**Create authentication service that returns JWT**

**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

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

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant.spring\_learn</groupId>

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

<version>0.0.1-SNAPSHOT</version>

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

<description>Spring Boot JWT Authentication Service</description>

<packaging>jar</packaging>

<parent>

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

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

<version>3.2.4</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<java.version>21</java.version>

</properties>

<dependencies>

<!-- Spring Web -->

<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>

<!-- JWT Dependencies -->

<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 support -->

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<!-- Optional: DevTools for live reload -->

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<!-- Optional: Testing (not used for this task, but included) -->

<dependency>

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

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

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Maven plugin for Spring Boot -->

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**JwtUtil.java:**

package com.cognizant.spring\_learn.jwt\_auth\_service.util;

import io.jsonwebtoken.\*;

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 Key *key* = Keys.*secretKeyFor*(SignatureAlgorithm.*HS256*);

public String generateToken(String username) {

return Jwts.*builder*()

.setSubject(username)

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

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

.signWith(*key*)

.compact();

}

}

**AuthenticationController.java:**

package com.cognizant.spring\_learn.jwt\_auth\_service.controller;

import com.cognizant.spring\_learn.jwt\_auth\_service.util.JwtUtil;

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

import org.springframework.http.ResponseEntity;

import org.springframework.security.core.annotation.AuthenticationPrincipal;

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

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

@RestController

public class AuthenticationController {

@Autowired

private JwtUtil jwtUtil;

@GetMapping("/authenticate")

public ResponseEntity<?> authenticate(@AuthenticationPrincipal UserDetails userDetails) {

if (userDetails == null) {

return ResponseEntity.*status*(401).body("User not authenticated");

}

String token = jwtUtil.generateToken(userDetails.getUsername());

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

}

}

**SecurityConfig.java:**

package com.cognizant.spring\_learn.jwt\_auth\_service.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.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() {

return new InMemoryUserDetailsManager(

User.*withUsername*("user")

.password("{noop}pwd") // {noop} means no password encoding

.roles("USER")

.build()

);

}

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

.authorizeHttpRequests(auth -> auth

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

.anyRequest().authenticated()

)

.httpBasic(Customizer.*withDefaults*())

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

return http.build();

}

}

**JwtAuthServiceApplication.java:**

package com.cognizant.spring\_learn.jwt\_auth\_service;

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);

}

}

**application.properties:**

spring.application.name=jwt-auth-service

server.port=8090

**Output:**





