**Cognizant Deep Nurture 4.0 Hands-on Exercise - Week 4**

EXERCISE 1:   
Create authentication controller and configure it in

SecurityConfig

AuthenticationController.java

Create new rest controller named AuthenticationController in controller package

Include method authenticate with "/authenticate" as the URL with @GetMapping.

To read the Authorization value from HTTP Header, include a parameter for authenticate method as specified below. Spring takes care of reading the Authorization value from HTTP Header and pass it as parameter.

@RequestHeader("Authorization") String authHeader

The return type of this method should be Map<String, String>

Include start and end logger in this method

Include a debug log for displaying the authHeader parameter

Create a new HashMap<String, String> and assign it to a map.

Put a new item into the map with key as "token" and value as empty string.

SecurityConfig.java

In the second configure method, include authenticate URL just after the countries URL defined earlier. Refer code below:

.antMatchers("/countries").hasRole("USER")

.antMatchers("/authenticate").hasAnyRole("USER", "ADMIN")

The above configuration sets that users of both roles can access /authenticate URL.

Testing

curl command:

curl -s -u user:pwd http://localhost:8090/authenticate

Expected Response:

{"token":""}

Log verification:

Check if Authorization header value is displayed with "Basic" prefix and

Base64 encoding of "user:pwd"   
   
   
   
AuthenticationController.java

java

CopyEdit

package com.cognizant.springlearn.controller;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import java.util.HashMap;

import java.util.Map;

@RestController

public class AuthenticationController {

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

@GetMapping("/authenticate")

public Map<String, String> authenticate(@RequestHeader("Authorization") String authHeader) {

LOGGER.info("Start authentication");

LOGGER.debug("Authorization Header: {}", authHeader);

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

map.put("token", "");

LOGGER.info("End authentication");

return map;

}

}

SecurityConfig.java

Ensure your class looks like this:

java

CopyEdit

package com.cognizant.springlearn.security;

import org.springframework.context.annotation.Configuration;

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

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

import org.springframework.security.config.annotation.web.configuration.\*;

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter {

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.inMemoryAuthentication()

.withUser("user").password("{noop}pwd").roles("USER")

.and()

.withUser("admin").password("{noop}pwd").roles("ADMIN");

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeRequests()

.antMatchers("/countries").hasRole("USER")

.antMatchers("/authenticate").hasAnyRole("USER", "ADMIN")

.and()

.httpBasic();

}

}

Testing with curl

Use this command:

bash

CopyEdit

curl -s -u user:pwd http://localhost:8090/authenticate   
   
OUTPUT: 

46e8aaae142a9e2d80516cdf115ad7b9.png