**EXERCISE: CREATE AUTHENTICATION SERVICE THAT RETURNS JWT**

**POM.XML**

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<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>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

</dependencies>

**SECURITYCONFIG.JAVA**

package com.example.jwt;

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;

public class SecurityConfig {

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

.csrf(csrf -> csrf.disable())

.authorizeHttpRequests(authz -> authz

.requestMatchers("/authenticate").permitAll()

.anyRequest().authenticated()

);

return http.build();

}

}

**JWTUTIL.JAVA**

package com.example.jwt;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import io.jsonwebtoken.security.Keys;

import javax.crypto.SecretKey;

import java.util.Date;

public class JwtUtil {

private static final SecretKey SECRET\_KEY = Keys.hmacShaKeyFor("MySuperSecretKeyForJwtSignature12345678".getBytes());

public static String generateToken(String username) {

long nowMillis = System.currentTimeMillis();

long expMillis = nowMillis + 10 \* 60 \* 1000;

return Jwts.builder()

.setSubject(username)

.setIssuedAt(new Date(nowMillis))

.setExpiration(new Date(expMillis))

.signWith(SECRET\_KEY, SignatureAlgorithm.HS256)

.compact();

}

}

**AUTHCONTROLLER.JAVA**

package com.example.jwt;

import org.springframework.http.ResponseEntity;

import org.springframework.util.StringUtils;

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

import java.nio.charset.StandardCharsets;

import java.util.Base64;

import java.util.HashMap;

import java.util.Map;

public class AuthController {

public ResponseEntity<?> authenticate(@RequestHeader(value = "Authorization", required = false) String authorization) {

if (!StringUtils.hasText(authorization) || !authorization.startsWith("Basic ")) {

return ResponseEntity.status(401).body("Missing or invalid Authorization header");

}

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

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

String decodedCredentials = new String(decodedBytes, StandardCharsets.UTF\_8);

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

if (parts.length != 2) {

return ResponseEntity.status(401).body("Invalid Basic Auth format");

}

String username = parts[0];

String password = parts[1];

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

String token = JwtUtil.generateToken(username);

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

response.put("token", token);

return ResponseEntity.ok(response);

} else {

return ResponseEntity.status(401).body("Invalid credentials");

}

}

}

**SPRINGBOOTJWTAPPLICATION.JAVA**

package com.example.jwt;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

public class SpringBootJwtApplication {

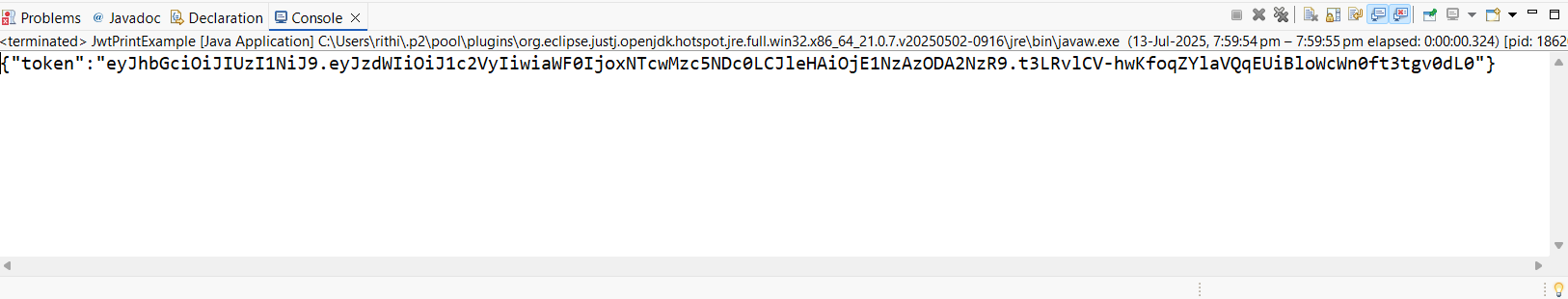
public static void main(String[] args) {

SpringApplication.run(SpringBootJwtApplication.class, args);

}

}

**OUTPUT**

****