**Create authentication service that returns JWT**

**AuthenticationController.java**

package com.cognizant.springlearn.controller;

import java.util.Base64;

import javax.servlet.http.HttpServletRequest;

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

import org.springframework.security.authentication.AuthenticationManager;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.AuthenticationException;

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

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

import com.cognizant.springlearn.util.JwtUtil;

@RestController

public class AuthenticationController {

@Autowired

private AuthenticationManager authenticationManager;

@Autowired

private JwtUtil jwtUtil;

@GetMapping("/authenticate")

public String generateToken(HttpServletRequest request) throws Exception {

String authHeader = request.getHeader("Authorization");

if (authHeader == null || !authHeader.startsWith("Basic ")) {

throw new Exception("Missing or invalid Authorization header");

}

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

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

String credentials = new String(decodedBytes, "UTF-8");

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

String username = values[0];

String password = values[1];

try {

authenticationManager.authenticate(new UsernamePasswordAuthenticationToken(username, password));

} catch (AuthenticationException e) {

throw new Exception("Invalid username/password");

}

String token = jwtUtil.generateToken(username);

return "{\"token\":\"" + token + "\"}";

}

}

**JwtUtil.java**

package com.cognizant.springlearn.util;

import java.util.Date;

import org.springframework.stereotype.Component;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

@Component

public class JwtUtil {

private String secret = "secret"; // You can change this to any secret

public String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

.setIssuedAt(new Date(System.currentTimeMillis()))

.setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 10)) // 10 minutes

.signWith(SignatureAlgorithm.HS256, secret)

.compact();

}

}

**SecurityConfig.java**

package com.cognizant.springlearn.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.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.web.SecurityFilterChain;

*@Configuration*

*@EnableWebSecurity*

public class SecurityConfig {

*@Bean*

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

.csrf().disable()

.authorizeRequests()

.antMatchers("/authenticate").permitAll() // allow /authenticate without login

.anyRequest().authenticated()

.and()

.httpBasic(); // enable basic auth

return http.build();

}

}

**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.**