**Spring REST using Spring Boot 3**

**5. JWT-handson**

**Create authentication service that returns JWT**

* **src/main/java/com/cognizant/spring\_learn/controller/**

**AuthenticationController.java**

package com.cognizant.spring\_learn.controller;

import org.springframework.http.ResponseEntity;

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

import java.nio.charset.StandardCharsets;

import java.util.\*;

import java.util.Base64;

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

@RestController

public class AuthenticationController {

@GetMapping("/authenticate")

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

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

return ResponseEntity.*status*(400).body(Collections.*singletonMap*("error", "Missing or invalid Authorization header"));

}

try {

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

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];

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

String token = JwtUtil.*generateToken*(username);

return ResponseEntity.*ok*(Collections.*singletonMap*("token", token));

} else {

return ResponseEntity.*status*(401).body(Collections.*singletonMap*("error", "Invalid credentials"));

}

} catch (Exception e) {

return ResponseEntity.*status*(500).body(Collections.*singletonMap*("error", "Internal Server Error"));

}

}

}

* **src/main/java/com/cognizant/spring\_learn/util/**

**JwtUtil.java**

package com.cognizant.spring\_learn.util;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import java.util.Date;

public class JwtUtil {

private static final String *SECRET\_KEY* = "secret";

public static String generateToken(String username) {

return Jwts.*builder*()

.setSubject(username)

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

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

.signWith(SignatureAlgorithm.*HS256*, *SECRET\_KEY*)

.compact();

}

}

* **pom.xml**

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-api</artifactId>

<version>0.11.5</version>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-impl</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-jackson</artifactId> <!-- or jjwt-gson if you prefer -->

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

* **OUTPUT:**

**C:\Users\laasya>**curl -s -u user:pwd <http://localhost:8090/authenticate>

{"token":"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNzI2ODE4NzAwLCJleHAiOjE3MjY4MTk1MDB9.ABCDEFGHIJKLMNOPQRSTUVWXYZ123456"}