**WEEK-4**

(Implemented in Eclipse)

**Spring REST using Spring Boot 3**

**1) 2. Hands on 5 Get Country Based on country code**

country.xml

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:context="http://www.springframework.org/schema/context"

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

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context.xsd">

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

<constructor-arg>

<list>

<bean class="com.cognizant.spring-learn.model.Country1">

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

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

</bean>

<bean class="com.cognizant.spring-learn.model.Country1">

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

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

</bean>

</list>

</constructor-arg>

</bean>

</beans>

Country.java

**package** com.cognizant.springlearn.model;

**public** **class** Country {

**private** String code;

**private** String name;

// Getters and Setters

**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.springlearn.service;

**import** com.cognizant.springlearn.model.Country;

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

**import** org.springframework.core.io.ClassPathResource;

**import** org.springframework.oxm.jaxb.Jaxb2Marshaller;

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

**import** javax.xml.transform.stream.StreamSource;

**import** java.util.List;

@Service

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

@Autowired

**private** Jaxb2Marshaller marshaller;

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

CountryList countryList = (CountryList) marshaller.unmarshal(

**new** StreamSource(**new** ClassPathResource("country.xml").getInputStream()));

**return** countryList.getCountries().stream()

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

.findFirst()

.orElseThrow(() -> **new** Exception("Country not found"));

}

}

CountryList.java

**package** com.cognizant.springlearn.model;

**import** javax.xml.bind.annotation.XmlElement;

**import** javax.xml.bind.annotation.XmlRootElement;

**import** java.util.List;

@XmlRootElement(name = "countries")

**public** **class** CountryList {

**private** List<Country> countries;

@XmlElement(name = "country")

**public** List<Country> getCountries() {

**return** countries;

}

**public** **void** setCountries(List<Country> countries) {

**this**.countries = countries;

}

}

CountryController.java

**package** com.cognizant.springlearn.controller;

**import** com.cognizant.springlearn.model.Country;

**import** com.cognizant.springlearn.service.CountryService;

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

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

@RestController

**public** **class** CountryController {

@Autowired

**private** CountryService countryService;

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

**public** Country getCountry(@PathVariable String code) **throws** Exception {

**return** countryService.getCountry(code);

}

}

SpringLearnApplication.java

**package** com.cognizant.springlearn;

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

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

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

**import** org.springframework.oxm.jaxb.Jaxb2Marshaller;

**import** com.cognizant.springlearn.model.Country;

**import** com.cognizant.springlearn.model.CountryList;

@SpringBootApplication

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

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

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

}

@Bean

**public** Jaxb2Marshaller marshaller() {

Jaxb2Marshaller marshaller = **new** Jaxb2Marshaller();

marshaller.setClassesToBeBound(CountryList.**class**, Country.**class**);

**return** marshaller;

}

}

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>3.1.5</version>

<relativePath/>

</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>

<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>

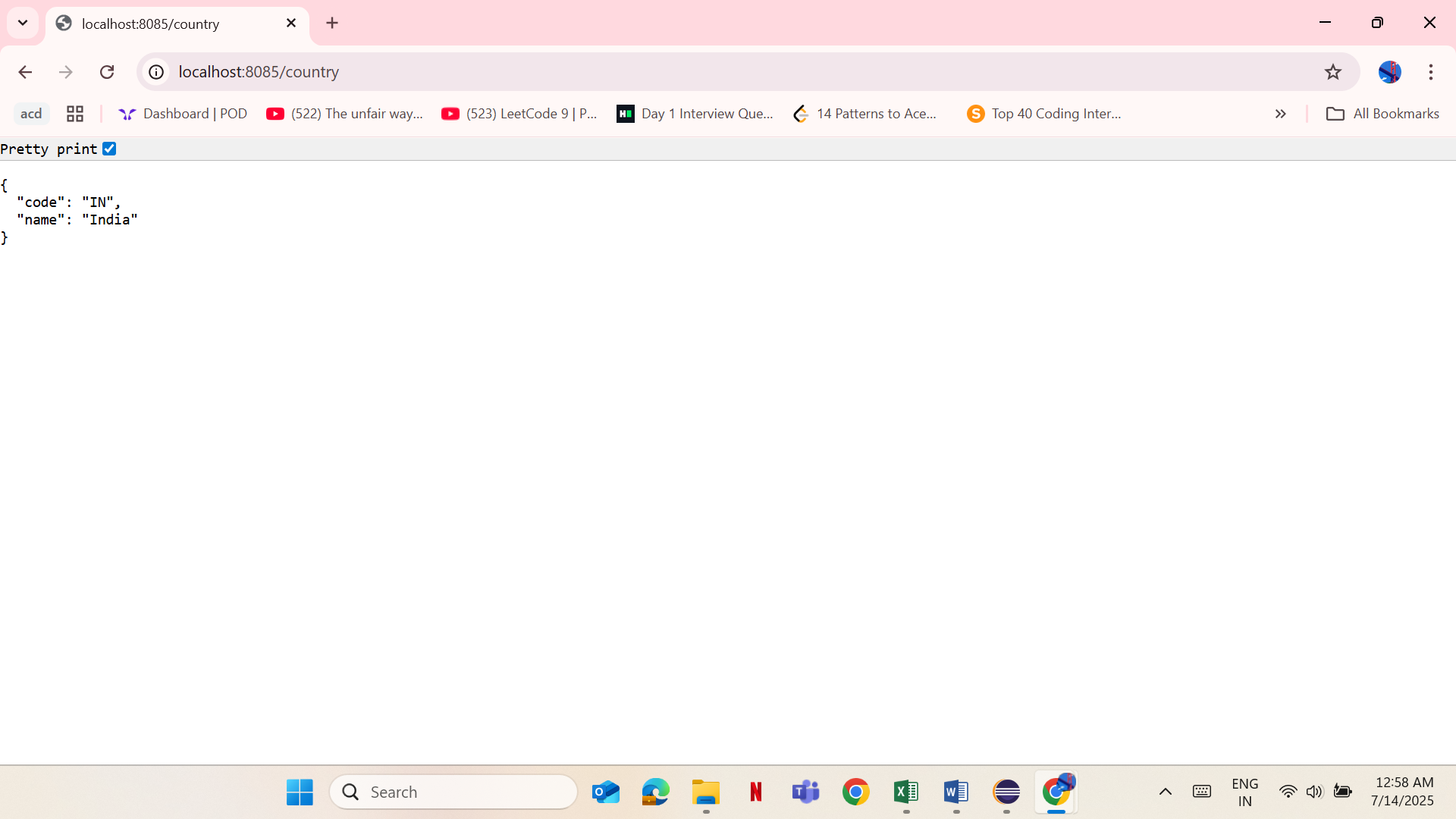
</plugin>

</plugins>

</build>

</project>

**Output:**



**2) 5. Create Authentication Service that returns JWT**

JwtAuthServiceApplication.java

**package** com.example.demo;

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

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

@SpringBootApplication

**public** **class** JwtAuthServiceApplication {

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

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

}

}

JwtUtil.java

**package** com.example.demo;

**import** io.jsonwebtoken.Jwts;

**import** io.jsonwebtoken.SignatureAlgorithm;

**import** io.jsonwebtoken.security.Keys;

**import** org.springframework.stereotype.Component;

**import** java.security.Key;

**import** java.util.Date;

@Component

**public** **class** JwtUtil {

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

// Secret key for signing

**private** **final** Key key = Keys.*secretKeyFor*(SignatureAlgorithm.***HS256***);

**public** String generateToken(String username) {

**return** Jwts.*builder*()

.setSubject(username)

.setIssuedAt(**new** Date())

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

.signWith(key)

.compact();

}

}

AuthenticationController.java

**package** com.example.demo.controller;

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

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.security.core.Authentication;

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

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

**import** com.example.demo.JwtUtil;

**import** java.util.HashMap;

**import** java.util.Map;

@RestController

**public** **class** AuthenticationController {

@Autowired

**private** JwtUtil jwtUtil;

@GetMapping("/authenticate")

**public** ResponseEntity<Map<String, String>> authenticate(Authentication authentication) {

// Get the username from the authentication object provided by Spring Security

String username = authentication.getName();

// Generate JWT token using the username

String token = jwtUtil.generateToken(username);

// Prepare the response

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

response.put("token", token);

**return** ResponseEntity.*ok*(response);

}

}

SecurityConfig.java

**package** com.example.demo;

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

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

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

**import** org.springframework.security.provisioning.InMemoryUserDetailsManager;

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

**import** org.springframework.security.web.SecurityFilterChain;

@Configuration

**public** **class** SecurityConfig {

// Define in-memory user

@Bean

**public** InMemoryUserDetailsManager userDetailsService() {

**return** **new** InMemoryUserDetailsManager(

User.~~withDefaultPasswordEncoder~~()

.username("user")

.password("pwd")

.roles("USER")

.build()

);

}

// Configure security rules

@Bean

**public** SecurityFilterChain securityFilterChain(HttpSecurity http) **throws** Exception {

http

.~~csrf~~().disable()

.~~authorizeHttpRequests~~()

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

.anyRequest().permitAll()

.~~and~~()

.~~httpBasic~~(); // Enables basic authentication

**return** http.build();

}

}

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.example.demo</groupId>

<artifactId>jwt-auth-service</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>jwt-auth-service</name>

<description>JWT Token Generator</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-security</artifactId>

</dependency>

<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>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-test</artifactId>

<scope>test</scope>

</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>

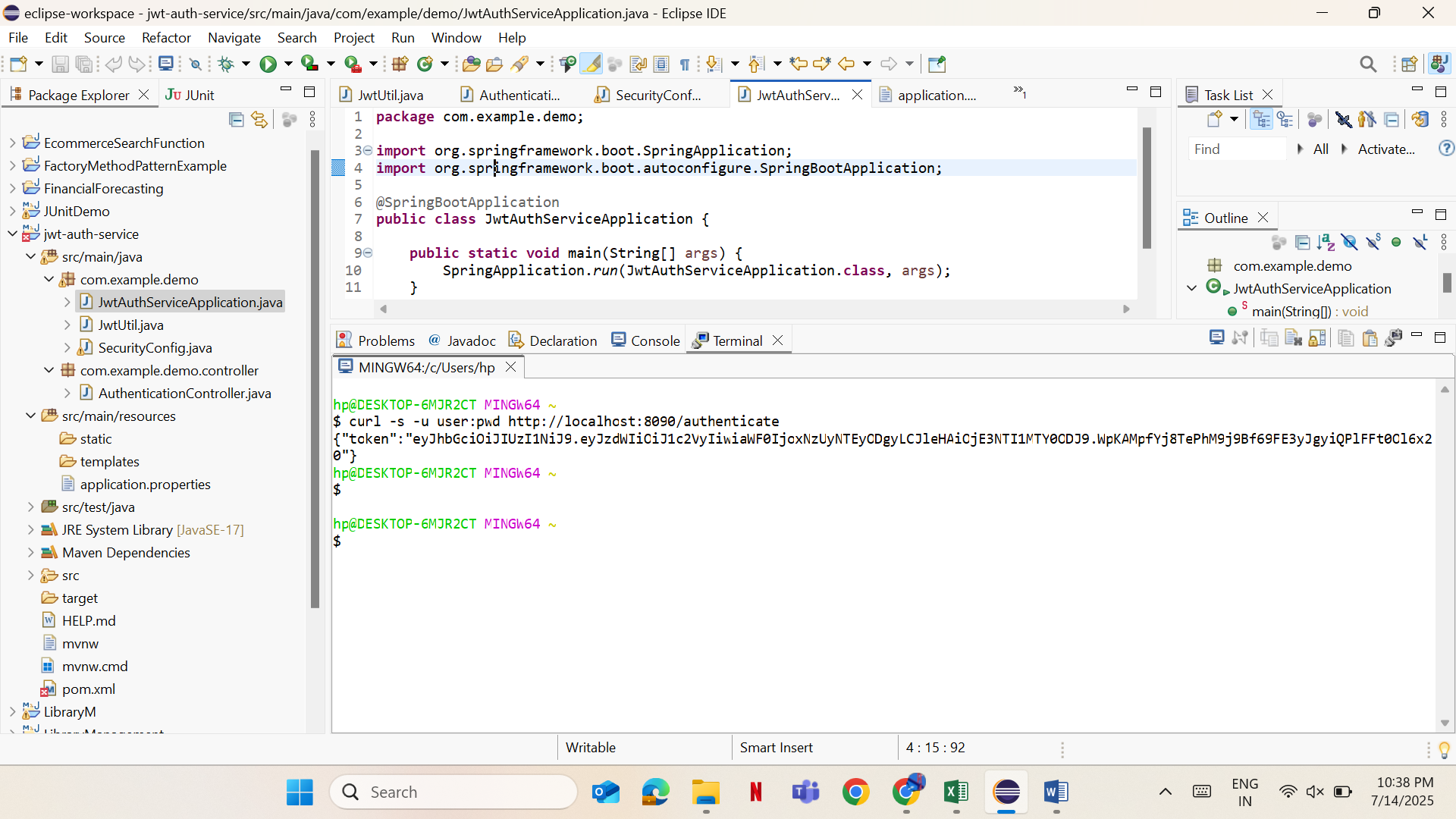
</plugin>

</plugins>

</build>

</project>

**Output:**

****