**Name:Amirtha A**

**Hands on 1**

**Create a Spring Web Project using Maven** 

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("the appliction is running...");

SpringApplication.run(SpringLearnApplication.class, args);

}

}

A screenshot of a computer

AI-generated content may be incorrect.

**Hands on 4**

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

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

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

@SpringBootApplication

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

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

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

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

*displayCountry*();

}

**public** **static** **void** displayCountry() {

***LOGGER***.info("START");

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

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

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

***LOGGER***.info("END");

}

}

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("Getter for code invoked");

**return** code;

}

**public** **void** setCode(String code) {

***LOGGER***.debug("Setter for code invoked");

**this**.code = code;

}

**public** String getName() {

***LOGGER***.debug("Getter for name invoked");

**return** name;

}

**public** **void** setName(String name) {

***LOGGER***.debug("Setter for name invoked");

**this**.name = name;

}

@Override

**public** String toString() {

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

}

}

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">

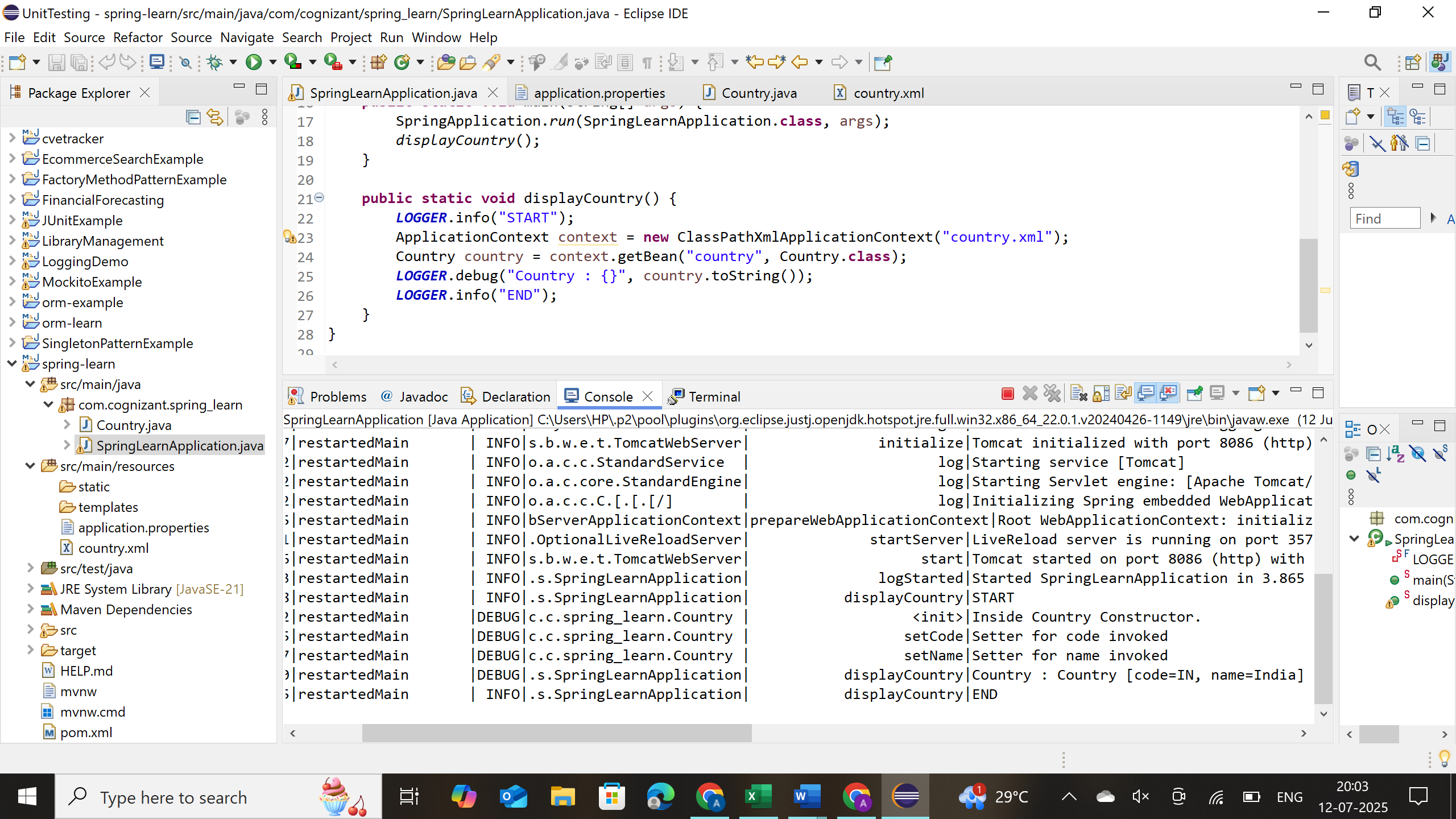
<property name="code" value="IN" />

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

</bean>

</beans>

Output:



**Hello World RESTful Web Service**   
  
Write a REST service in the spring learn application created earlier, that returns the text "Hello World!!" using Spring Web Framework. Refer details below:  
  
**Method:** GET  
**URL:** /hello  
**Controller:** com.cognizant.spring-learn.controller.HelloController  
**Method Signature:** public String sayHello()  
**Method Implementation:** return hard coded string "Hello World!!"  
**Sample Request**: http://localhost:8083/hello  
**Sample Response:** Hello World!!

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***.info("START");

String message = "Hello World!!";

***LOGGER***.debug("Message: {}", message);

***LOGGER***.info("END");

**return** message;

}

}

Output:  
  
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**REST - Country Web Service**   
  
Write a REST service that returns India country details in the earlier created spring learn application.  
  
**URL**: /country  
**Controller**: com.cognizant.spring-learn.controller.CountryController  
**Method Annotation**: @RequestMapping  
**Method Name**: getCountryIndia()  
**Method Implementation**: Load India bean from spring xml configuration and return  
**Sample Request**: http://localhost:8083/country  
**Sample Response**:

{

  "code": "IN",

  "name": "India"

}

**CountryController.java:**  
**package** com.cognizant.spring\_learn.controller;

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

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**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;

@RestController

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

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

@RequestMapping("/country")

**public** Country getCountryIndia() {

***LOGGER***.info("START");

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

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

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

***LOGGER***.info("END");

**return** country;

}

}

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**REST - Get country based on country code**   
  
Write a REST service that returns a specific country based on country code. The country code should be case insensitive.  
  
**Controller**: com.cognizant.spring-learn.controller.CountryController  
**Method Annotation:** @GetMapping("/countries/{code}")  
**Method Name**: getCountry(String code)  
**Method Implemetation**: Invoke countryService.getCountry(code)   
**Service Method:**com.cognizant.spring-learn.service.CountryService.getCountry(String code)

**CountryCountroller.java:**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.Country;

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

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

@RestController

public class CountryController {

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

@Autowired

private CountryService countryService;

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

public Country getCountry(@PathVariable String code) {

*LOGGER*.info("START - getCountry controller");

Country country = countryService.getCountry(code);

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

*LOGGER*.info("END - getCountry controller");

return country;

}

}

**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("Getter for code invoked");

return code;

}

public void setCode(String code) {

LOGGER.debug("Setter for code invoked");

this.code = code;

}

public String getName() {

LOGGER.debug("Getter for name invoked");

return name;

}

public void setName(String name) {

LOGGER.debug("Setter for name invoked");

this.name = name;

}

@Override

public String toString() {

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

}

}

**CountryService.java:**

package com.cognizant.spring\_learn.service;

import com.cognizant.spring\_learn.Country;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CountryService {

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

public Country getCountry(String code) {

*LOGGER*.info("START - getCountry with code: {}", code);

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

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

Country result = countryList.stream()

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

.findFirst()

.orElse(null);

*LOGGER*.debug("Matching country: {}", result);

*LOGGER*.info("END");

return result;

}

}  
**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="countryList" class="java.util.ArrayList">

<constructor-arg>

<list>

<bean class="com.cognizant.spring\_learn.Country">

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

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

</bean>

<bean class="com.cognizant.spring\_learn.Country">

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

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

</bean>

<bean class="com.cognizant.spring\_learn.Country">

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

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

</bean>

<bean class="com.cognizant.spring\_learn.Country">

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

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

</bean>

</list>

</constructor-arg>

</bean>

</beans>

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Create authentication service that returns JWT**   
  
As part of first step of JWT process, the user credentials needs to be sent to authentication service request that generates and returns the JWT.  
  
Ideally when the below curl command is executed that calls the new authentication service, the token should be responded. Kindly note that the credentials are passed using -u option.  
  
**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.web.SecurityFilterChain;

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

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

import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

import org.springframework.security.authentication.dao.DaoAuthenticationProvider;

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

import org.springframework.security.provisioning.InMemoryUser DetailsManager;

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

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeRequests()

.antMatchers("/authenticate").authenticated()

.and()

.httpBasic();

return http.build();

}

@Bean

public UserDetailsService userDetailsService() {

UserDetails user = org.springframework.security.core.userdetails.User.withUsername("user")

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

.roles("USER")

.build();

return new InMemoryUser DetailsManager(user);

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

}  
  
**AuthController.java:**

package com.cognizant.springlearn.controller;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.http.ResponseEntity;

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

import java.util.Date;

@RestController

@RequestMapping("/authenticate")

public class AuthController {

@GetMapping

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

String token = generateToken(authHeader);

return ResponseEntity.ok().body("{\"token\":\"" + token + "\"}");

}

private String generateToken(String authHeader) {

String username = extractUsername(authHeader);

return Jwts.builder()

.setSubject(username)

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + 86400000)) // 1 day expiration

.signWith(SignatureAlgorithm.HS256, "secretkey") // Use a strong secret key

.compact();

}

private String extractUsername(String authHeader) {

String[] parts = authHeader.split(" ");

String credentials = new String(java.util.Base64.getDecoder().decode(parts[1]));

return credentials.split(":")[0]; // Return username

}

}

**SpringlearnApplication.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);

}

}