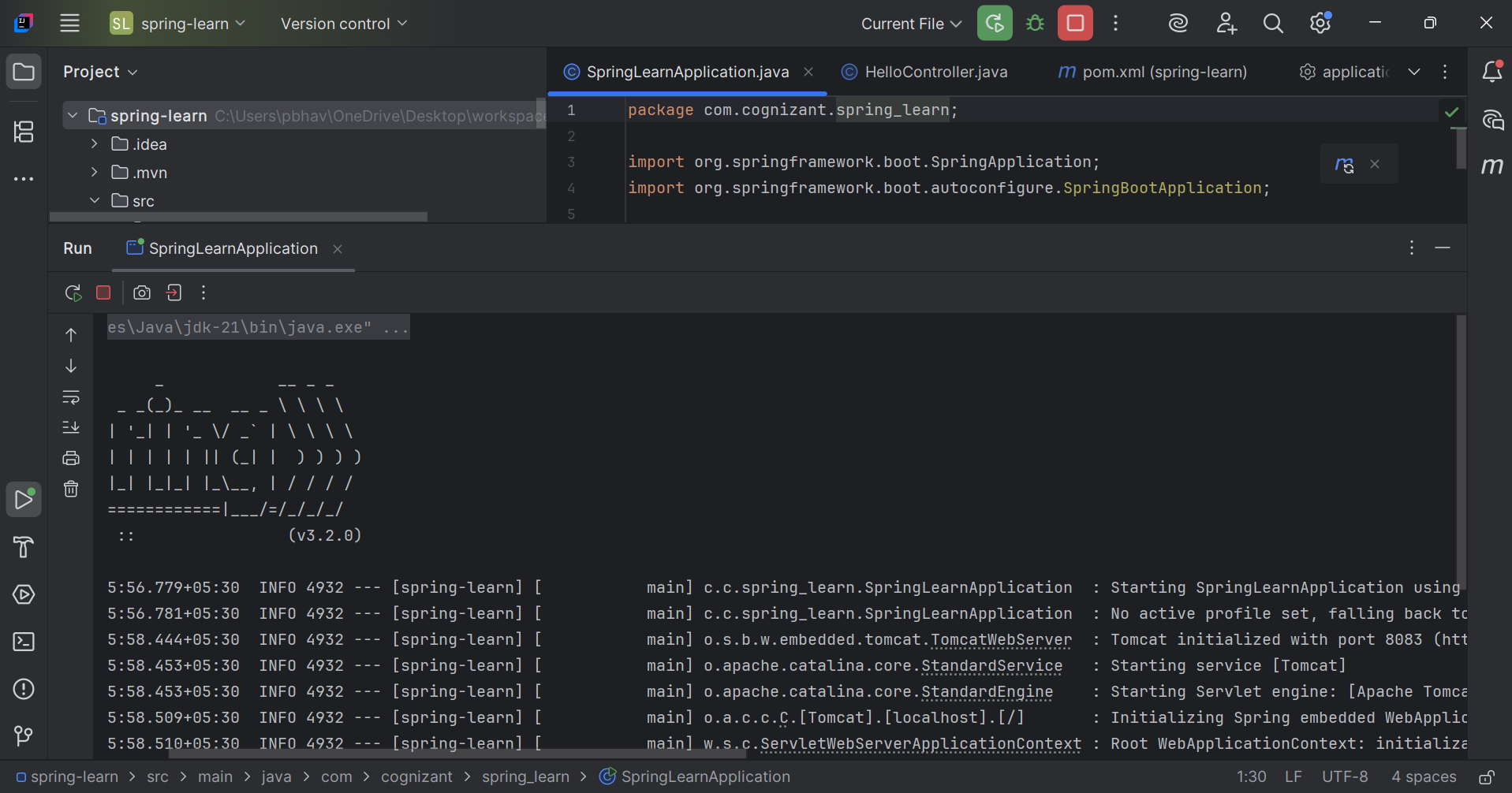
**3.application.properties**

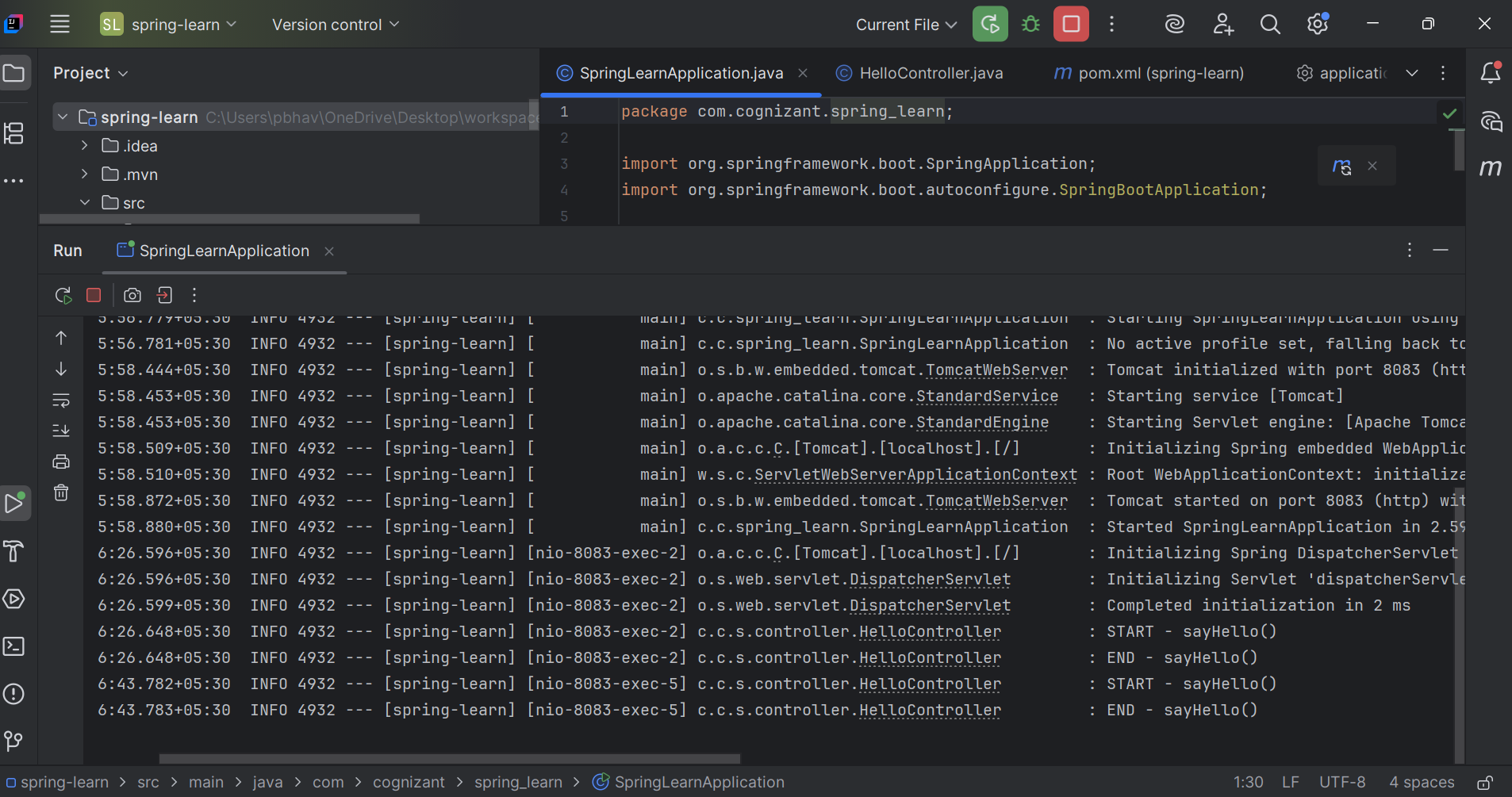
spring.application.name=spring-learn  
server.port=8083

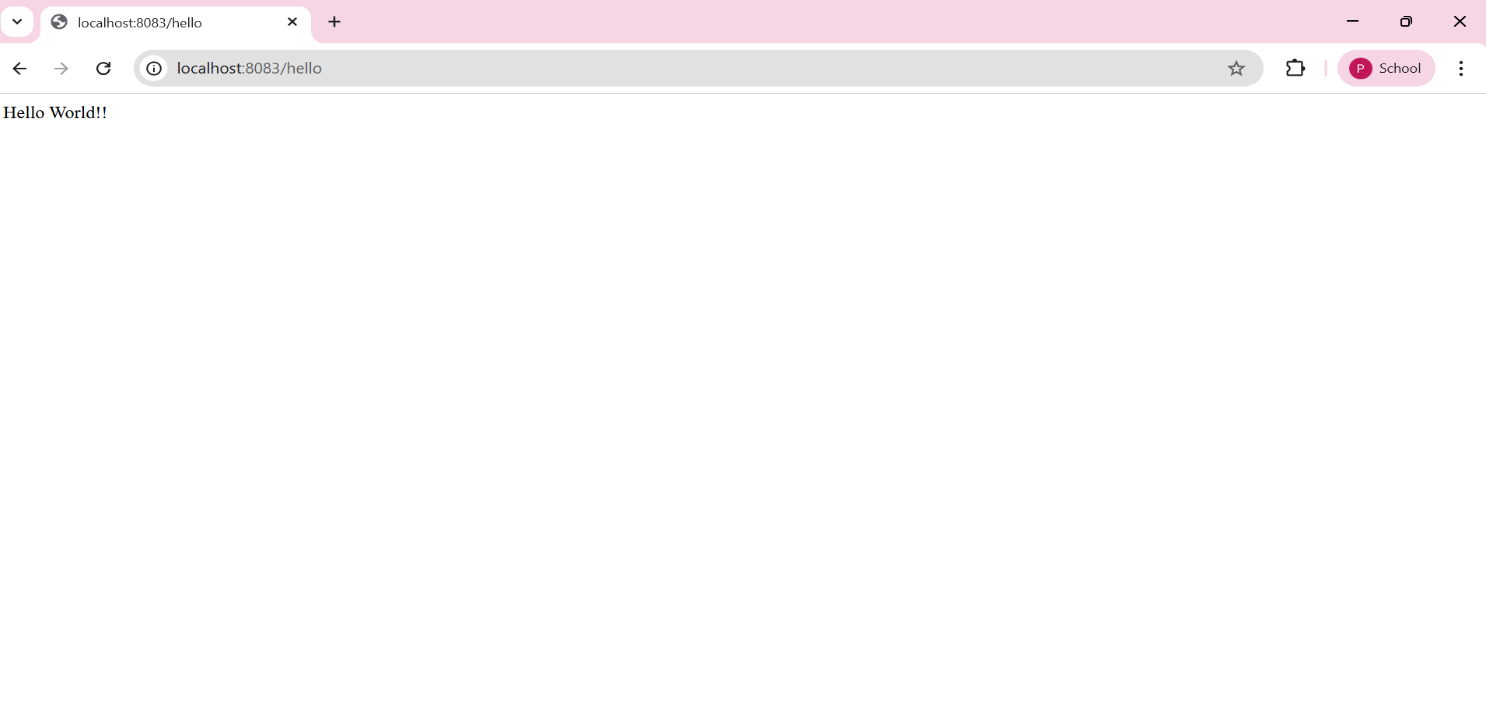
**4.pom.xml**

<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  
 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>com.cognizant</groupId>  
 <artifactId>spring-learn</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <packaging>jar</packaging>  
  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.2.0</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
  
 <dependencies>  
 <!-- Spring Boot Web -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 <version>2.7.18</version>  
 </dependency>  
  
 <!-- SLF4J Logging (optional) -->  
 <dependency>  
 <groupId>org.slf4j</groupId>  
 <artifactId>slf4j-api</artifactId>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

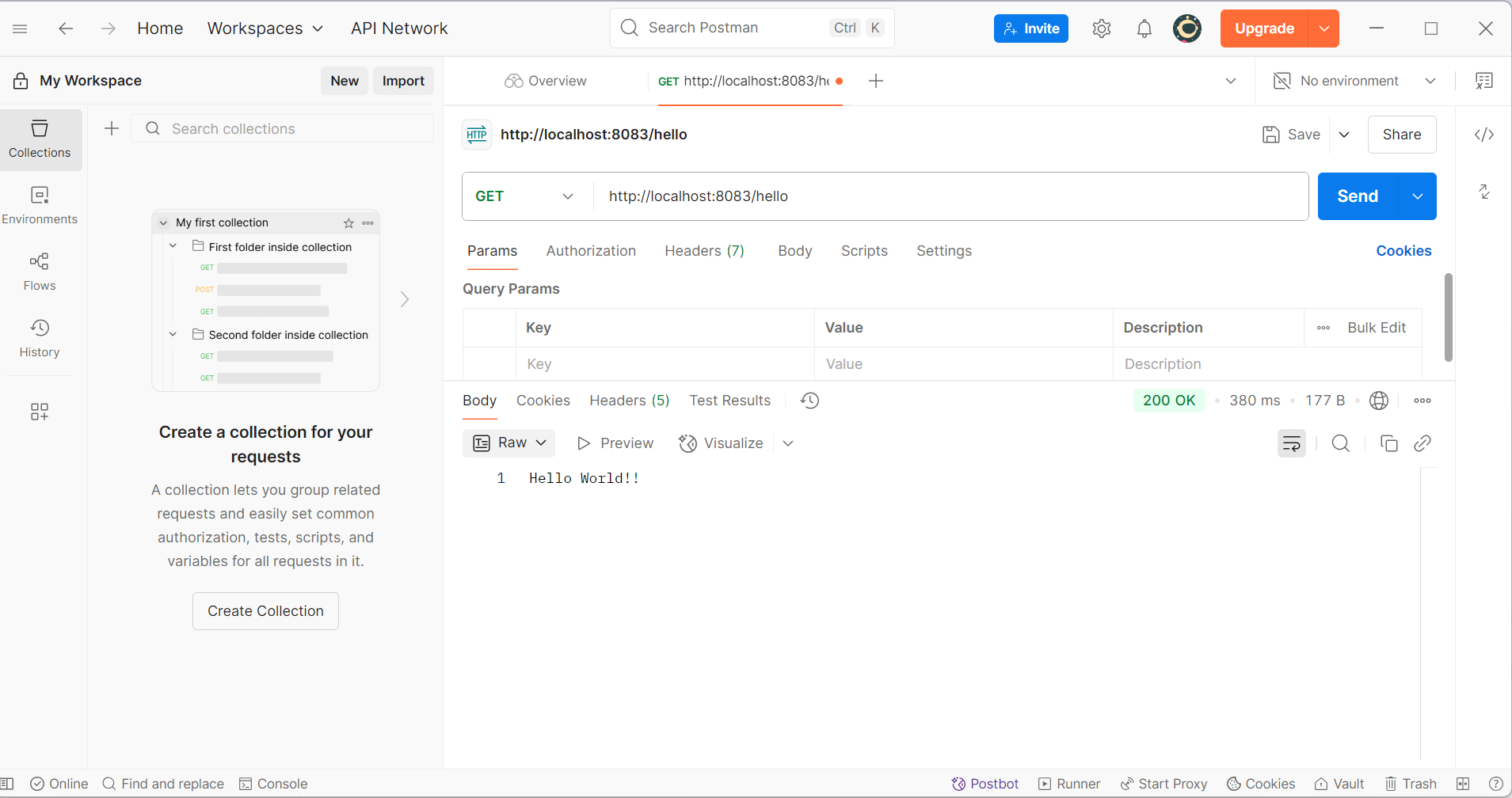
**5.Output:**







**Postman Output:**



**Exercise 4: REST - Country Web Service – "India" Response**

**Project Structure:**

spring-learn/

├── src/

│ ├── main/

│ │ ├── java/

│ │ │ └── com/

│ │ │ └── cognizant/

│ │ │ └── spring\_learn/

│ │ │ ├── controller/

│ │ │ │ └── CountryController.java

│ │ │ ├── Country.java

│ │ │ └── SpringLearnApplication.java

│ │ └── resources/

│ │ ├── application.properties

│ │ └── applicationContext.xml

**Source Codes:**

**1.Country.java**

package com.cognizant.spring\_learn;

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; }

}

**2.applicationContext.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="in" class="com.cognizant.spring\_learn.Country">

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

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

</bean>

</beans>

**3.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 - getCountryIndia()");

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

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

LOGGER.info("END - getCountryIndia()");

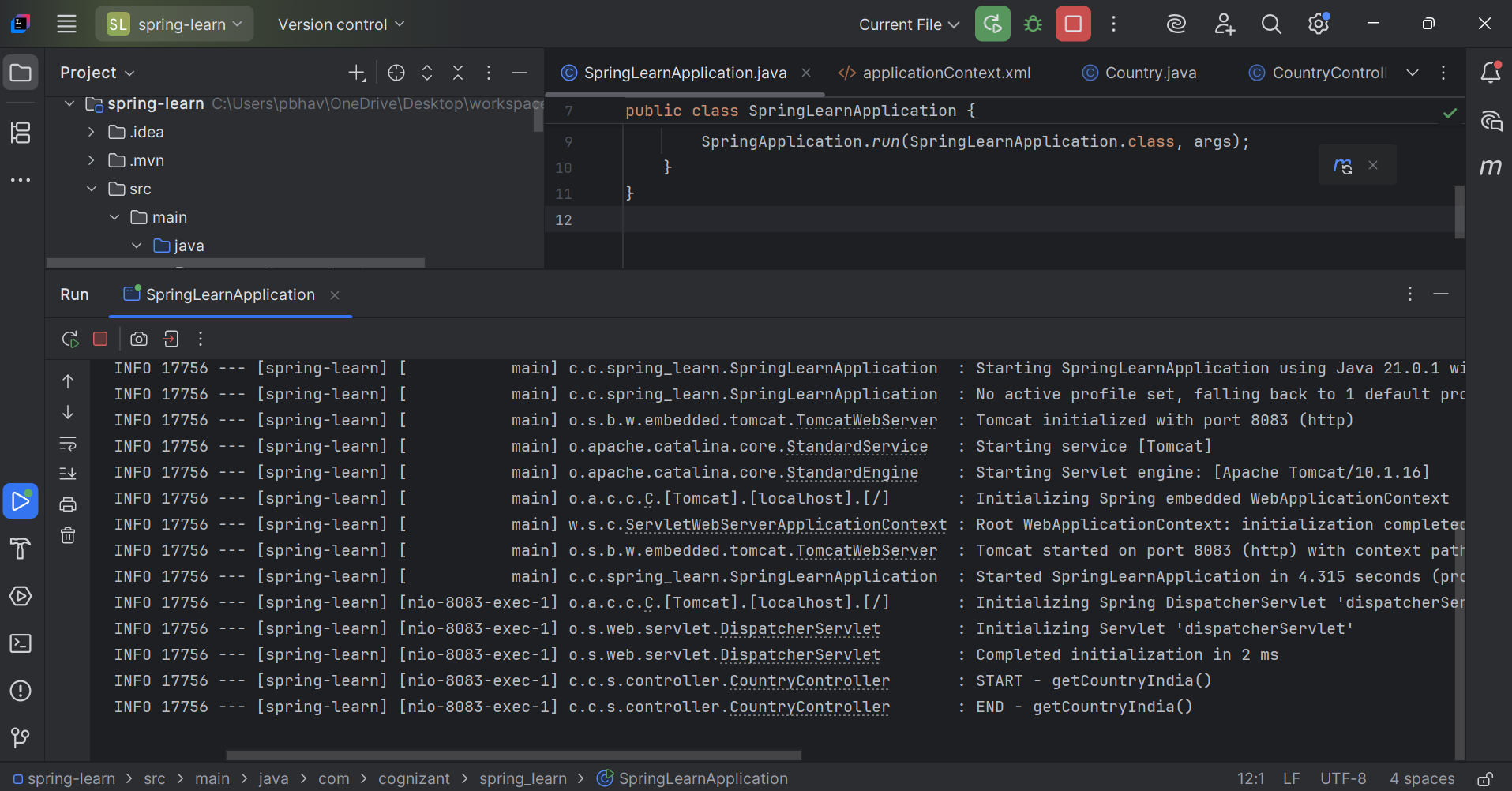
return country;

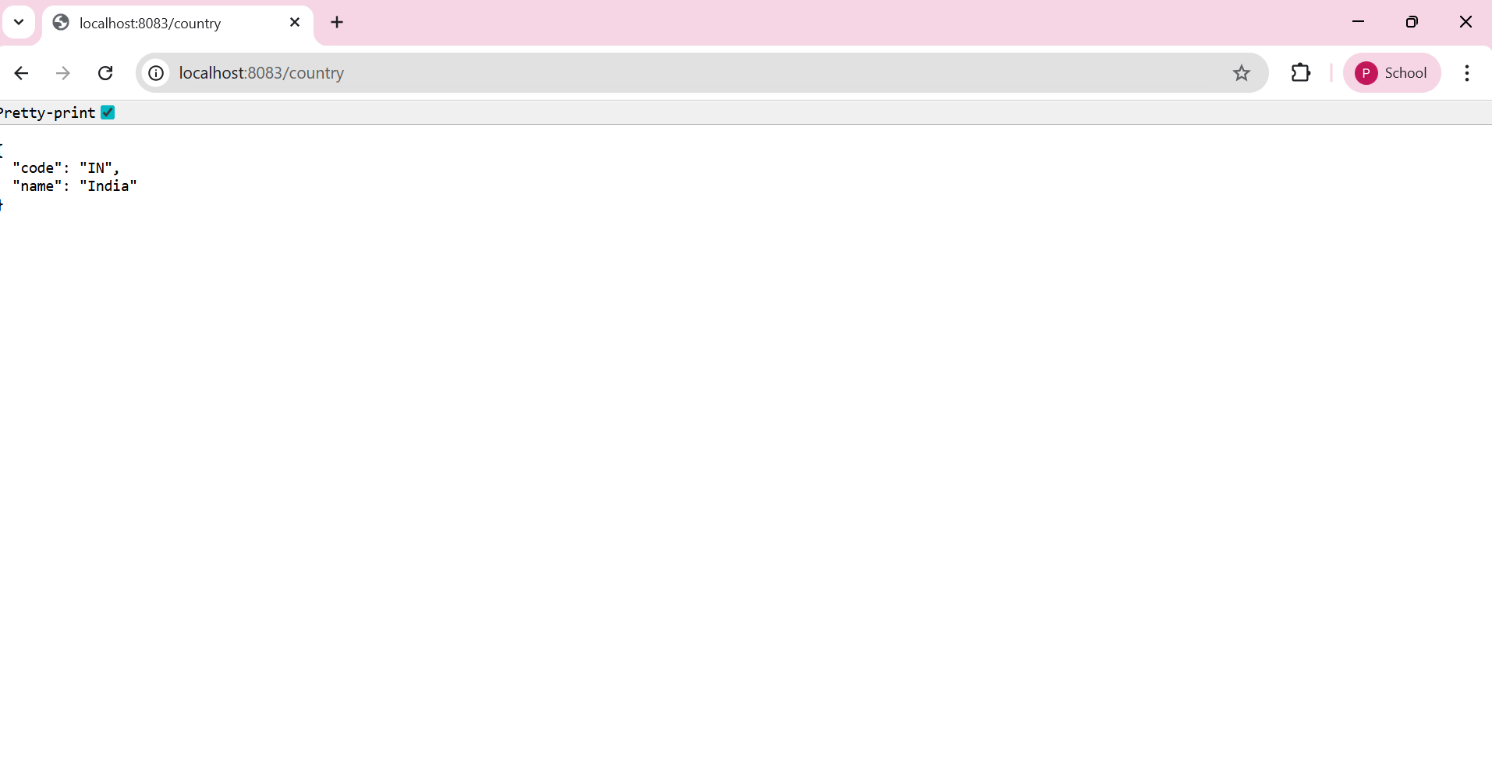
}

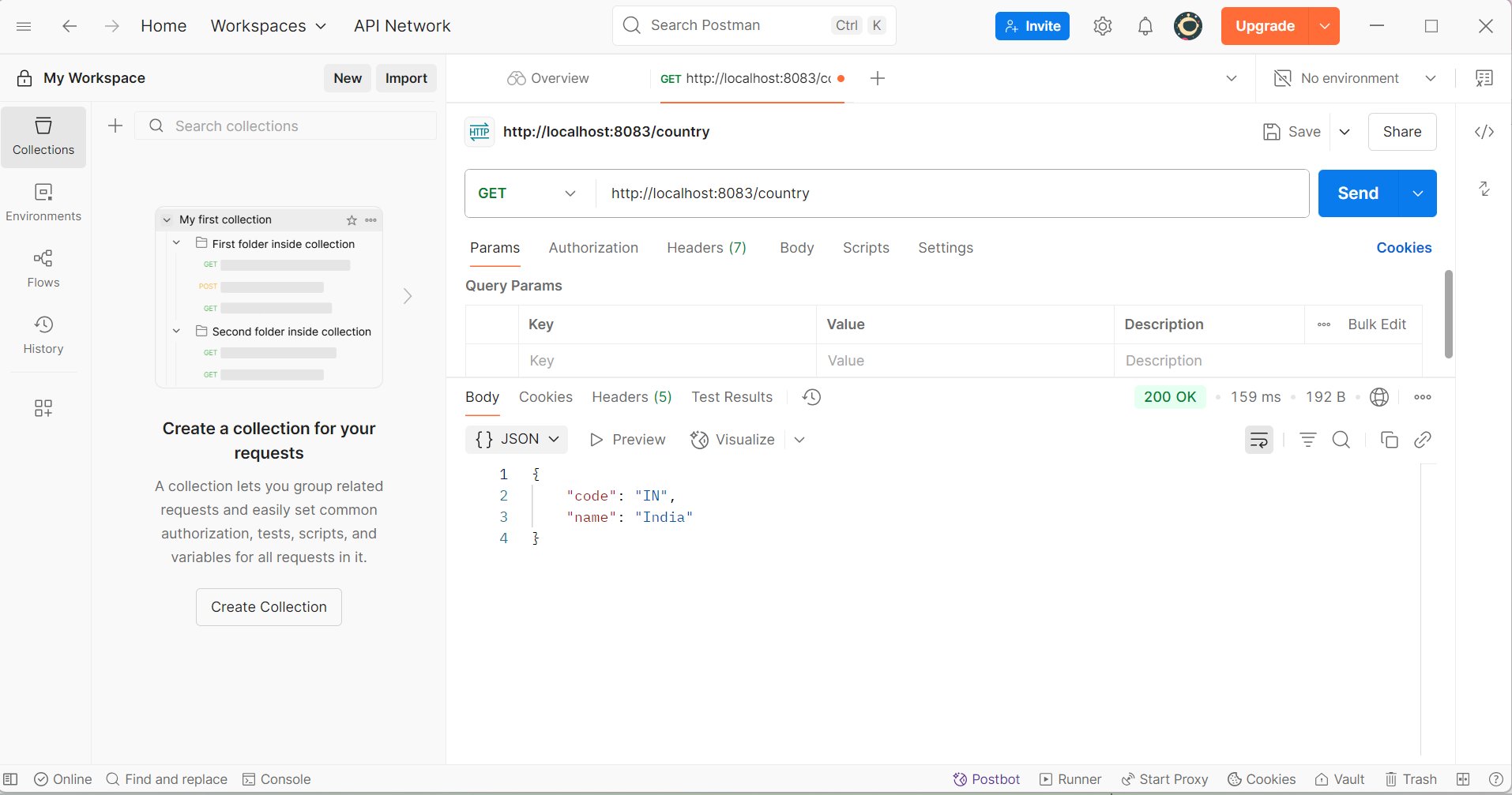
}

**->Run SpringLearnApplication.java**

**Output:**

****

****

****

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

**Project Structure**

spring-learn/

├── src/main/java/

│ └── com/cognizant/spring\_learn/

│ ├── Country.java

│ ├── SpringLearnApplication.java

│ ├── controller/

│ │ ├── CountryController.java

│ │ └── HelloController.java

│ └── service/

│ ├── CountryService.java

│ └── CountryServiceImpl.java

├── src/main/resources/

│ ├── applicationContext.xml

│ └── application.properties

**Configuration File - applicationContext.xml**

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

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

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

</bean>

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

<constructor-arg>

<list>

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

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

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

</bean>

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

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

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

</bean>

</list>

</constructor-arg>

</bean>

**CountryService.java**

package com.cognizant.spring\_learn.service;  
  
import com.cognizant.spring\_learn.Country;  
  
public interface CountryService {  
 Country getCountry(String code);  
}

**CountryServiceImpl.java**

package com.cognizant.spring\_learn.service;  
  
import com.cognizant.spring\_learn.Country;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
  
@Service  
public class CountryServiceImpl implements CountryService {  
  
 @Override  
 public Country getCountry(String code) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
 List<Country> countryList = (List<Country>) context.getBean("countryList");  
  
 return countryList.stream()  
 .filter(c -> c.getCode().equalsIgnoreCase(code))  
 .findFirst()  
 .orElse(null); // Or throw an exception if preferred  
 }  
}

**Controller - CountryController.java**

package com.cognizant.spring\_learn.controller;  
  
import com.cognizant.spring\_learn.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.http.HttpStatus;  
import org.springframework.web.bind.annotation.\*;  
import org.springframework.web.server.ResponseStatusException;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
@RestController  
public class CountryController {  
  
 private static final Logger LOGGER= LoggerFactory.getLogger(CountryController.class);  
  
 @Autowired  
 private CountryService countryService;  
  
 // GET /country -> returns India from applicationContext.xml  
 @GetMapping("/country")  
 public Country getCountryIndia() {  
 *LOGGER*.info("START - getCountryIndia()");  
 ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
 Country country = (Country) context.getBean("in");  
 LOGGER.info("END - getCountryIndia()");  
 return country;  
 }  
  
 // GET /countries/{code} -> returns country from countryList by code  
 @GetMapping("/countries/{code}")  
 public Country getCountry(@PathVariable String code) {  
 LOGGER.info("Request received for country code: {}", code);  
 try {  
 Country country = countryService.getCountry(code);  
 if (country == null) {  
 LOGGER.warn("Country not found for code: {}", code);  
 throw new ResponseStatusException(HttpStatus.NOT\_FOUND, "Country not found");  
 }  
 LOGGER.info("Country found: {}", country.getName());  
 return country;  
 } catch (Exception e) {  
 LOGGER.error("Error while getting country", e);  
 throw new ResponseStatusException(HttpStatus.INTERNAL\_SERVER\_ERROR, "Error retrieving country");  
 }  
 }  
}

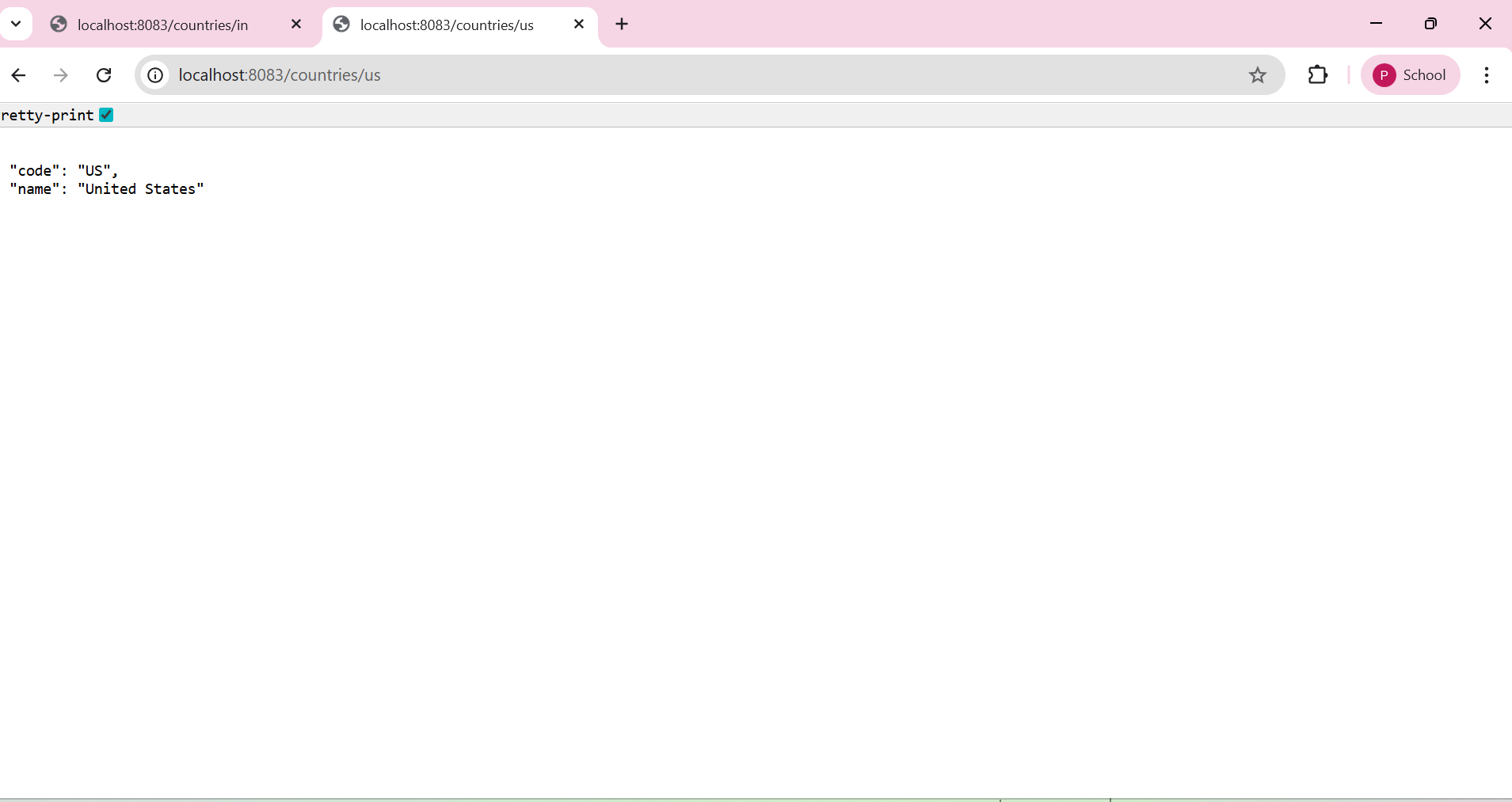
**How to run:**

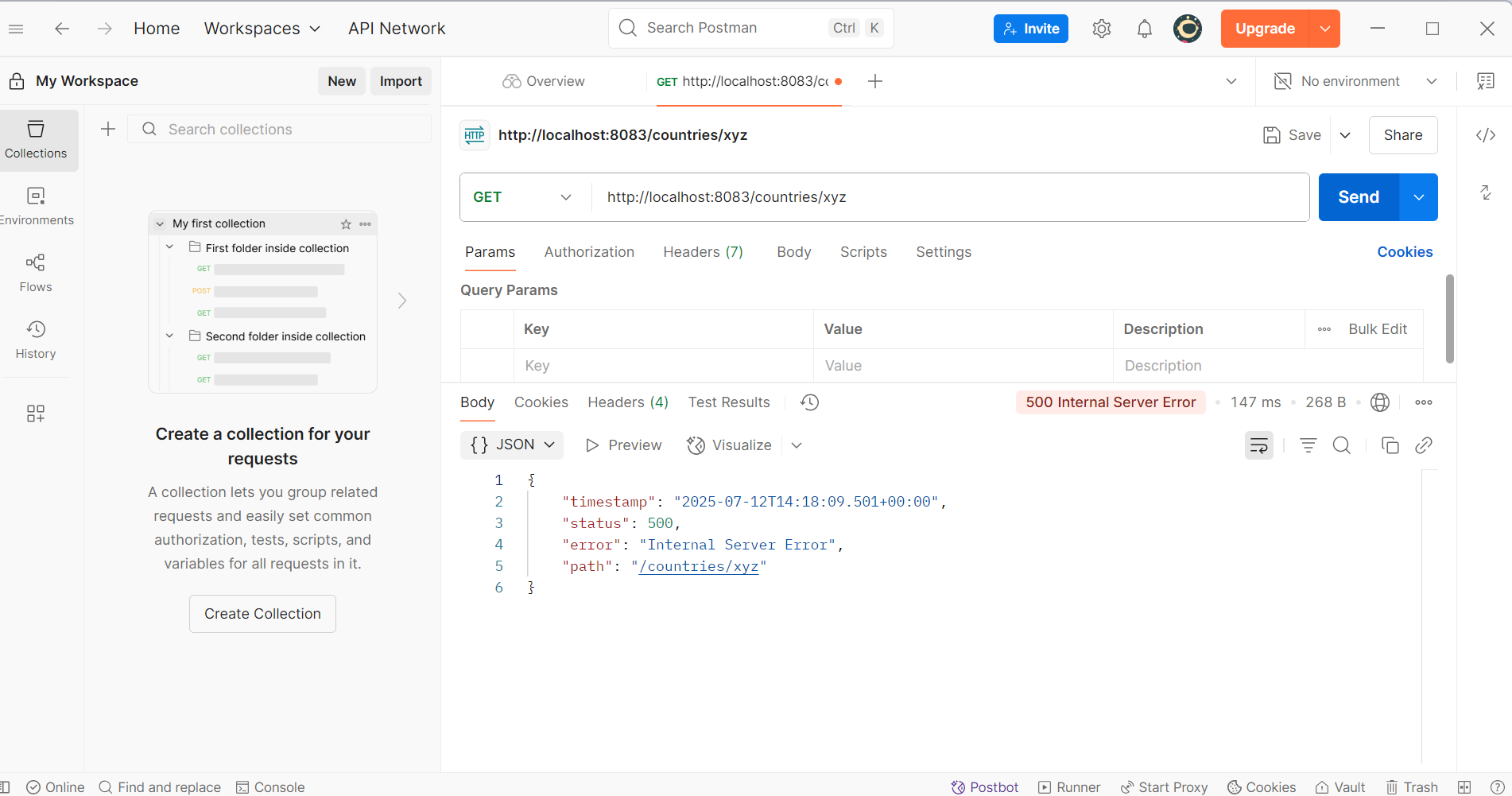
**GET /countries/{code}**

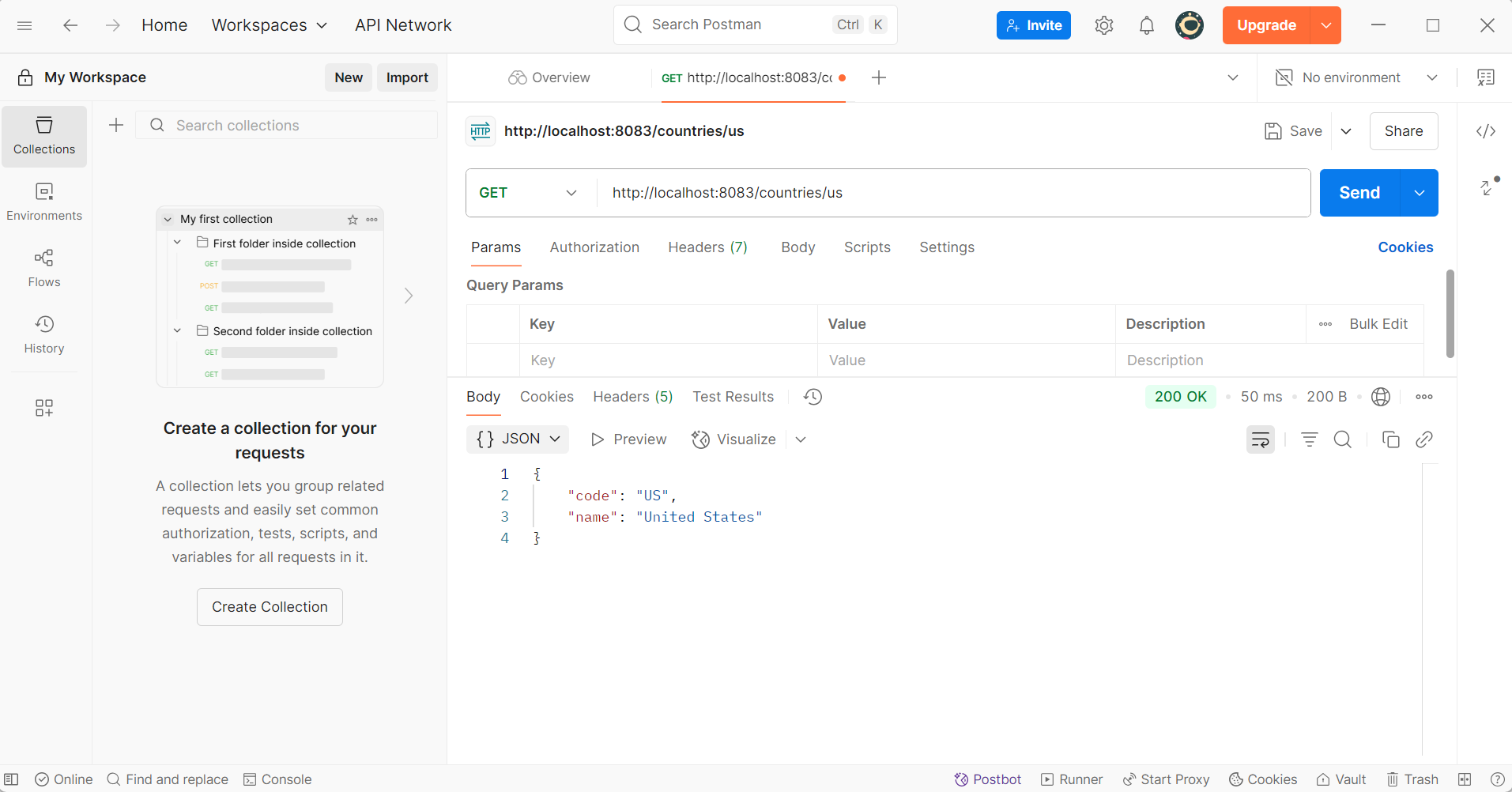
* **Description**: Returns a country from the list based on code.
* **Case insensitive** matching.
* **Sample Request**:

**GET http://localhost:8083/countries/us**

**Responses:**

****

****

****

**Exercise 6: Create authentication service that returns JWT**

**Project Structure:**

demo/

├── src/

│ └── main/

│ ├── java/

│ │ └── com/example/demo/

│ │ ├── DemoApplication.java

│ │ ├── config/SecurityConfig.java

│ │ ├── controller/AuthController.java

│ │ ├── model/AuthResponse.java

│ │ └── service/JwtService.java

│ └── resources/

│ └── application.properties

├── pom.xml

**1.pom.xml**

<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

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>demo</artifactId>

<version>1.0.0</version>

<packaging>jar</packaging>

<name>JWT Demo</name>

<description>Spring Boot project with JWT Auth</description>

<parent>

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

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

<version>3.1.0</version>

<relativePath/>

</parent>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<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>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.1</version>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**2.SecurityConfig.java**

package com.example.demo.config;

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.web.SecurityFilterChain;

@Configuration

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

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

.authorizeHttpRequests(auth -> auth.anyRequest().authenticated())

.httpBasic(Customizer.withDefaults());

return http.build();

}

}

**3.AuthController.java**

package com.example.demo.controller;

import com.example.demo.model.AuthResponse;

import com.example.demo.service.JwtService;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

import jakarta.servlet.http.HttpServletRequest;

import java.util.Base64;

@RestController

public class AuthController {

private final JwtService jwtService;

public AuthController(JwtService jwtService) {

this.jwtService = jwtService;

}

@GetMapping("/authenticate")

public ResponseEntity<AuthResponse> authenticate(HttpServletRequest request) {

String authHeader = request.getHeader("Authorization");

if (authHeader != null && authHeader.startsWith("Basic ")) {

String base64Credentials = authHeader.substring("Basic ".length());

byte[] credDecoded = Base64.getDecoder().decode(base64Credentials);

String credentials = new String(credDecoded);

String[] values = credentials.split(":", 2);

String username = values[0];

String password = values[1];

if ("user".equals(username) && "pwd".equals(password)) {

String token = jwtService.generateToken(username);

return ResponseEntity.ok(new AuthResponse(token));

}

}

return ResponseEntity.status(401).build();

}

}

**4.JwtService.java**

package com.example.demo.service;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Service;

import java.util.Date;

@Service

public class JwtService {

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

public String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

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

.setExpiration(new Date(System.currentTimeMillis() + 3600000)) // 1 hour

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

}

**5.AuthResponse.java**

package com.example.demo.model;

public class AuthResponse {

private String token;

public AuthResponse(String token) {

this.token = token;

}

public String getToken() {

return token;

}

public void setToken(String token) {

this.token = token;

}

}

**6.application.properties**

server.port=8080

🡪Run the app with:

mvn spring-boot:run

**Test Authentication**

**Using curl (Windows):**

curl.exe -i -u user: pwd <http://localhost:8080/authenticate>

**7.Output:**

**Output1:**

{

"token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiZXhwIjoxNzI2NjU4NzI5fQ.DyFmp0mEbzX2KU0kzCVaPUNL6CTgZevGc9Vt44rYcOo"

}

