**Create authentication service that returns JWT**

**SCENARIO:**

As part of first step of JWT process, the user credentials needs to be sent to authentication service request that generates and returns the JWT.

**CODE:**

**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.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

import org.springframework.security.web.SecurityFilterChain;

*@Configuration*

public class SecurityConfig {

*@Bean*

public UserDetailsService userDetailsService() {

InMemoryUserDetailsManager manager = new InMemoryUserDetailsManager();

manager.createUser(User.*withUsername*("user").password("{noop}pwd").roles("USER").build());

return manager;

}

*@Bean*

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeHttpRequests()

.anyRequest().authenticated()

.and()

.httpBasic();

return http.build();

}

}

**AuthenticationController.java:**

package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.util.JwtUtil;

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

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

import java.security.Principal;

import java.util.HashMap;

import java.util.Map;

*@RestController*

public class AuthenticationController {

*@Autowired*

private JwtUtil jwtUtil;

*@GetMapping*("/authenticate")

public Map<String, String> authenticate(Principal principal) {

String token = jwtUtil.generateToken(principal.getName());

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

response.put("token", token);

return response;

}

}

**JwtUtil.java:**

package com.cognizant.springlearn.util;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import io.jsonwebtoken.security.Keys;

import org.springframework.stereotype.Component;

import java.security.Key;

import java.util.Date;

*@Component*

public class JwtUtil {

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

private static final String ***SECRET*** = "mysecretkey1234567890mysecretkey1234567890"; // must be at least 256 bits (32 chars)

private final Key key = Keys.*hmacShaKeyFor*(***SECRET***.getBytes());

public String generateToken(String username) {

return Jwts.*builder*()

.setSubject(username)

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

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

.signWith(key, *SignatureAlgorithm*.***HS256***)

.compact();

}

}

**Pom.xml:**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

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

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

<version>3.1.3</version>

<relativePath/>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>jwt-auth-service</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>jwt-auth-service</name>

<description>JWT Auth Service</description>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<!-- Web support -->

<dependency>

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

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

</dependency>

<!-- Spring Security -->

<dependency>

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

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

</dependency>

<!-- JSON Web Token (JJWT 0.11.5 Recommended for Java 17+) -->

<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> <!-- Use jjwt-gson if preferred -->

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<!-- Dev Tools (Hot reload) -->

<dependency>

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

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<!-- Testing -->

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**OUTPUT:**



