**Project objective**

As a developer, build Authentication Provider in Spring Security.

**Problem statement**

You have been assigned a task by the team to add more flexibility rather than using the standard scenario in building Spring Security.

**Source Code**

User

**package** com.org;

**public** **class** User {

**private** String name;

**private** String password;

**private** String role;

**public** User(String name, String password, String role) {

**super**();

**this**.name = name;

**this**.password = password;

**this**.role = role;

}

**public** String getName() {

**return** name;

}

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

**this**.name = name;

}

**public** String getPassword() {

**return** password;

}

**public** **void** setPassword(String password) {

**this**.password = password;

}

**public** String getRole() {

**return** role;

}

**public** **void** setRole(String role) {

**this**.role = role;

}

}

CustomAuthenticationProvider

package com.org;

import org.springframework.security.authentication.\*;

import org.springframework.security.core.\*;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import java.util.ArrayList;

import java.util.List;

import java.util.Optional;

public class CustomAuthenticationProvider implements AuthenticationProvider {

List<User> dummyUsers = new ArrayList<>();

public CustomAuthenticationProvider() {

dummyUsers.add(new User("john", "secret", "ROLE\_USER"));

dummyUsers.add(new User("admin", "supersecret", "ROLE\_ADMIN"));

}

@Override

public Authentication authenticate(Authentication authentication) throws AuthenticationException {

String name = authentication.getName();

String password = authentication.getCredentials().toString();

Optional<User> authenticatedUser = dummyUsers.stream().filter(

user -> user.getName().equals(name) && user.getPassword().equals(password)

).findFirst();

if(!authenticatedUser.isPresent()){

throw new BadCredentialsException("Some Text");

}

List<GrantedAuthority> authorities = new ArrayList<>();

authorities.add(new SimpleGrantedAuthority(authenticatedUser.get().getRole()));

Authentication auth = new UsernamePasswordAuthenticationToken(name, password, authorities);

return auth;

}

@Override

public boolean supports(Class<?> aClass) {

return aClass.equals(UsernamePasswordAuthenticationToken.class);

}

}

SpringSecurityConfig

package com.org;

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.WebSecurityConfigurerAdapter;

@Configuration

public class SpringSecurityConfig extends WebSecurityConfigurerAdapter {

// Protecting the urls with a role-based access.

@Override

protected void configure(HttpSecurity http) throws Exception {

http.httpBasic().and().authorizeRequests()

.antMatchers("/").permitAll()

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

.antMatchers("/admin").hasRole("ADMIN");

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.authenticationProvider(new CustomAuthenticationProvider());

}

}

MainController

package com.org;

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

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

@RestController

public class MainController {

@RequestMapping("/")

public String hello(){

return "Hello World";

}

@RequestMapping("/protected")

public String protectedHello(){

return "Hello World, i was protected";

}

@RequestMapping("/admin")

public String admin(){

return "Hello World from admin";

}

}

Application

package com.org;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}