**JWT HANDSON**

**pom.xml**

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.0</version>

</dependency>

</dependencies>

**SecurityConfig.java**

package com.cognizant.spring\_learn.security;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.authentication.AuthenticationManager;

import org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfiguration;

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

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

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

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

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

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

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

import org.springframework.security.web.SecurityFilterChain;

@Configuration

public class SecurityConfig {

@Bean

public UserDetailsService userDetailsService(PasswordEncoder encoder) {

UserDetails user = User.withUsername("user").password(encoder.encode("pwd")).roles("USER").build();

UserDetails admin = User.withUsername("admin").password(encoder.encode("pwd")).roles("ADMIN").build();

return new InMemoryUserDetailsManager(user, admin);

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Bean

public AuthenticationManager authenticationManager(AuthenticationConfiguration config) throws Exception {

return config.getAuthenticationManager();

}

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.httpBasic()

.and()

.authorizeHttpRequests()

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

.anyRequest().authenticated();

return http.build();

}

}

**AuthenticationController.java**

package com.cognizant.spring\_learn.controller;

import java.util.\*;

import java.util.Base64;

import java.util.Date;

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

import io.jsonwebtoken.\*;

@RestController

public class AuthenticationController {

@GetMapping("/authenticate")

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

String user = getUser(authHeader);

String token = generateJwt(user);

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

map.put("token", token);

return map;

}

private String getUser(String authHeader) {

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

byte[] decodedBytes = Base64.getDecoder().decode(encoded);

String decoded = new String(decodedBytes);

return decoded.split(":")[0];

}

private String generateJwt(String user) {

JwtBuilder builder = Jwts.builder();

builder.setSubject(user);

builder.setIssuedAt(new Date());

builder.setExpiration(new Date(System.currentTimeMillis() + 1200000));

builder.signWith(SignatureAlgorithm.HS256, "secretkey");

return builder.compact();

}

}

**Postman Test**

**Method:** GET  
**URL:** http://localhost:8080/authenticate  
**Auth tab:** Basic Auth  
**Username:** user  
**Password:** pwd

**OUTPUT:**



