**1. spring-rest-handson**

**Hands on 1**

**Create a Spring Web Project using Maven**

**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>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>spring-learn</name>

<description>Demo project for Spring Boot</description>

<url/>

<licenses>

<license/>

</licenses>

<developers>

<developer/>

</developers>

<scm>

<connection/>

<developerConnection/>

<tag/>

<url/>

</scm>

<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-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</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>

**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);

}

}

**Hands on 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*

*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>

* **Create Country class with following aspects:**
  + **Instance variables for code and name**
  + **Implement empty parameter constructor with inclusion of debug log within the constructor with log message as “Inside Country Constructor.”**
  + **Generate getters and setters with inclusion of debug with relevant message within each setter and getter method.**
  + **Generate toString() method**

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 + "]";

}

}

**Create a method displayCountry() in 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;

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) {

SpringApplication.*run*(SpringLearnApplication.class, args);

*displayCountry*();

}

public static void displayCountry() {

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

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

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

}

}

A screen shot of a computer

AI-generated content may be incorrect.

**2. spring-rest-handson**

**Hello World RESTful Web Service**

**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() {

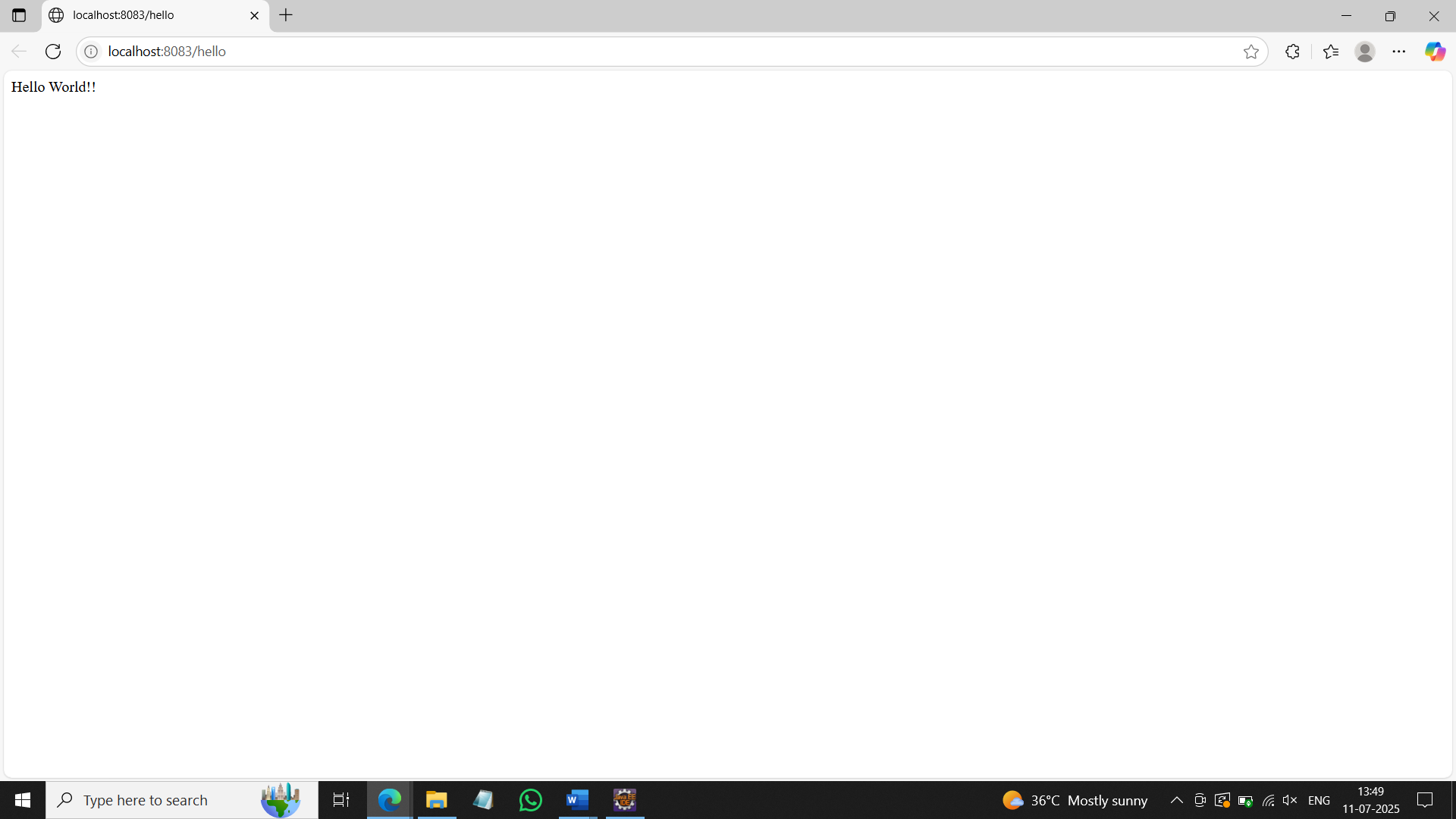
*LOGGER*.info("START");

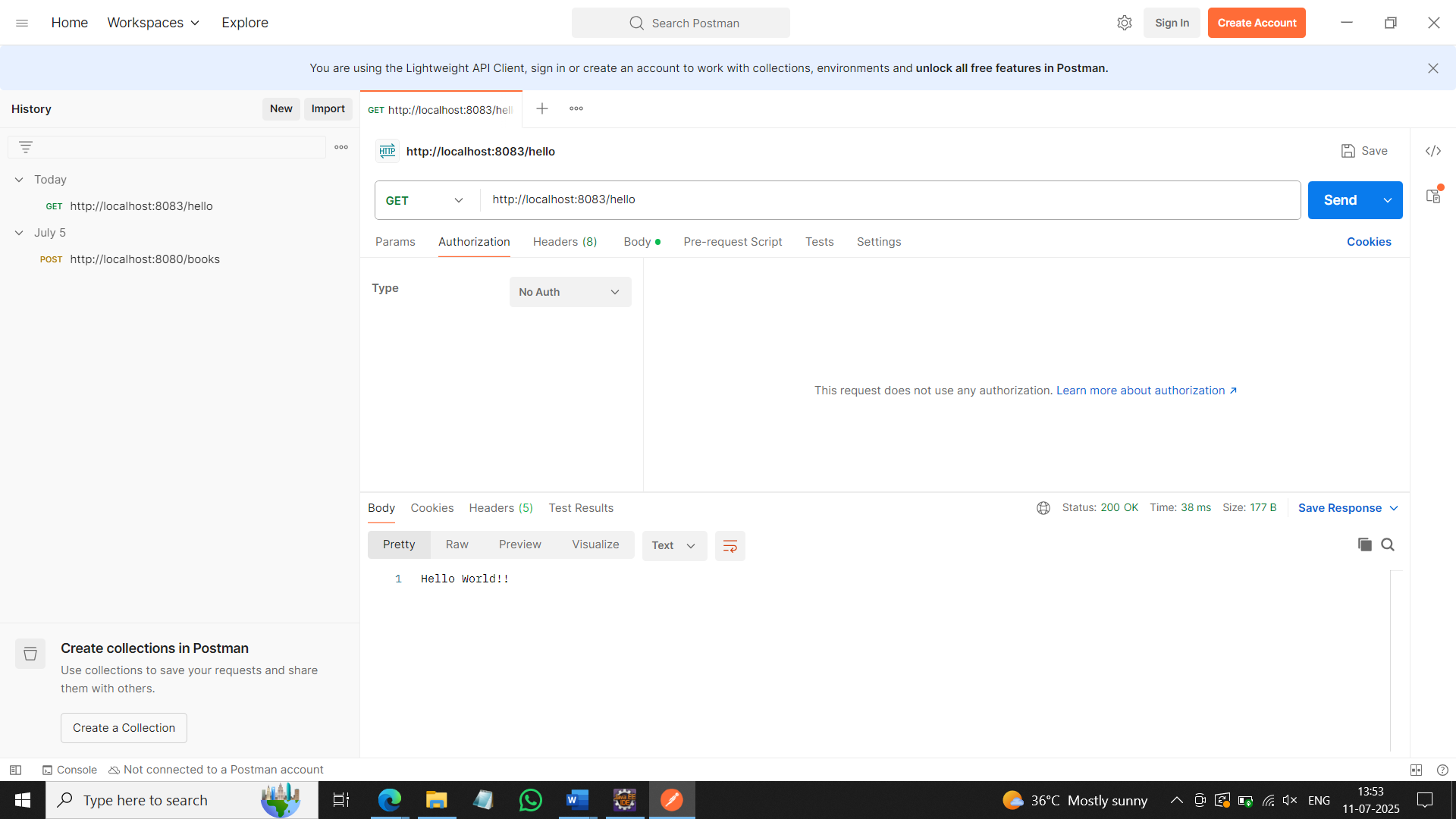
*LOGGER*.info("END");

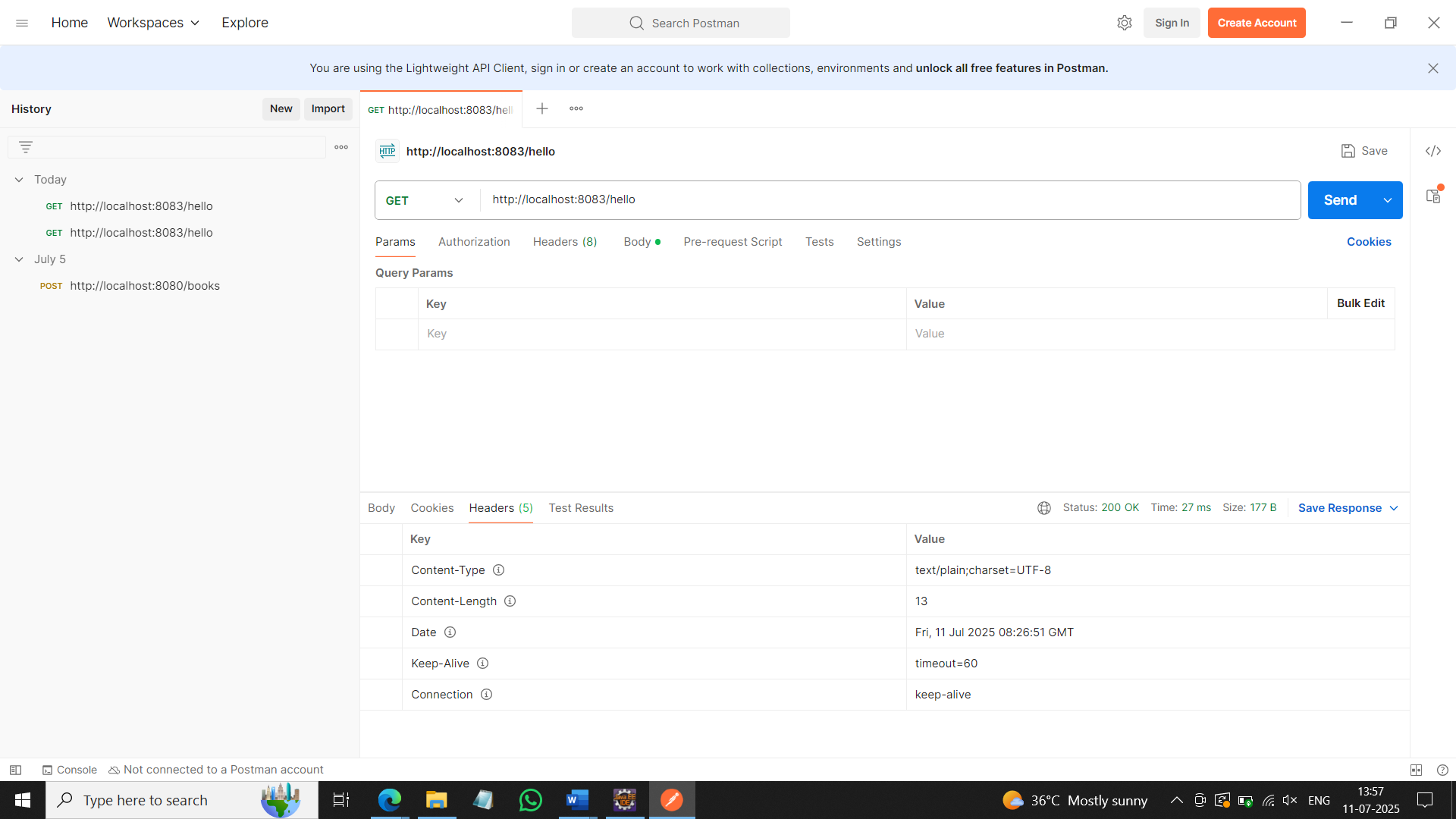
return "Hello World!!";

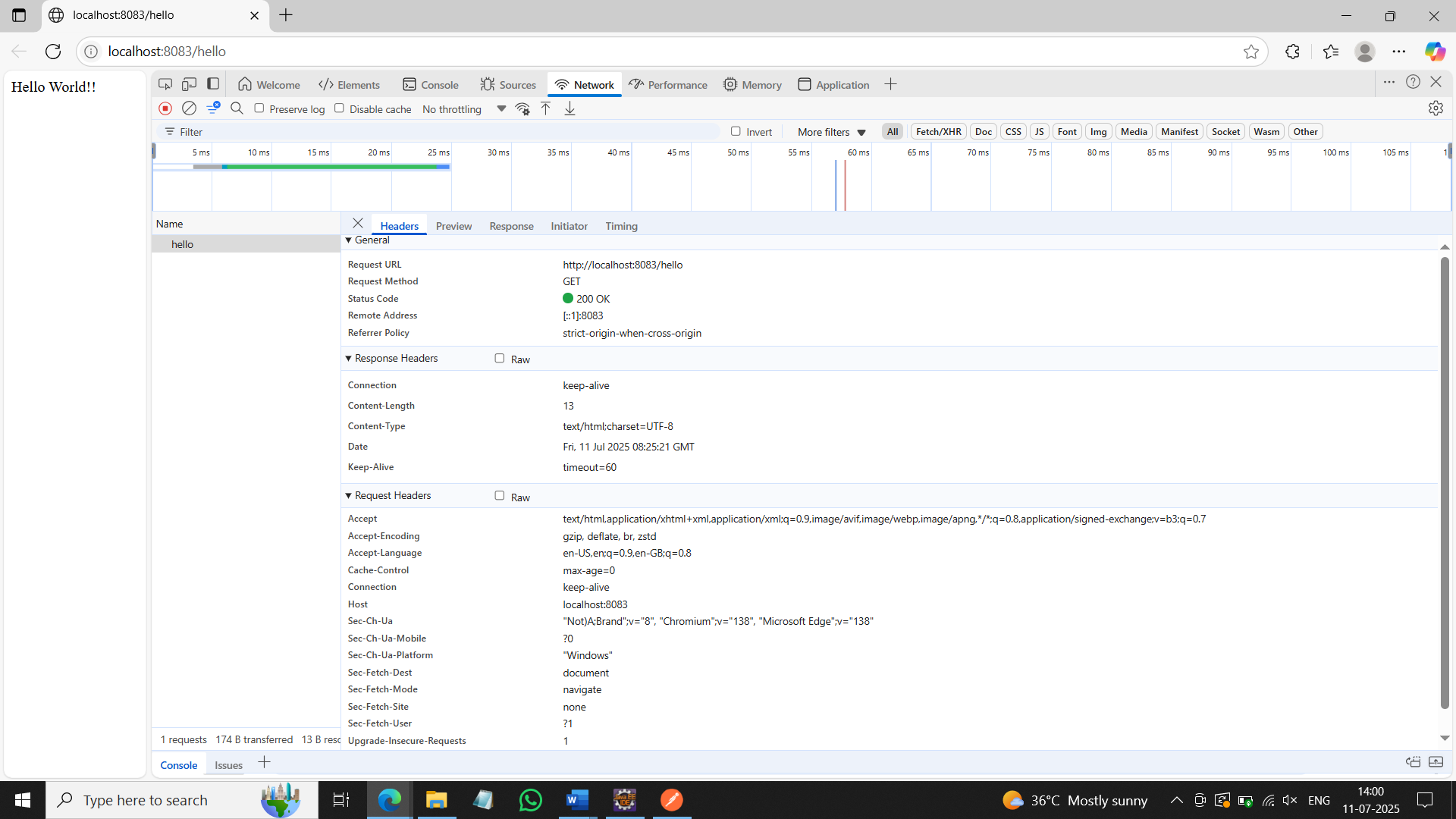
}

}









**REST - Country Web Service**

**CountryController.java:**

package com.cognizant.spring\_learn.controller;

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;

import com.cognizant.spring\_learn.Country;

@RestController

public class CountryController {

@RequestMapping("/country")

public Country getCountryIndia() {

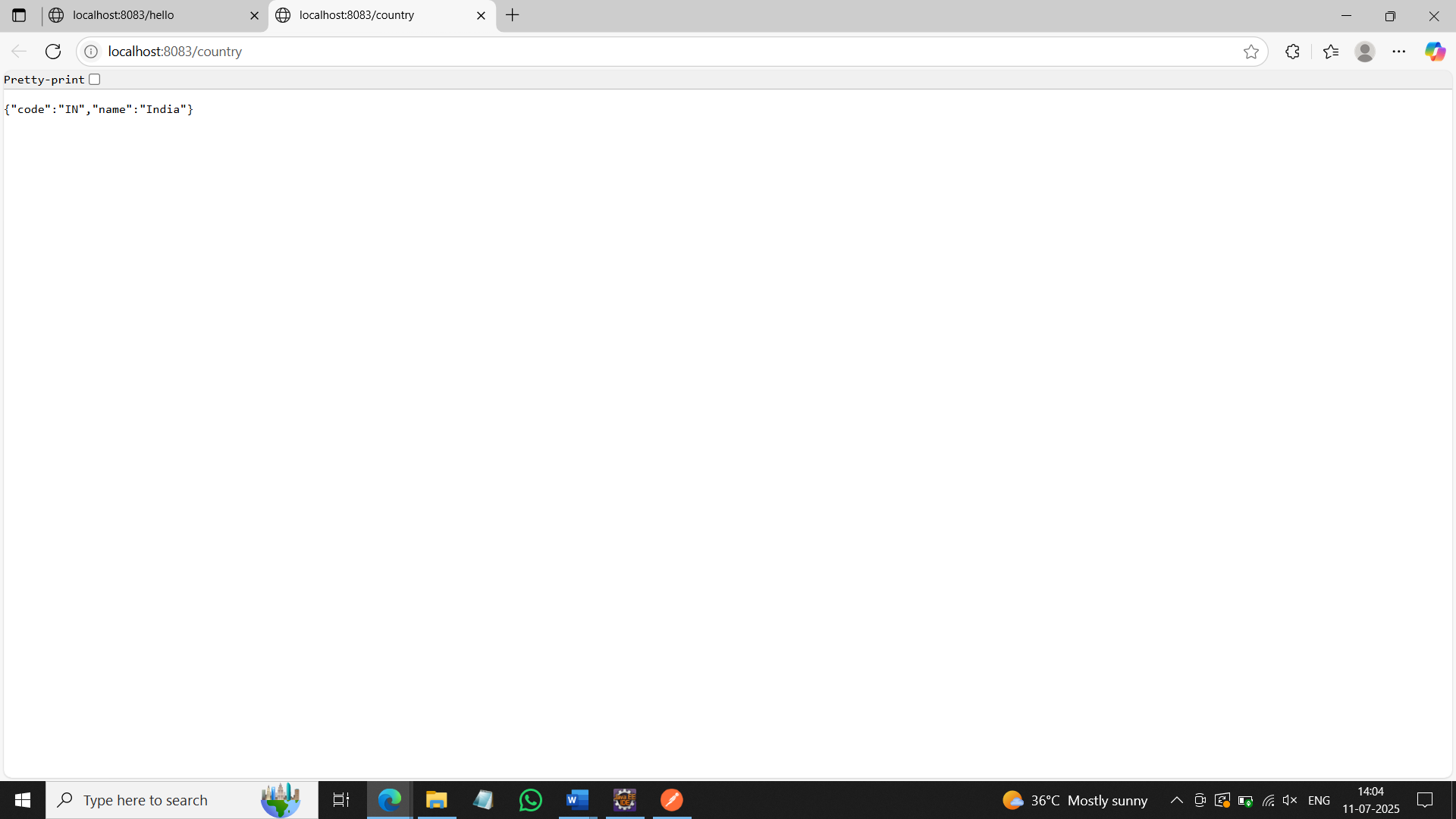
ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

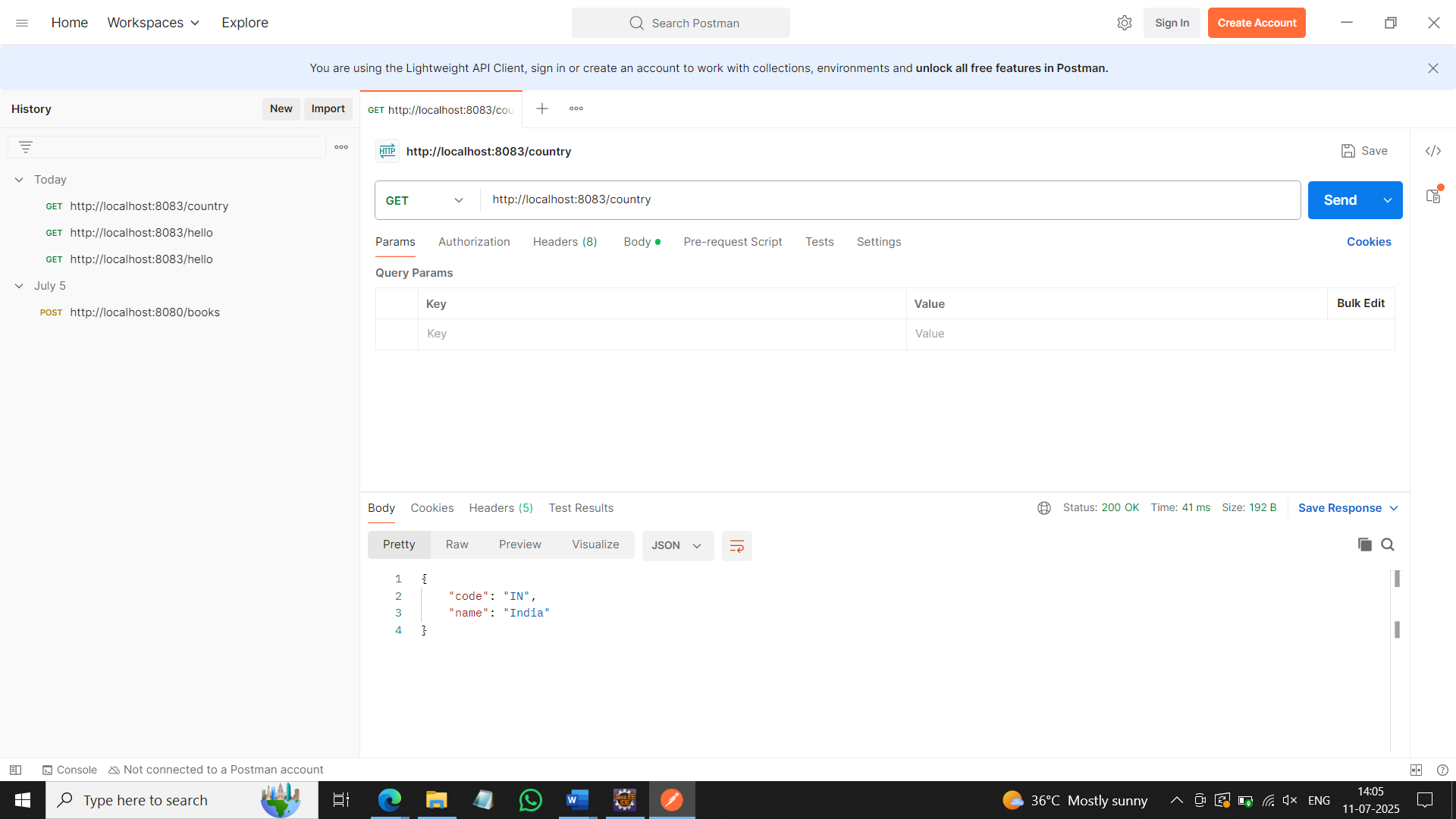
Country country = (Country) context.getBean("country");

return country;

}

}





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

**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=*"countryList"* class=*"java.util.ArrayList"*>

<constructor-arg>

<list>

<bean class=*"com.cognizant.springlearn.Country"*>

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

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

</bean>

<bean class=*"com.cognizant.springlearn.Country"*>

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

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

</bean>

<bean class=*"com.cognizant.springlearn.Country"*>

<property name=*"code"* value=*"CN"*/>

<property name=*"name"* value=*"China"*/>

</bean>

</list>

</constructor-arg>

</bean>

</beans>

**CountryService.java:**

package com.cognizant.spring\_learn.service;

import java.util.List;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import com.cognizant.spring\_learn.Country;

@Service

public class CountryService {

public Country getCountry(String code) {

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

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

return countryList.stream()

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

.findFirst()

.orElse(null);

}

}

**CountryController.java:**

package com.cognizant.spring\_learn.controller;

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

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

import com.cognizant.spring\_learn.Country;

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

@RestController

public class CountryController {

@Autowired

private CountryService countryService;

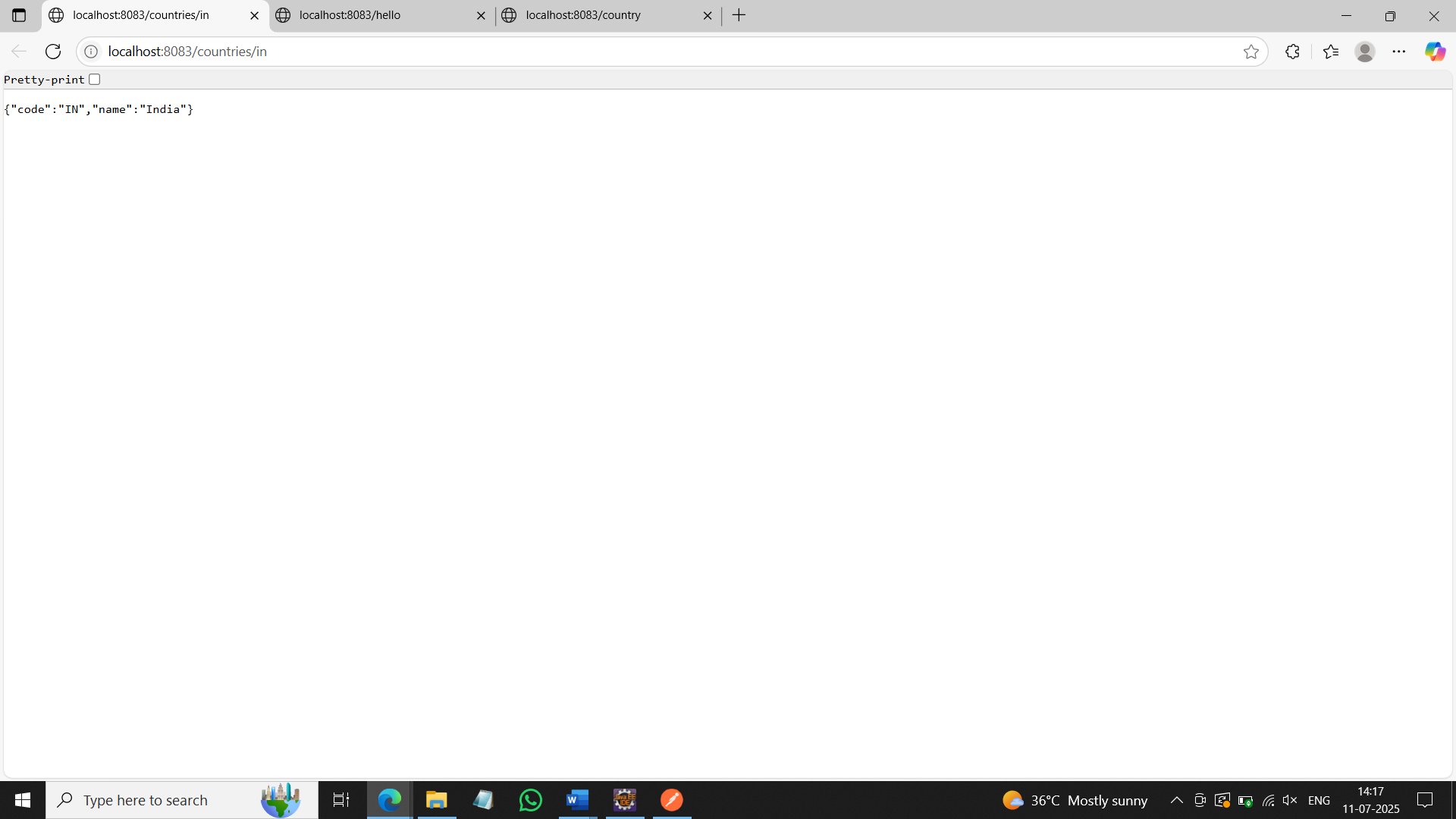
@GetMapping("/countries/{code}")

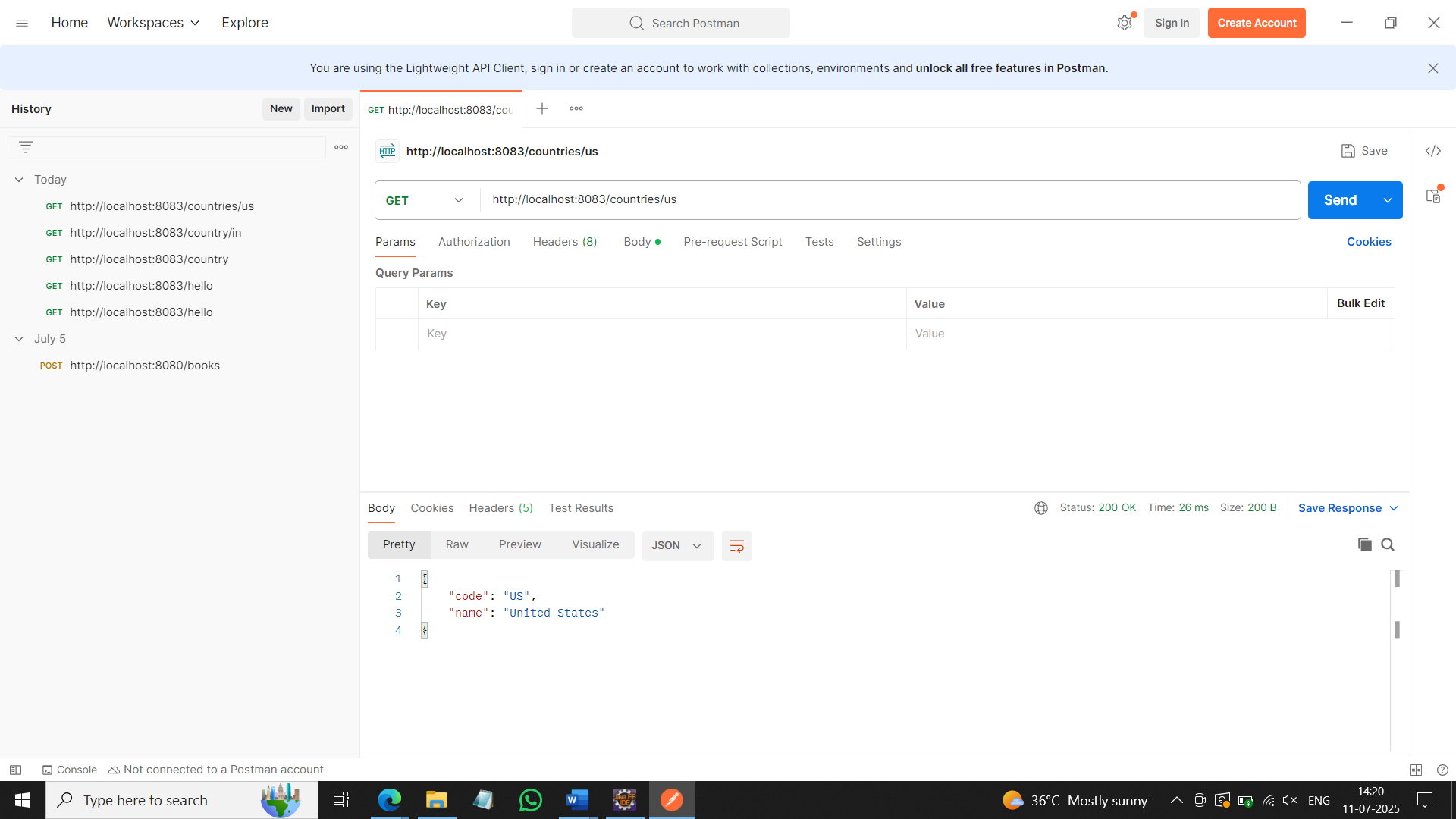
public Country getCountry(@PathVariable String code) {

return countryService.getCountry(code);

}

}





**5. JWT-handson**

**Create authentication service that returns JWT**

**JwtUtil.java:**

package com.cognizant.spring\_learn.util;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Component;

import java.util.Date;

@Component

public class JwtUtil {

private String secretKey = "mySecretKey";

public String generateToken(String username) {

return Jwts.*builder*()

.setSubject(username)

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

.setExpiration(new Date(System.*currentTimeMillis*() + 1000 \* 60 \* 60 \* 10)) // 10 hours validity

.signWith(SignatureAlgorithm.*HS256*, secretKey)

.compact();

}

}

**AuthenticationController.java:**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.util.JwtUtil;

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

import org.springframework.http.ResponseEntity;

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

import java.util.Base64;

@RestController

public class AuthenticationController {

@Autowired

private JwtUtil jwtUtil;

@RequestMapping(value="/authenticate", method=RequestMethod.*GET*)

public ResponseEntity<?> authenticate(@RequestHeader("Authorization") String authHeader) {

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

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

String decodedCredentials = new String(decodedBytes);

String[] userDetails = decodedCredentials.split(":", 2);

String username = userDetails[0];

String password = userDetails[1];

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

String token = jwtUtil.generateToken(username);

return ResponseEntity.*ok*("{\"token\":\"" + token + "\"}");

} else {

return ResponseEntity.*status*(401).body("Invalid credentials");

}

}

}

**SecurityConfig.java:**

package com.cognizant.spring\_learn.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.web.SecurityFilterChain;

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

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

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

.authorizeHttpRequests(auth -> auth

.requestMatchers("/authenticate").permitAll()

.anyRequest().authenticated()

)

.httpBasic();

return http.build();

}

}

