**SOURCE CODE**

**Pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.7.3</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.test</groupId>

<artifactId>Security</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>SpringSecurityUserAuthentication</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>11</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**SpringSecurityUserAuthenticationApplication.java**

package com.demo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringSecurityUserAuthenticationApplication {

public static void main(String[] args) {

SpringApplication.run(SpringSecurityUserAuthenticationApplication.class, args);

}

}

**User.java**

package com.demo;

public class User {

private String name;

private String password;

private String role;

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

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;

}

}

**CustomAuthenticator.java**

package com.demo;

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 CustomAuthenticator implements AuthenticationProvider {

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

public CustomAuthenticator() {

dummyUsers.add(new User("Harsha", "123", "ROLE\_USER"));

dummyUsers.add(new User("admin", "1234", "ROLE\_ADMIN"));

}

@Override

public Authentication authenticate(Authentication authentication) throws AuthenticationException {

String name = authentication.getName();

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

//jdk 8 -- stream

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.java**

package com.demo;

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 CustomAuthenticator());

}

}

**MainController.java**

package com.demo;

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";

}

}