**EXERCISE 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.info("Start of sayHello() method");

        String response = "Hello World!!";

        LOGGER.info("End of sayHello() method");

        return response;

    }

}

SpringApplication.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);

    }

}

Application.properties

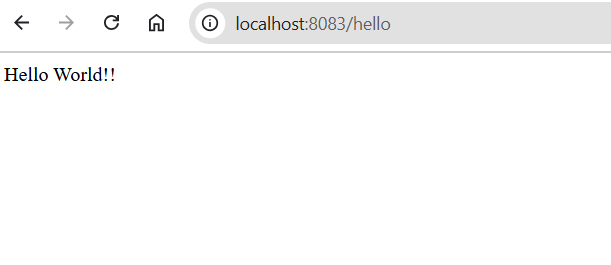
# Server port configuration

server.port=8083

# Logging configuration

logging.level.com.cognizant.springlearn=DEBUG

OUTPUT:



**EXERCISE 4**: **REST - COUNTRY WEB SERVICE**

CountryController.java

package com.cognizant.springlearn.controller;

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

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

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

import com.cognizant.springlearn.model.Country;

import org.springframework.context.support.ClassPathXmlApplicationContext;

@RestController

@RequestMapping("/country")

public class CountryController {

    @GetMapping

    public Country getCountryIndia() {

        ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("beans.xml");

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

        context.close();

        return country;

    }

}

Country.java

package com.cognizant.springlearn.model;

public class Country {

    private String code;

    private String name;

    // Constructors

    public Country() {

    }

    public Country(String code, String name) {

        this.code = code;

        this.name = 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;

    }

    @Override

    public String toString() {

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

    }

}

SpringApplication.java

package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ImportResource;

@SpringBootApplication

@ImportResource({"classpath:beans.xml"})

public class SpringLearnApplication {

    public static void main(String[] args) {

        SpringApplication.run(SpringLearnApplication.class, args);

    }

}

Beans.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="in" class="com.cognizant.springlearn.model.Country">

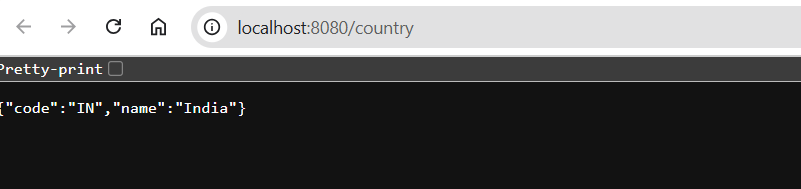
        <property name="code" value="IN"/>

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

    </bean>

</beans>

OUTPUT:



**EXERCISE 5**:**REST - GET COUNTRY BASED ON COUNTRY CODE**

CountryController.java

package com.cognizant.springlearn.controller;

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

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

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

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

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

import com.cognizant.springlearn.model.Country;

import com.cognizant.springlearn.service.CountryService;

@RestController

@RequestMapping("/countries")

public class CountryController {

    @Autowired

    private CountryService countryService;

    @GetMapping("/{code}")

    public Country getCountry(@PathVariable String code) {

        return countryService.getCountry(code);

    }

}

Country.java

package com.cognizant.springlearn.model;

public class Country {

    private String code;

    private String name;

    // Constructors, getters, setters

    public Country() {}

    public Country(String code, String name) {

        this.code = code;

        this.name = 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 java.util.List;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import com.cognizant.springlearn.model.Country;

@Service

public class CountryService {

    public Country getCountry(String code) {

        ClassPathXmlApplicationContext context =

            new ClassPathXmlApplicationContext("country.xml");

        List<Country> countries =

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

        context.close();

        return countries.stream()

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

            .findFirst()

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

    }

}

SpringApplication.java

package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ImportResource;

@SpringBootApplication

@ImportResource({"classpath:country.xml"})

public class SpringLearnApplication {

    public static void main(String[] args) {

        SpringApplication.run(SpringLearnApplication.class, args);

    }

}

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="in" class="com.cognizant.springlearn.model.Country">

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

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

    </bean>

    <bean id="us" class="com.cognizant.springlearn.model.Country">

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

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

    </bean>

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

        <constructor-arg>

            <list>

                <ref bean="in"/>

                <ref bean="us"/>

                <!-- Add more countries as needed -->

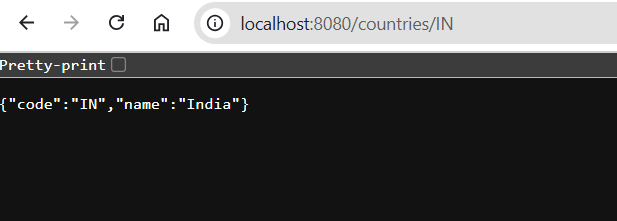
            </list>

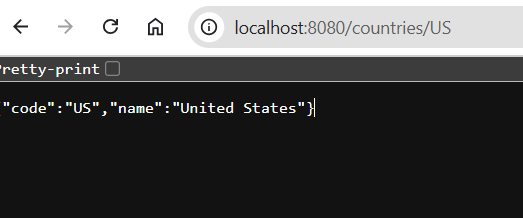
        </constructor-arg>

    </bean>

</beans>

OUTPUT:





**EXERCISE 6:** **CREATE AUTHENTICATION SERVICE THAT RETURNS JWT**

**AuthenticationController.java**

package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.config.JwtTokenUtil;

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

import org.springframework.http.ResponseEntity;

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

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

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

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

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

import javax.servlet.http.HttpServletRequest;

import java.util.Base64;

import java.util.HashMap;

import java.util.Map;

@RestController

public class AuthenticationController {

@Autowired

private JwtTokenUtil jwtTokenUtil;

@Autowired

private InMemoryUserDetailsManager inMemoryUserDetailsManager;

@Autowired

private PasswordEncoder passwordEncoder;

@GetMapping("/authenticate")

public ResponseEntity<?> createAuthenticationToken(HttpServletRequest request) {

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

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

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

String credentials = new String(Base64.getDecoder().decode(base64Credentials));

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

if (values.length == 2) {

String username = values[0];

String password = values[1];

try {

UserDetails userDetails = inMemoryUserDetailsManager.loadUserByUsername(username);

if (passwordEncoder.matches(password, userDetails.getPassword())) {

String token = jwtTokenUtil.generateToken(username);

return ResponseEntity.ok(new HashMap<>() {{

put("token", token);

}});

}

} catch (Exception e) {

return ResponseEntity.badRequest().body("Invalid credentials");

}

}

}

return ResponseEntity.badRequest().body("Authentication failed");

}

}

**SecurityConfig.java**

package com.cognizant.springlearn.config;

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.config.annotation.web.configuration.EnableWebSecurity;

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

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

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

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

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

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter {

@Override

protected void configure(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeRequests()

.antMatchers("/authenticate").permitAll()

.anyRequest().authenticated();

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Bean

public InMemoryUserDetailsManager inMemoryUserDetailsManager() {

UserDetails user = User.builder()

.username("user")

.password(passwordEncoder().encode("pwd"))

.roles("USER")

.build();

return new InMemoryUserDetailsManager(user);

}

}

**JwtTokenUtil.java**

package com.cognizant.springlearn.config;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Component;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

@Component

public class JwtTokenUtil {

private static final String SECRET\_KEY = "cognizant-secret-key-123"; // Stronger secret key

private static final long EXPIRATION\_TIME = 3600000; // 1 hour

public String generateToken(String username) {

Map<String, Object> claims = new HashMap<>();

return Jwts.builder()

.setClaims(claims)

.setSubject(username)

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

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

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

}

**Pom.XML**

<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-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>

</dependencies>

**OUTPUT:**

