**Spring-REST using Spring Boot 3**

**Hands on 1**

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

🡺solution:

Dependencies

Spring boot dev tools, Spring web

SpringLearnApplication.java

package com.cognizant.spring\_learn;

import org.slf4j.\*;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

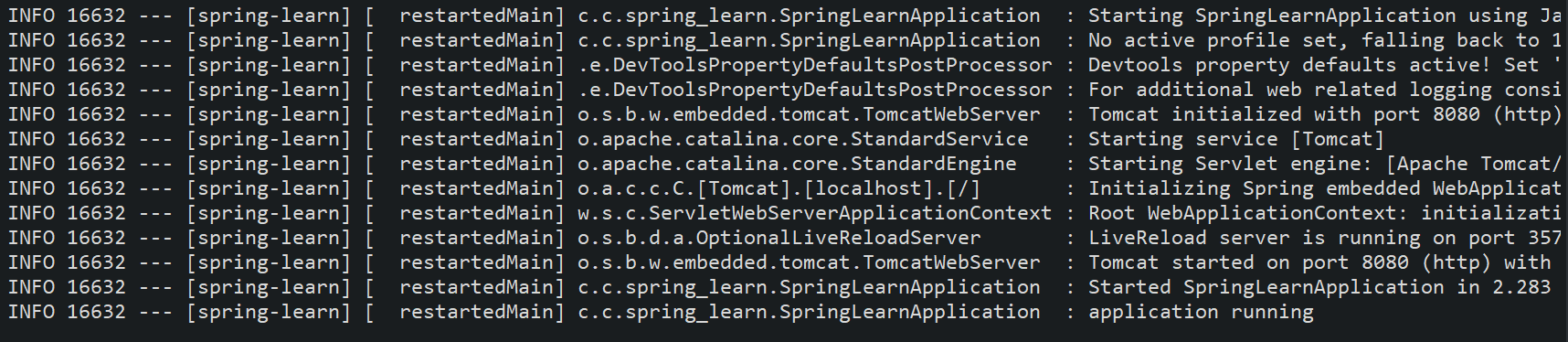
public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

logger.info("application running");

}

}

****

**Hands on 4**

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

🡺Solution:

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>

Country.java

package com.cognizant.spring\_learn;

import org.slf4j.\*;

public class Country {

private String code;

private String name;

private static final Logger logger = LoggerFactory.getLogger(Country.class);

public Country() {

logger.debug("inside country constructor");

}

public String getCode() {

logger.debug("getCode method invoked");

return code;

}

public void setCode(String code) {

logger.debug("setCode method invoked");

this.code = code;

}

public String getName() {

logger.debug("getName method invoked");

return name;

}

public void setName(String name) {

logger.debug("setName method invoked");

this.name = name;

}

@Override

public String toString() {

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

}

}

SpringBootApplication.java

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.slf4j.\*;

@SpringBootApplication

public class SpringLearnApplication{

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

public static void displayCountry() {

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

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

logger.debug(country.toString());

}

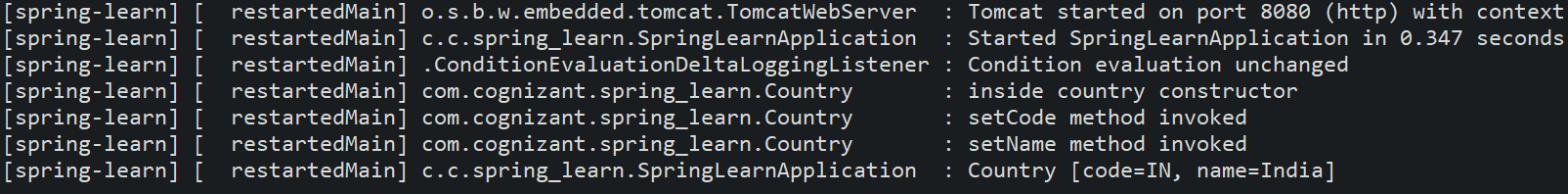
public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args)

displayCountry();

}

}



* **Hello World RESTful Web Service**

🡺solution:

HelloController.java

package com.cognizant.spring\_learn.controller;

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

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

import org.slf4j.\*;

@RestController

public class HelloController {

private static final Logger logger = LoggerFactory.getLogger(HelloController.class);

@GetMapping("/hello")

public String sayHello() {

logger.info("start");

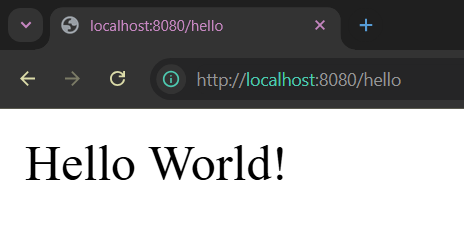
String msg = "Hello World!";

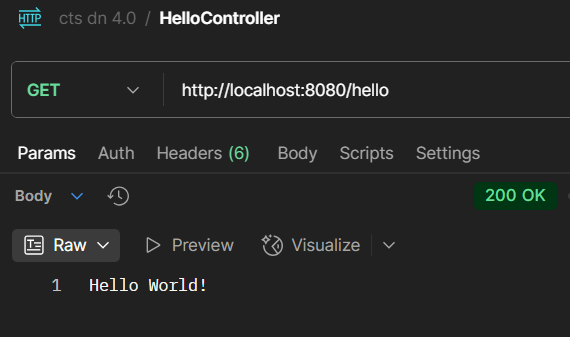
logger.info("end");

return msg;

}

}



****

* **REST - Country Web Service**

🡺solution

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

return country;

}

}

****

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

**🡺solution:**

Country.java

package com.cognizant.spring\_learn;

import org.slf4j.\*;

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;

}

@Override

public String toString() {

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

}

}

CountryController.java

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

import com.cognizant.spring\_learn.Country;

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

@RestController

public class CountryController {

@Autowired

private CountryService countryService;

@GetMapping("/country/{countryCode}")

public Country getCountry(@PathVariable String countryCode) {

return countryService.getCountry(countryCode);

}

}

CountryService.java

package com.cognizant.spring\_learn.service;

import java.util.Arrays;

import java.util.List;

import org.springframework.stereotype.Service;

import com.cognizant.spring\_learn.Country;

@Service

public class CountryService {

public List<Country> getCountries(){

return Arrays.asList(

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

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

);

}

public Country getCountry(String countryCode) {

List<Country> countries = getCountries();

return countries.stream()

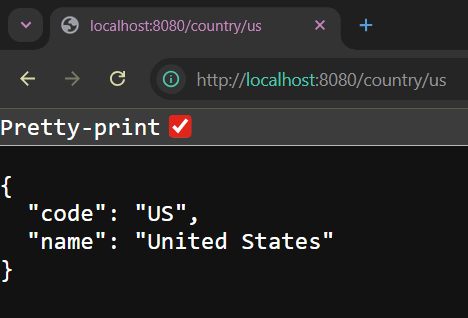
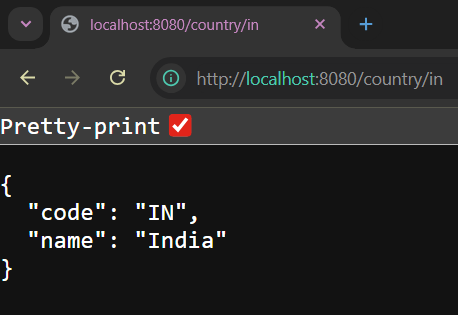
.filter(c-> c.getCode().equalsIgnoreCase(countryCode))

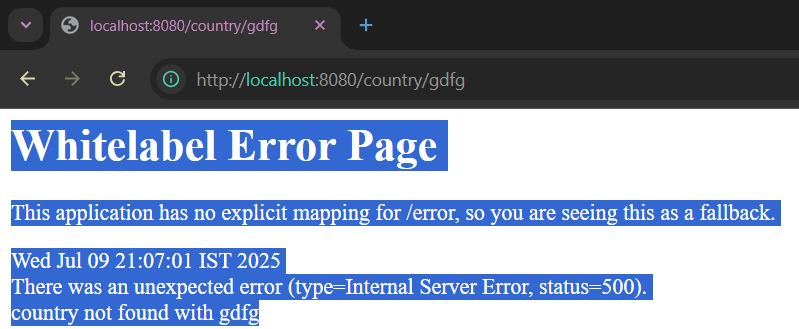
.findFirst()

.orElseThrow(()-> new RuntimeException("country not found with " + countryCode));

}

}



****

------------------------------------------------------------------------------------------------

* **Create authentication service that returns JWT**

🡺solution

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.builders.HttpSecurity;

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

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

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

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

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

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

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

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

.anyRequest().authenticated()

)

.httpBasic(org.springframework.security.config.Customizer.withDefaults());

return http.build();

}

@Bean

public UserDetailsService userDetailsService() {

UserDetails user = User.builder()

.username("user")

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

.roles("USER")

.build();

return new InMemoryUserDetailsManager(user);

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

}

JwtUtil.java

package com.cognizant.spring\_learn.util;

import java.util.Date;

import org.springframework.stereotype.Component;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

@Component

public class JwtUtil {

private static final String SECRET\_KEY = "fdjaknfkje456j3k4rtrteqt45tio5o5o4qtoohrhfgfdsgfg";

public String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

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

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

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

}

AuthenticationController.java

package com.cognizant.spring\_learn.controller;

import java.nio.charset.StandardCharsets;

import java.util.Base64;

import java.util.Map;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

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

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

@RestController

public class AuthenticationController {

@Autowired

private JwtUtil jwtUtil;

@GetMapping("/authenticate")

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

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

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

String credentials = new String(decodedBytes, StandardCharsets.UTF\_8);

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

String username = values[0];

String password = values[1];

if (!"user".equals(username) || !"pwd".equals(password)) {

return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Invalid username or password");

}

String token = jwtUtil.generateToken(username);

return ResponseEntity.ok(Map.of("token", token));

}

}

