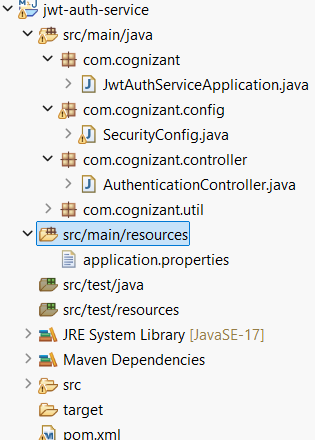
**Create authentication service that returns JWT**

****

**JwtAuthServiceApplication.java**

package com.cognizant;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class JwtAuthServiceApplication {

public static void main(String[] args) {

SpringApplication.*run*(JwtAuthServiceApplication.class, args);

}

}

**SecurityConfig.java**

**package** com.cognizant.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.UserDetails;

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

**import** org.springframework.security.provisioning.InMemoryUserDetailsManager;

**import** org.springframework.security.crypto.password.NoOpPasswordEncoder;

**import** org.springframework.security.web.SecurityFilterChain;

@Configuration

**public** **class** SecurityConfig {

// Define an in-memory user

@Bean

**public** UserDetailsService userDetailsService() {

UserDetails user = User

.*withUsername*("user")

.password("pwd")

.roles("USER")

.build();

**return** **new** InMemoryUserDetailsManager(user);

}

// Configure HTTP security rules

@Bean

**public** SecurityFilterChain securityFilterChain(HttpSecurity http) **throws** Exception {

http.csrf().disable()

.authorizeHttpRequests()

.requestMatchers("/authenticate").authenticated()

.anyRequest().permitAll()

.and()

.httpBasic(); // Enable HTTP Basic Auth

**return** http.build();

}

// For demo/testing only (no password encryption)

@SuppressWarnings("deprecation")

@Bean

**public** **static** org.springframework.security.crypto.password.PasswordEncoder passwordEncoder() {

**return** NoOpPasswordEncoder.getInstance();

}

}

**AuthenticationController.java**

package com.cognizant.controller;

import com.cognizant.util.JwtUtil;

import org.springframework.http.ResponseEntity;

import org.springframework.security.core.Authentication;

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

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

@RestController

public class AuthenticationController {

private final JwtUtil jwtUtil;

public AuthenticationController(JwtUtil jwtUtil) {

this.jwtUtil = jwtUtil;

}

@GetMapping("/authenticate")

public ResponseEntity<?> generateToken(Authentication authentication) {

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

return ResponseEntity.*ok*().body("{\"token\":\"" + token + "\"}");

}

}

**JwtUtil.java**

package com.cognizant.util;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Component;

import java.util.Date;

@Component

public class JwtUtil {

private final String SECRET\_KEY = "mySecretKey";

public String generateToken(String username) {

return Jwts.*builder*()

.setSubject(username)

.setIssuedAt(new Date())

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

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

.compact();

}

}

**Pom.xml**

<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

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

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

<version>1.0.0</version>

<packaging>jar</packaging>

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

<description>Spring Boot project for JWT-based authentication</description>

<parent>

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

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

<version>3.1.1</version>

<relativePath/>

</parent>

<properties>

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

</properties>

<dependencies>

<!-- Spring Boot Web Starter -->

<dependency>

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

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

</dependency>

<!-- Spring Boot Security -->

<dependency>

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

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

</dependency>

<!-- JWT (JSON Web Token) -->

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.1</version>

</dependency>

<!-- Optional: Spring Boot DevTools (for hot reload) -->

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<!-- Optional: Test support -->

<dependency>

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

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

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Spring Boot Maven Plugin -->

<plugin>

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

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

<version>3.1.1</version>

</plugin>

</plugins>

</build>

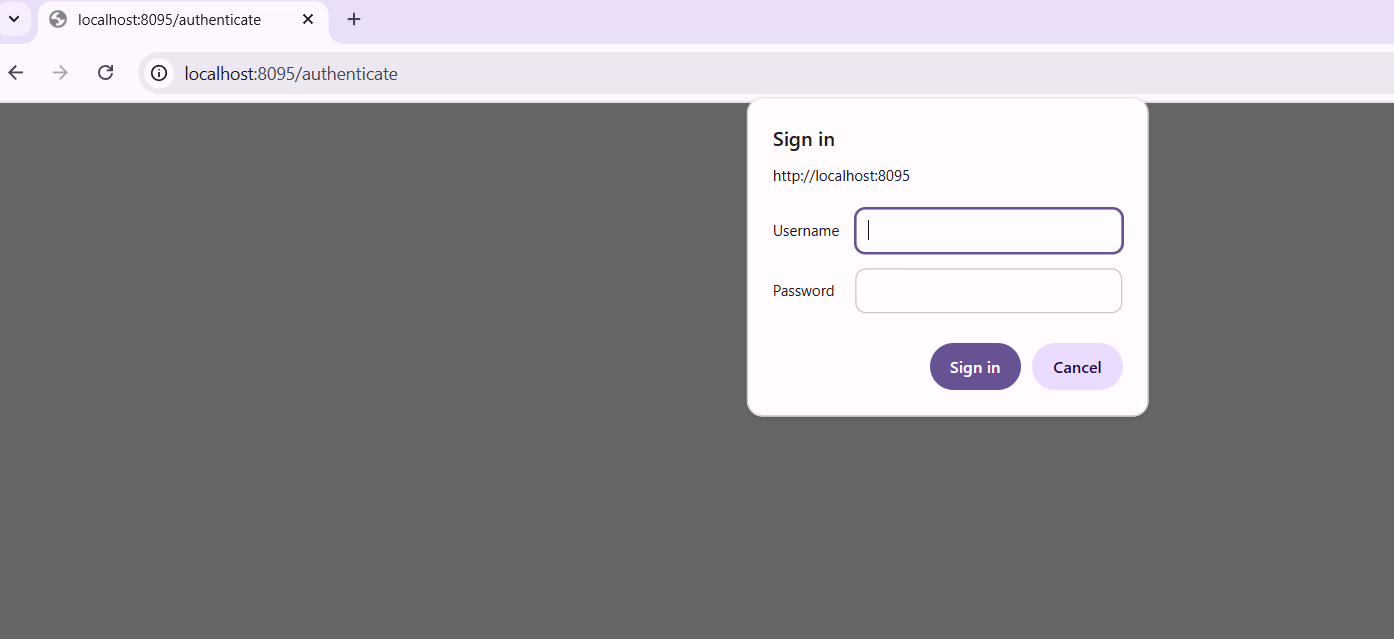
</project>

**Application.properties**

server.port=8095

**OUTPUT:**

****

****