## **Create Authentication Service that Returns JWT**

### **Objective:**

Create a REST endpoint that authenticates user credentials using **Basic Auth** and returns a **JWT token** as a response.

### **1. Add JWT Dependencies to pom.xml**

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.1</version>

</dependency>

### **2. Create JWT Utility Class**

package com.cognizant.springlearn.util;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import java.util.Date;

public class JwtUtil {

private static final String SECRET\_KEY = "secretkey"; // Use env variable in real apps

private static final long EXPIRATION\_TIME = 1000 \* 60 \* 10; // 10 minutes

public static String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

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

.setExpiration(new Date(System.currentTimeMillis() + EXPIRATION\_TIME))

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

}

### **3. Create Authentication Controller**

package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.util.JwtUtil;

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

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

import javax.servlet.http.HttpServletRequest;

import java.util.Base64;

import java.util.HashMap;

import java.util.Map;

@RestController

public class AuthenticationController {

@GetMapping("/authenticate")

public Map<String, String> authenticate(HttpServletRequest request) {

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

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

throw new RuntimeException("Missing or invalid Authorization header.");

}

// Decode Base64 credentials

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

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

String credentials = new String(decodedBytes);

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

String username = parts[0];

String password = parts[1];

// Dummy check (replace with actual user check)

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

String token = JwtUtil.generateToken(username);

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

response.put("token", token);

return response;

} else {

throw new RuntimeException("Invalid username or password");

}

}

}

### **4. Configure Spring Security**

package com.cognizant.springlearn.config;

import org.springframework.context.annotation.Configuration;

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

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

@Configuration

public class SecurityConfig extends WebSecurityConfigurerAdapter {

@Override

protected void configure(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeRequests().anyRequest().permitAll();

}

}

### **5 . Output :**

