**6. Create authentication service that returns JWT:**

**CODE:**

**SpringjwtAuthApplication.java:**

package com.example.jwt;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
@SpringBootApplication  
public class SpringJwtAuthApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringJwtAuthApplication.class, args);  
 }  
}

**SecurityConfig.java:**

package com.example.jwt.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  
 .securityMatcher("/\*\*") // Optional matcher for all routes  
 .authorizeHttpRequests(authorize -> authorize  
 .requestMatchers("/authenticate").permitAll()  
 .anyRequest().authenticated()  
 )  
 .formLogin(login -> login.disable()) // Disable form login  
 .httpBasic(basic -> basic.disable()) // Optional: disable basic auth  
 .csrf(csrf -> csrf.disable()); // New lambda style for disabling CSRF  
  
 return http.build();  
 }  
}

**Jwtutil.java:**

package com.example.jwt.util;  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
import java.util.Date;  
public class JwtUtil {  
 private static final String *SECRET\_KEY* = "secret123";  
 public static String generateToken(String username) {  
 return Jwts.*builder*()  
 .setSubject(username)  
 .setIssuedAt(new Date())  
 .setExpiration(new Date(System.*currentTimeMillis*() + 10 \* 60 \* 1000)) // 10 mins  
 .signWith(SignatureAlgorithm.*HS256*, *SECRET\_KEY*)  
 .compact();  
 }  
}

**AuthController.java:**

package com.example.cognizant.springlearn.controller;  
import com.example.jwt.util.JwtUtil;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
import javax.servlet.http.HttpServletRequest;  
import java.util.Base64;  
@RestController  
public class AuthController {  
  
 @GetMapping("/authenticate")  
 public ResponseEntity<?> authenticate(HttpServletRequest request) {  
 String authHeader = request.getHeader("Authorization");  
 if (authHeader == null || !authHeader.startsWith("Basic ")) {  
 return ResponseEntity.*status*(401).body("Missing or invalid Authorization header");  
 }  
  
 String base64Credentials = authHeader.substring("Basic ".length());  
 byte[] decoded = Base64.*getDecoder*().decode(base64Credentials);  
 String[] credentials = new String(decoded).split(":", 2);  
 String username = credentials[0];  
 String password = credentials[1];  
 if ("user".equals(username) && "pwd".equals(password)) {  
 String token = JwtUtil.*generateToken*(username);  
 return ResponseEntity.*ok*().body("{\"token\":\"" + token + "\"}");  
 } else {  
 return ResponseEntity.*status*(403).body("Invalid credentials");  
 }  
 }  
}

**application.properties:**

server.port=9000

**OUTPUT:**

