WEEK-4

1.Create a Spring Web Project using Maven

Code:

Controller.java:

package com.cognizant.Spring\_learn;

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

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

@RestController

public class Controller {

@GetMapping("/hello")

public String sayHello() {

return "Hello, Spring Boot!";

}

}

SpringLearnApplication.java:

package com.cognizant.Spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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("Inside main");

}

}

SpringLearnApplicationTests.java:

package com.cognizant.Spring\_learn;

import org.junit.jupiter.api.Test;

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

import org.springframework.boot.test.context.SpringBootTest;

import org.springframework.boot.test.web.client.TestRestTemplate;

import org.springframework.http.ResponseEntity;

import static org.assertj.core.api.Assertions.assertThat;

@SpringBootTest(webEnvironment = SpringBootTest.WebEnvironment.RANDOM\_PORT)

class SpringLearnApplicationTests {

@Autowired

private TestRestTemplate restTemplate;

@Test

void helloEndpointReturnsExpectedString() {

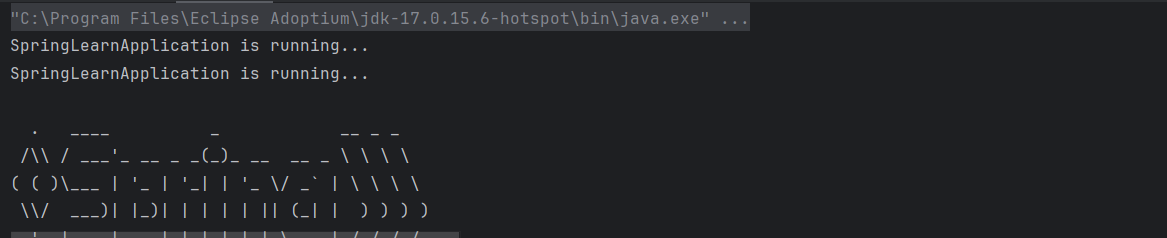
ResponseEntity<String> response = restTemplate.getForEntity("/hello", String.class);

assertThat(response.getBody()).isEqualTo("Hello, Spring Boot!");

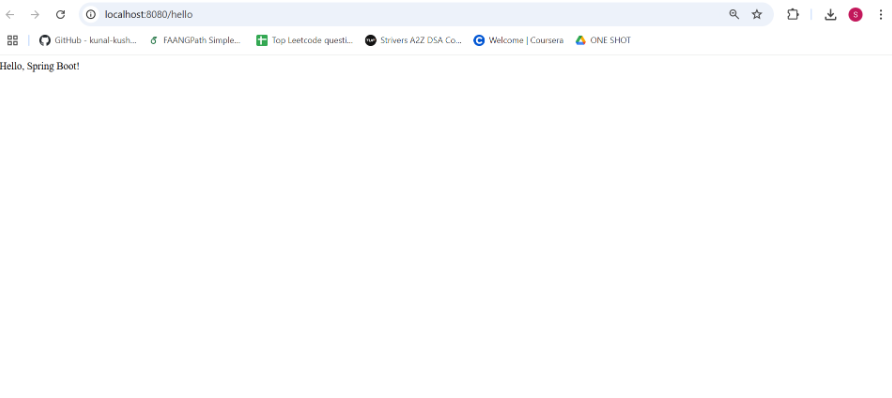
}

}

Output:

****

# Its compiled in localhost:8080/Hello



2. Spring Core – Load Country from Spring Configuration XML

Code:

Country.java:

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 + "]";

}

}

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;

public class SpringLearnApplication {

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

public static void main(String[] args) {

LOGGER.debug("START");

displayCountry();

LOGGER.debug("END");

}

public static void displayCountry() {

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

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

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

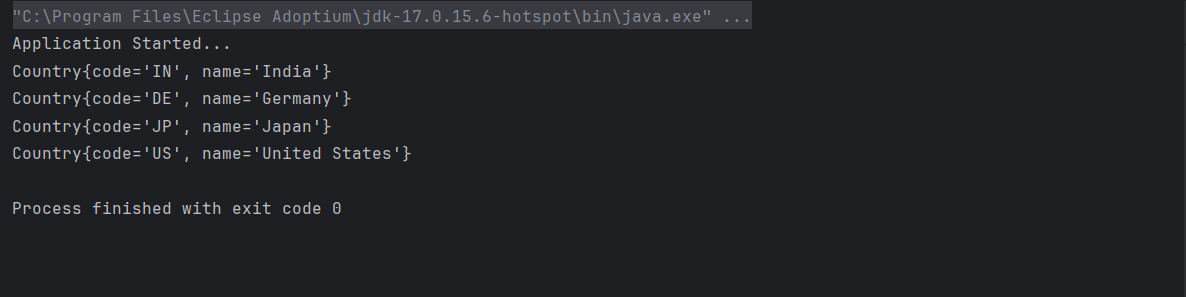
}

}

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="countryIN" class="com.cognizant.springlearn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
  
 <bean id="countryDE" class="com.cognizant.springlearn.Country">  
 <property name="code" value="DE"/>  
 <property name="name" value="Germany"/>  
 </bean>  
  
 <bean id="countryJP" class="com.cognizant.springlearn.Country">  
 <property name="code" value="JP"/>  
 <property name="name" value="Japan"/>  
 </bean>  
  
 <bean id="countryUS" class="com.cognizant.springlearn.Country">  
 <property name="code" value="US"/>  
 <property name="name" value="United States"/>  
 </bean>  
  
</beans>

Output:

****

3. Hello World RESTful Web Service

Code:

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.debug("START");

LOGGER.debug("END");

return " Hello, Spring Boot!";

}

}

SpringLearnApplcation.java:

package com.cognizant.Spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

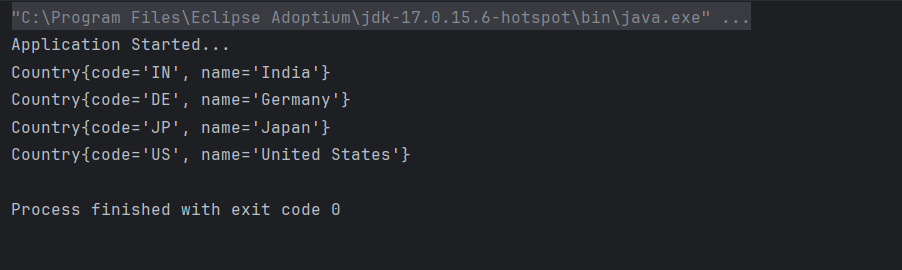
public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

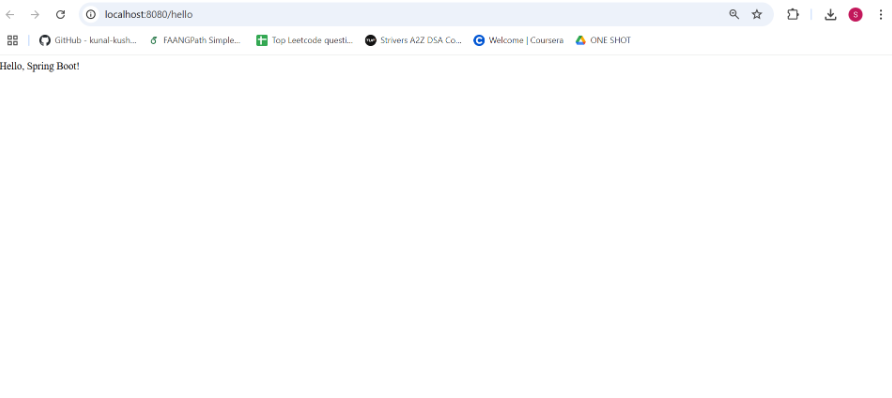
}

}

Output:

****

#which is printed in localhost8080/hello



4. REST - Country Web Service

Code:

Country.java:

package com.cognizant.Spring\_learn.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;

}

}

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.model.Country;

@RestController

public class CountryController {

@RequestMapping("/country")

public Country getCountryIndia() {

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

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

return country;

}

}

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) {

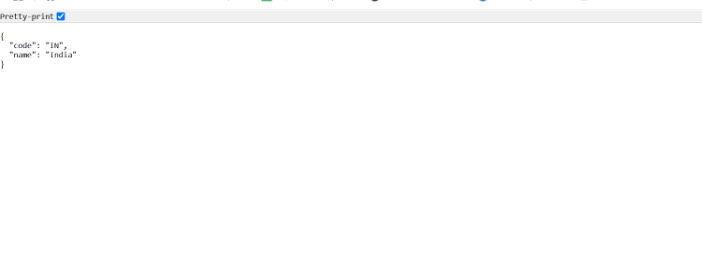
SpringApplication.run(SpringLearnApplication.class, args);

}

}

Output:

#For the convenience and extra code I took only one country code and country.



5. REST - Get country based on country code

Code:

Country.java:

package com.cognizant.Spring\_learn.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;

}

}

CountrySearvice.java:

package com.cognizant.Spring\_learn.service;

import com.cognizant.Spring\_learn.model.Country;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

public Country getCountry(String code) {

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

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

// Using lambda to find case-insensitive match

return countries.stream()

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

.findFirst()

.orElse(null);

}

}

CountryController.java:

package com.cognizant.Spring\_learn.controller;

import com.cognizant.Spring\_learn.model.Country;

import com.cognizant.Spring\_learn.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) {

return countryService.getCountry(code);

}

}

SpringLearnApplication.java:

package com.cognizant.Spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

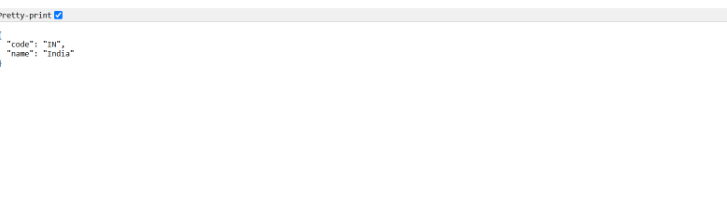
public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

}

Output:



JWT

1. Create authentication service that returns JWT

Code:

Country.java:

package com.cognizant.Spring\_learn.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.service;

import com.cognizant.Spring\_learn.model.Country;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

public Country getCountry(String code) throws Exception {

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

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

return countries.stream()

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

.findFirst()

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

}

}

CountryController.java:

package com.cognizant.Spring\_learn.controller;

import com.cognizant.Spring\_learn.model.Country;

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

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

@RestController

@RequestMapping("/country")

public class CountryController {

private final CountryService countryService;

public CountryController(CountryService countryService) {

this.countryService = countryService;

}

@GetMapping("/{code}")

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

return countryService.getCountry(code);

}

}

SecurityConfig.java:

package com.cognizant.Spring\_learn.security;

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().disable()

.authorizeHttpRequests(auth -> auth.anyRequest().permitAll());

return http.build();

}

}

AuthenticationController.java:

package com.cognizant.Spring\_learn.controller;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import jakarta.servlet.http.HttpServletRequest;

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

import java.util.Base64;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

@RestController

public class AuthenticationController {

private static final String SECRET\_KEY = "secret123";

@GetMapping("/authenticate")

public Map<String, String> authenticate(HttpServletRequest request) {

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

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

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

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

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

if ("user".equals(userDetails[0]) && "pwd".equals(userDetails[1])) {

String jwt = Jwts.builder()

.setSubject(userDetails[0])

.setIssuedAt(new Date())

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

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

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

response.put("token", jwt);

return response;

}

}

throw new RuntimeException("Invalid credentials");

}

@GetMapping("/ping")

public String ping() {

return "Server up!";

}

}

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);

}

}

Output:

