5. JWT-HANDSON

1) Create authentication service that returns JWT

Code:

AuthController.java:

**package** com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.model.AuthResponse;

import com.cognizant.spring\_learn.util.JwtUtil;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpHeaders;

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

import java.nio.charset.StandardCharsets;

import java.util.Base64;

@RestController

public **class** AuthController {

@Autowired

private JwtUtil jwtUtil;

@GetMapping("/authenticate")

public AuthResponse authenticate(@RequestHeader(HttpHeaders.AUTHORIZATION) String authHeader) {

String base64Credentials = authHeader.substring("Basic ".length());

byte[] credDecoded = Base64.getDecoder().decode(base64Credentials);

String credentials = new String(credDecoded, StandardCharsets.UTF\_8);

String[] values = credentials.split(":", 2);

String username = values[0];

String password = values[1];

// Hardcoded check for simplicity

if ("user".equals(username) && "pwd".equals(password)) {

String token = jwtUtil.generateToken(username);

return new AuthResponse(token);

} **else** {

throw new RuntimeException("Invalid credentials");

}

}

}

SecurityConfig.java:

**package** com.cognizant.spring\_learn.config;

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.web.SecurityFilterChain;

@Configuration

public **class** SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeHttpRequests()

.anyRequest().permitAll();

return http.build();

}

}

Output: