**WEEK – 4**

**1.Create a Spring Web Project using Maven**

**Spring Rest using Spring Boot**

**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...");

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

}

}

**Pom.xml (Important Dependencies)**

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

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<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>

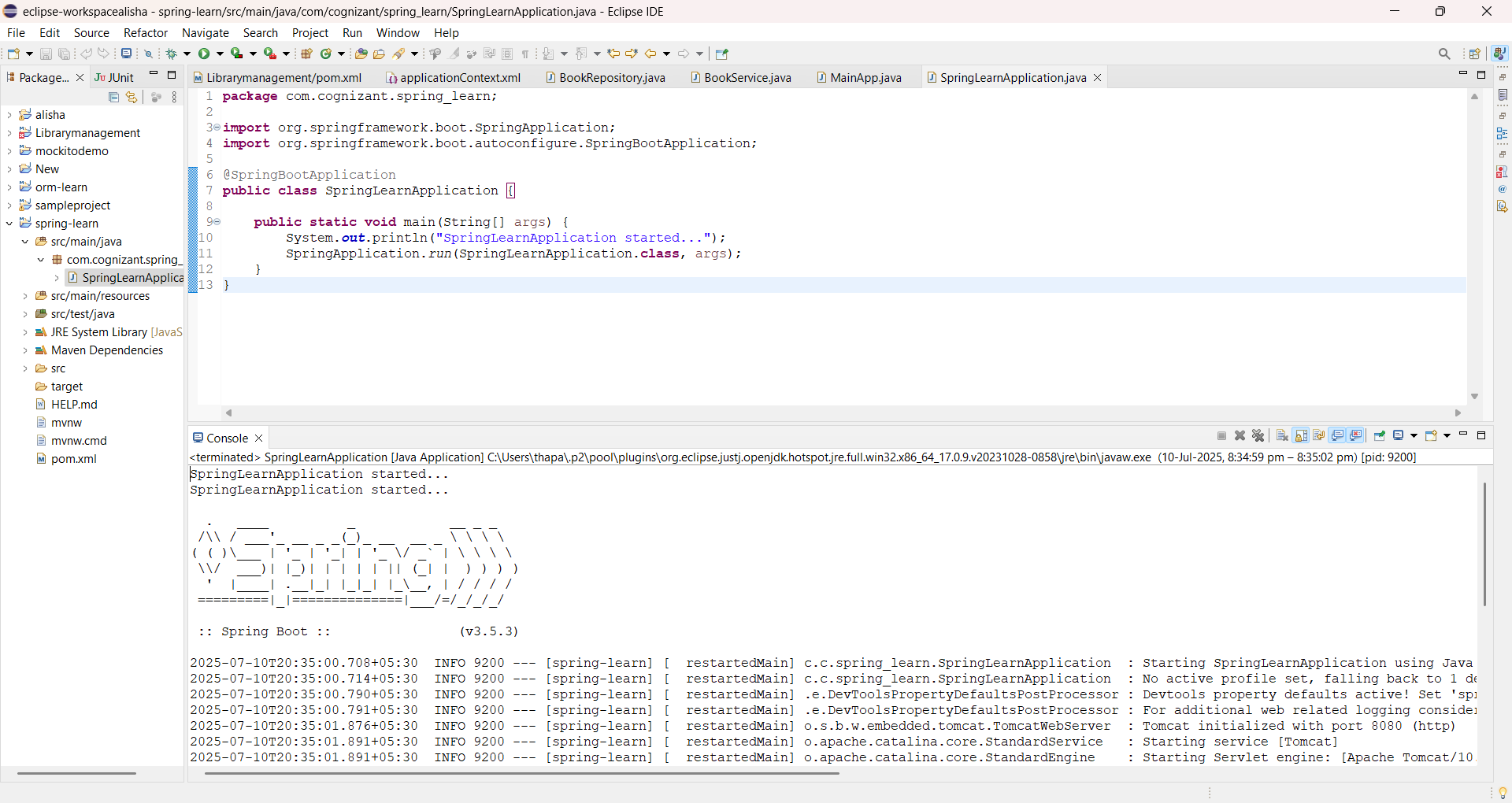
<optional>true</optional>

</dependency>

</dependencies>

</project>

**Output:**



**2.Spring Core – Load Country from Spring Configuration XML**

**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

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

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

<artifactId>country-project</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<maven.compiler.source>1.8</maven.compiler.source> <!-- Java version -->

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

<dependencies

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.33</version>

</dependency>

<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>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

<version>3.10.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**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.springlearn.Country">

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

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

</bean>

</beans>

**Country.java**

**package** com.cognizant.springlearn;

**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.java**

package com.cognizant.springlearn;

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) {

System.*out*.println("=== Inside Main Method ===");

*LOGGER*.debug("START main()");

*displayCountry*();

*LOGGER*.debug("END main()");

}

public static void displayCountry() {

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

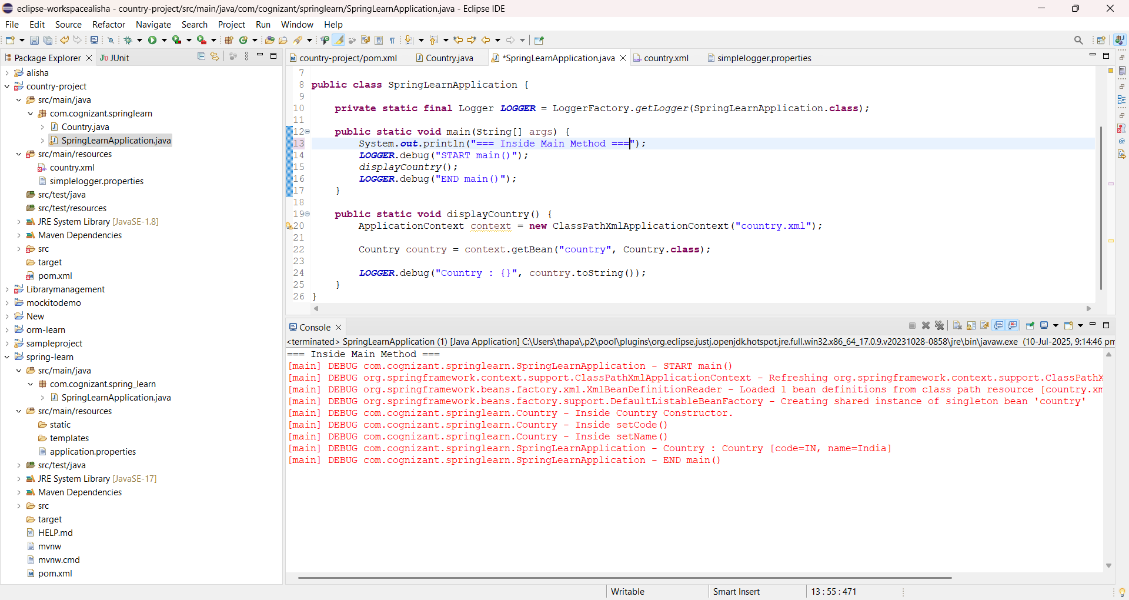
Country country = context.getBean("country", Country.class);

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

}

}

**Output:**



**3.Hello World RESTful Web Service**

**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*.debug("START sayHello()");

String message = "Hello World!!";

*LOGGER*.debug("END sayHello()");

return message;

}

}

**SpringLearnApplication.java**

**package** com.cognizant.spring\_learn;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan("com.cognizant.springlearn.controller")

**public** **class** SpringLearnApplication {

**public** **static** **void** main(String[] args) {

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

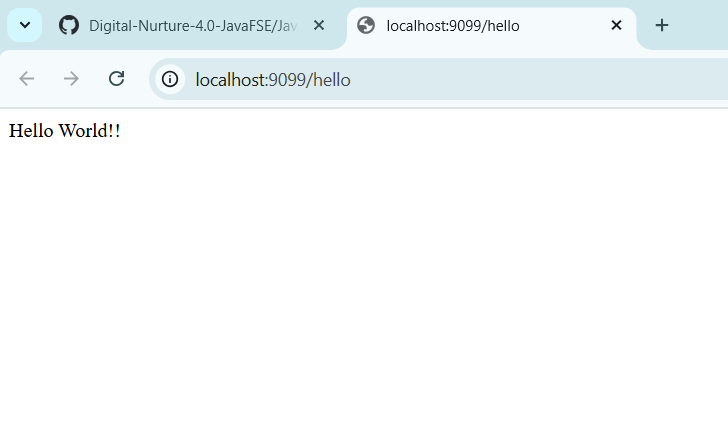
**application.properties**

spring.application.name=spring-learn

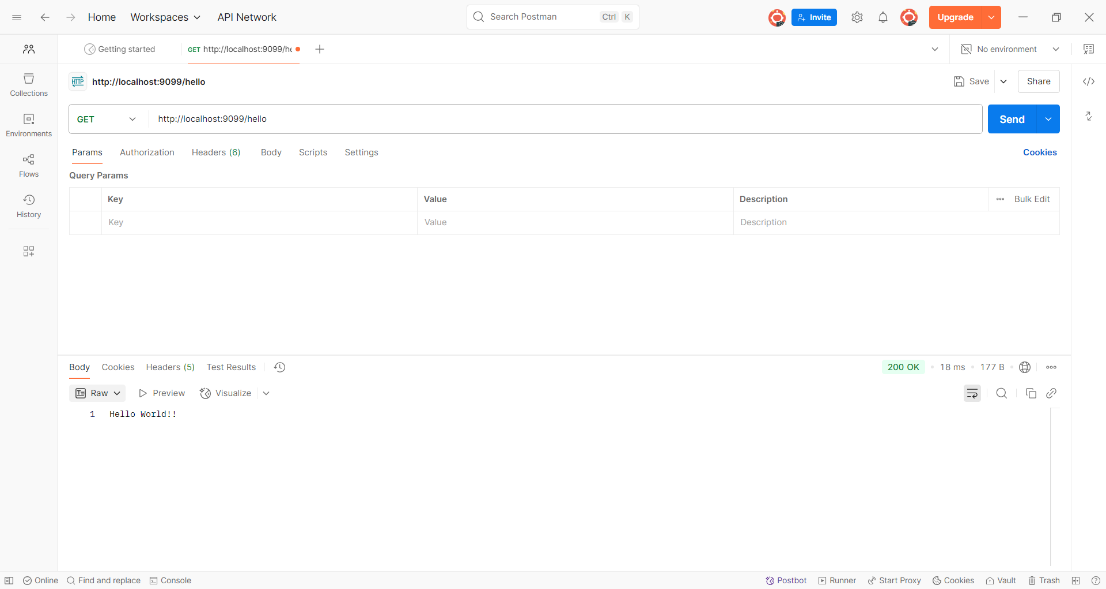
server.port=9099

**Output:**

* **Chrome:**

****

* Postman:



**4.Country Web Service**

**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/>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn-rest</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>spring-learn-rest</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-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>

**CountryController.java**

package com.cognizant.spring\_learn\_rest.controller;

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

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

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

@RestController

public class CountryController {

@GetMapping("/country")

public Country getCountryIndia() {

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

}

}

**Country.java**

**package** com.cognizant.spring\_learn\_rest.model;

**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;

}

}

**SpringLearnRestApplication.java**  
package com.cognizant.spring\_learn\_rest;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnRestApplication {

public static void main(String[] args) {

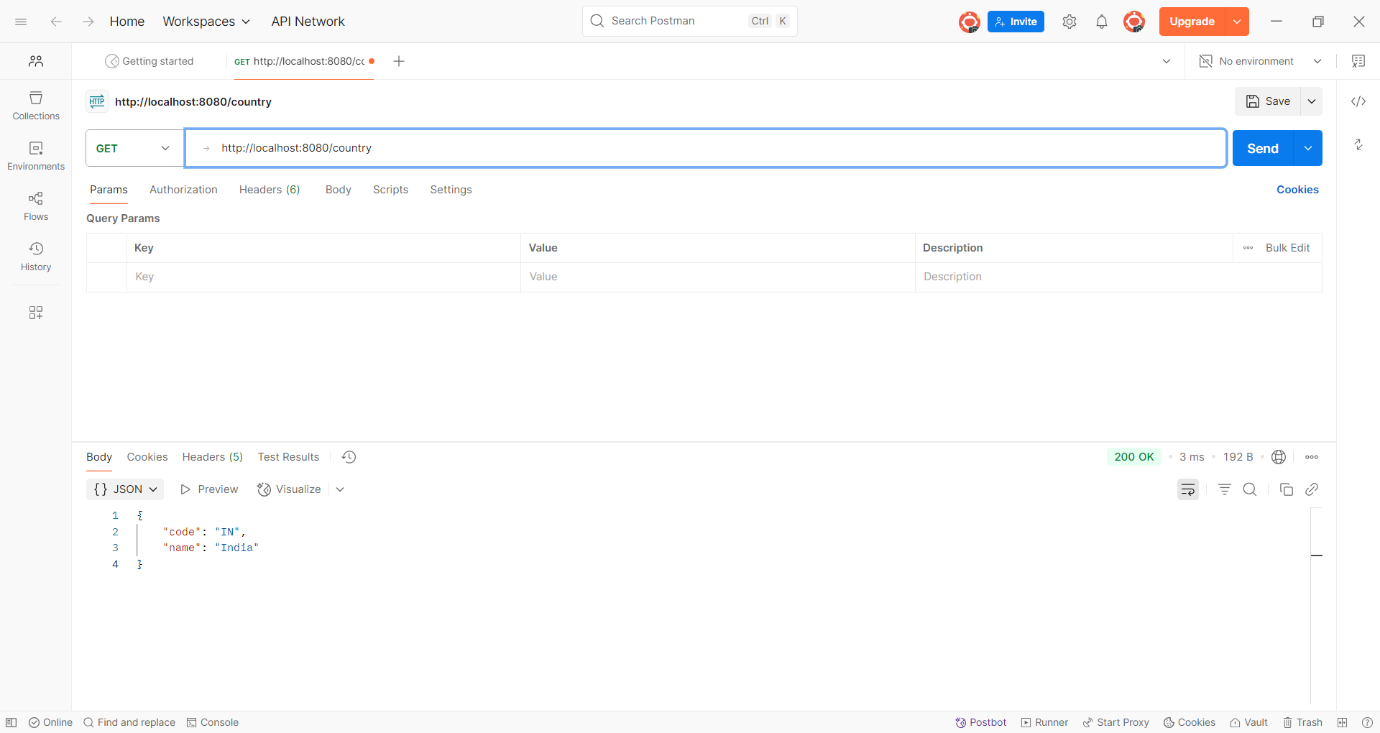
SpringApplication.run(SpringLearnRestApplication.class, args);

}

}

**Output:**





**5.Get country based on country code**

**CountryController.java**

package com.cognizant.spring\_learn\_rest.controller;

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

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

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

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

@RestController

public class CountryController {

@Autowired

private CountryService countryService;

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

public Country getCountry(@PathVariable String code) {

return countryService.getCountry(code);

}

}

**Country.java**

**package** com.cognizant.spring\_learn\_rest.model;

**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;

}

}

**CountryService.java**

**package** com.cognizant.spring\_learn\_rest.service;

**import** org.springframework.stereotype.Service;

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

**import** java.util.List;

@Service

**public** **class** CountryService {

**private** List<Country> countryList;

**public** Country getCountry(String code) {

countryList = List.*of*(

**new** Country("IN", "India"),

**new** Country("US", "United States"),

**new** Country("JP", "Japan")

);

**return** countryList.stream()

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

.findFirst()

.orElse(**null**);

}

}

**SpringLearnRestApplication.java**

package com.cognizant.spring\_learn\_rest;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnRestApplication {

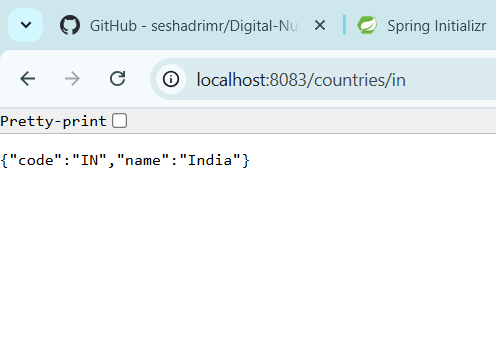
public static void main(String[] args) {

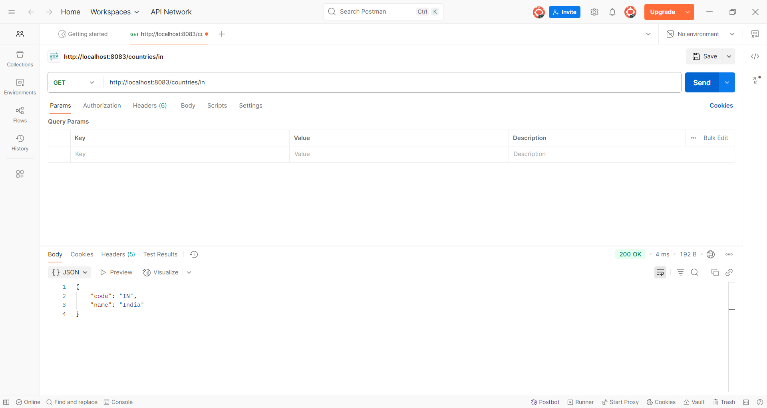
SpringApplication.run(SpringLearnRestApplication.class, args);

}

}

**Output:**





**6.Create authentication service that returns JWT**

**JwtUtil.java**

package com.cognizant.spring\_learn.service;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

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

import org.springframework.stereotype.Service;

import java.util.Date;

@Service

public class JwtUtil {

private final String SECRET\_KEY = "mysecretkey";

private final long EXPIRATION\_TIME = 1000 \* 60 \* 60; // 1 hour

public String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + EXPIRATION\_TIME))

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

public String extractUsername(String token) {

return Jwts.parser()

.setSigningKey(SECRET\_KEY)

.parseClaimsJws(token)

.getBody()

.getSubject();

}

public boolean validateToken(String token, UserDetails userDetails) {

final String username = extractUsername(token);

return username.equals(userDetails.getUsername());

}

}

AuthenticationController.java

package com.cognizant.spring\_learn.controller;

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

import org.springframework.http.ResponseEntity;

import org.springframework.security.core.Authentication;

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

import java.util.HashMap;

import java.util.Map;

@RestController

public class AuthenticationController {

private final JwtUtil jwtUtil;

public AuthenticationController(JwtUtil jwtUtil) {

this.jwtUtil = jwtUtil;

}

@GetMapping("/authenticate")

public ResponseEntity<?> authenticate(Authentication authentication) {

String username = authentication.getName(); // From Basic Auth

String token = jwtUtil.generateToken(username);

Map<String, String> response = new HashMap<>();

response.put("token", token);

return ResponseEntity.ok(response);

}

**JwtRequestFilter.java**

package com.cognizant.spring\_learn.filter;

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

import jakarta.servlet.FilterChain;

import jakarta.servlet.ServletException;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

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

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.context.SecurityContextHolder;

import org.springframework.security.core.userdetails.\*;

import org.springframework.stereotype.Component;

import org.springframework.web.filter.OncePerRequestFilter;

import java.io.IOException;

@Component

public class JwtRequestFilter extends OncePerRequestFilter {

@Autowired

private JwtUtil jwtUtil;

@Autowired

private UserDetailsService userDetailsService;

@Override

protected void doFilterInternal(HttpServletRequest request,

HttpServletResponse response,

FilterChain chain)

throws ServletException, IOException {

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

String username = null;

String jwt = null;

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

jwt = authHeader.substring(7);

try {

username = jwtUtil.extractUsername(jwt);

} catch (Exception e) {

System.out.println("Invalid JWT: " + e.getMessage());

}

}

if (username != null && SecurityContextHolder.getContext().getAuthentication() == null) {

UserDetails userDetails = userDetailsService.loadUserByUsername(username);

if (jwtUtil.validateToken(jwt, userDetails)) {

UsernamePasswordAuthenticationToken token =

new UsernamePasswordAuthenticationToken(userDetails, null, userDetails.getAuthorities());

SecurityContextHolder.getContext().setAuthentication(token);

}

}

chain.doFilter(request, response);

}

}

**SecurityConfig.java**

package com.cognizant.spring\_learn.config;

import com.cognizant.spring\_learn.filter.JwtRequestFilter;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

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

import org.springframework.security.core.userdetails.\*;

import org.springframework.security.crypto.password.NoOpPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.security.web.SecurityFilterChain;

import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;

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

@Configuration

public class SecurityConfig {

@Autowired

private JwtRequestFilter jwtRequestFilter;

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeHttpRequests()

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

.anyRequest().authenticated()

.and()

.httpBasic();

http.addFilterBefore(jwtRequestFilter, UsernamePasswordAuthenticationFilter.class);

return http.build();

}

@Bean

public UserDetailsService userDetailsService() {

UserDetails user = User.withUsername("user")

.password("pwd")

.roles("USER")

.build();

return new InMemoryUserDetailsManager(user);

}

@Bean

public PasswordEncoder passwordEncoder() {

return NoOpPasswordEncoder.getInstance(); // For demo only

}

}

**HelloController.java**

package com.cognizant.spring\_learn.controller;

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

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

@RestController

public class HelloController {

@GetMapping("/hello")

public String hello() {

return "Hello, secure world!";

}

}

**Output:**

