**Create authentication service that returns JWT**

**SecurityConfig**

package com.cognizant.authservice.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.Customizer;

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

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

import org.springframework.security.web.SecurityFilterChain;

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {

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

.authorizeHttpRequests(auth -> auth

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

.anyRequest().authenticated()

)

.httpBasic(Customizer.withDefaults());

return http.build();

}

}

**AuthenticationController.java**

package com.cognizant.authservice.controller;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import jakarta.servlet.http.HttpServletRequest;

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

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

import javax.crypto.spec.SecretKeySpec;

import java.security.Key;

import java.util.Base64;

import java.util.Date;

@RestController

public class AuthenticationController {

private static final String SECRET\_KEY = "myVerySecretKeyThatIsLongEnough123!";

@GetMapping("/authenticate")

public String authenticate(HttpServletRequest request) {

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

System.out.println("Authorization Header: " + authHeader); // Debug print

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

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

}

try {

// Decode Base64 credentials

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

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

String credentials = new String(credDecoded);

System.out.println("Decoded credentials: " + credentials); // Debug print

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

if (values.length != 2) {

throw new RuntimeException("Invalid credentials format");

}

String username = values[0];

String password = values[1];

// Validate credentials (hardcoded)

if (!username.equals("user") || !password.equals("pwd")) {

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

}

// Generate signing key

Key key = new SecretKeySpec(SECRET\_KEY.getBytes(), SignatureAlgorithm.HS256.getJcaName());

String token = Jwts.builder()

.setSubject(username)

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + 3600 \* 1000)) // 1 hour

.signWith(key)

.compact();

return "{\"token\":\"" + token + "\"}";

} catch (Exception e) {

e.printStackTrace();

throw new RuntimeException("Authentication failed");

}

}

}

**AuthServiceApplication**

package com.cognizant.authservice;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class AuthServiceApplication {

public static void main(String[] args) {

SpringApplication.run(AuthServiceApplication.class, args);

}

}

**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>3.5.3</version>

<relativePath/>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>auth-service</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>auth-service</name>

<description>auth-service</description>

<properties>

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

</properties>

<dependencies>

<!-- Spring Boot -->

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<!-- JWT dependencies (0.11.5) -->

<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> <!-- for JSON serialization -->

<version>0.11.5</version>

<scope>runtime</scope>

</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>

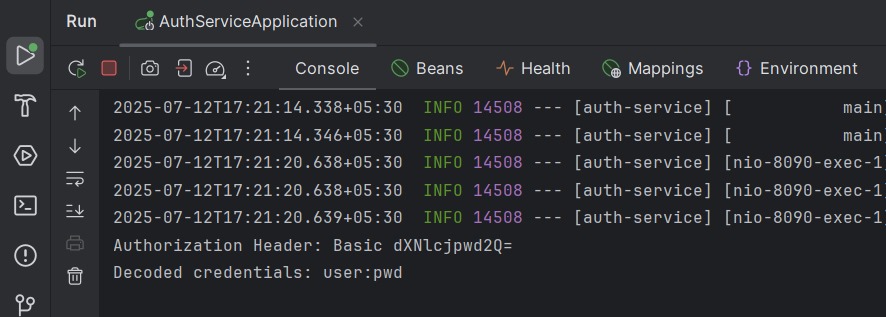
</plugin>

</plugins>

</build>

</project>

**OUTPUT:**

****

