**Create authentication service that returns JWT   
AuthenticationController.java**

package com.example.jwt\_auth\_service.controller;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import io.jsonwebtoken.JwtBuilder;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import io.jsonwebtoken.security.Keys;

import java.security.Key;

import java.util.\*;

@RestController

public class AuthenticationController {

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

@GetMapping("/authenticate")

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

*log*.info("Start authenticate");

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

String user = getUser(authHeader);

*log*.debug("User extracted: {}", user);

String token = generateJwt(user);

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

map.put("token", token);

*log*.info("End authenticate");

return map;

}

private String getUser(String authHeader) {

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

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

String decodedCredentials = new String(decodedBytes);

*log*.debug("Decoded credentials: {}", decodedCredentials);

String user = decodedCredentials.split(":")[0];

return user;

}

private String generateJwt(String user) {

Key key = Keys.*hmacShaKeyFor*("secretkeysecretkeysecretkeysecretkey".getBytes());

String token = Jwts.*builder*()

.setSubject(user)

.setIssuedAt(new Date())

.setExpiration(new Date(System.*currentTimeMillis*() + 20 \* 60 \* 1000))

.signWith(key)

.compact();

return token;

}

}

**Securityconfig.java**

package com.example.jwt\_auth\_service.security;

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.core.userdetails.\*;

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

import org.springframework.security.web.SecurityFilterChain;

@Configuration

public class SecurityConfig {

@Bean

public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeHttpRequests()

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

.anyRequest().authenticated()

.and()

.httpBasic();

return http.build();

}

@Bean

public UserDetailsService userDetailsService() {

UserDetails user = User

.*withUsername*("user")

.password("{noop}pwd") // {noop} is for plain text passwords

.roles("USER")

.build();

return new InMemoryUserDetailsManager(user);

}

}

**Output**:

Token is generated

