**JAVA FSE WEEK 4**

**JWT**

**Mandatory Hands-on Excercises**

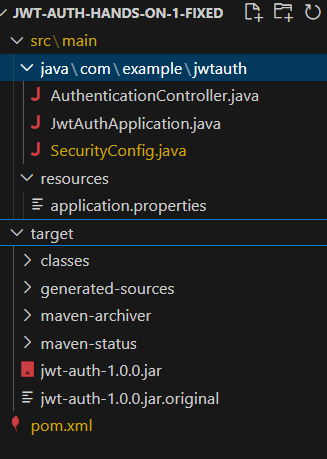
**Create Authentication Service that Returns JWT**

**Step 1**

**Objective:**

Implement a Spring Boot authentication endpoint /authenticate that accepts credentials via Basic Auth and returns a dummy JWT token.

**1.Project Structure:**

****

**2. pom.xml**

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt</artifactId>

<version>0.9.1</version>

</dependency>

<dependency>

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

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

</dependency>

</dependencies>

**3.application.properties:**

server.port=8090

**4.JwtAuthApplication.java:**

package com.example.jwtauth;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class JwtAuthApplication {

public static void main(String[] args) {

SpringApplication.run(JwtAuthApplication.class, args);

}

}

**5.AuthenticationController.java:**

package com.example.jwtauth;

import org.springframework.http.HttpHeaders;

import org.springframework.http.ResponseEntity;

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

import java.util.Collections;

@RestController

public class AuthenticationController {

@GetMapping("/authenticate")

public ResponseEntity<?> authenticate(@RequestHeader(HttpHeaders.AUTHORIZATION) String authHeader) {

return ResponseEntity.ok(Collections.singletonMap("token", "dummy-token"));

}

}

**6.SecurityConfig.java**

package com.example.jwtauth;

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.config.annotation.web.configuration.EnableWebSecurity;

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.web.SecurityFilterChain;

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

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

.authorizeHttpRequests(auth -> auth

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

.anyRequest().authenticated()

)

.httpBasic(httpBasic -> {});

return http.build();

}

@Bean

public UserDetailsService userDetailsService() {

UserDetails user = User

.withUsername("user")

.password("{noop}pwd") // No password encoding for now

.roles("USER")

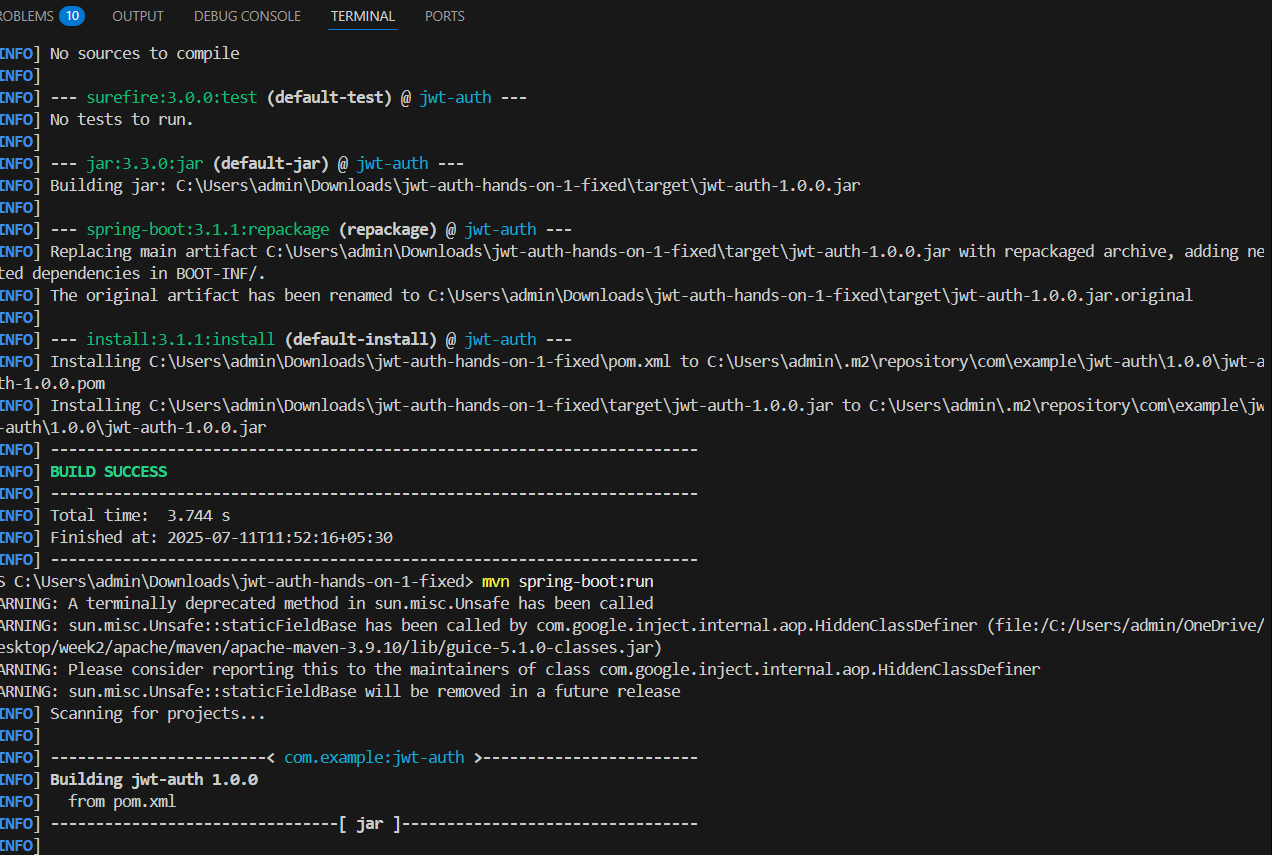
.build();

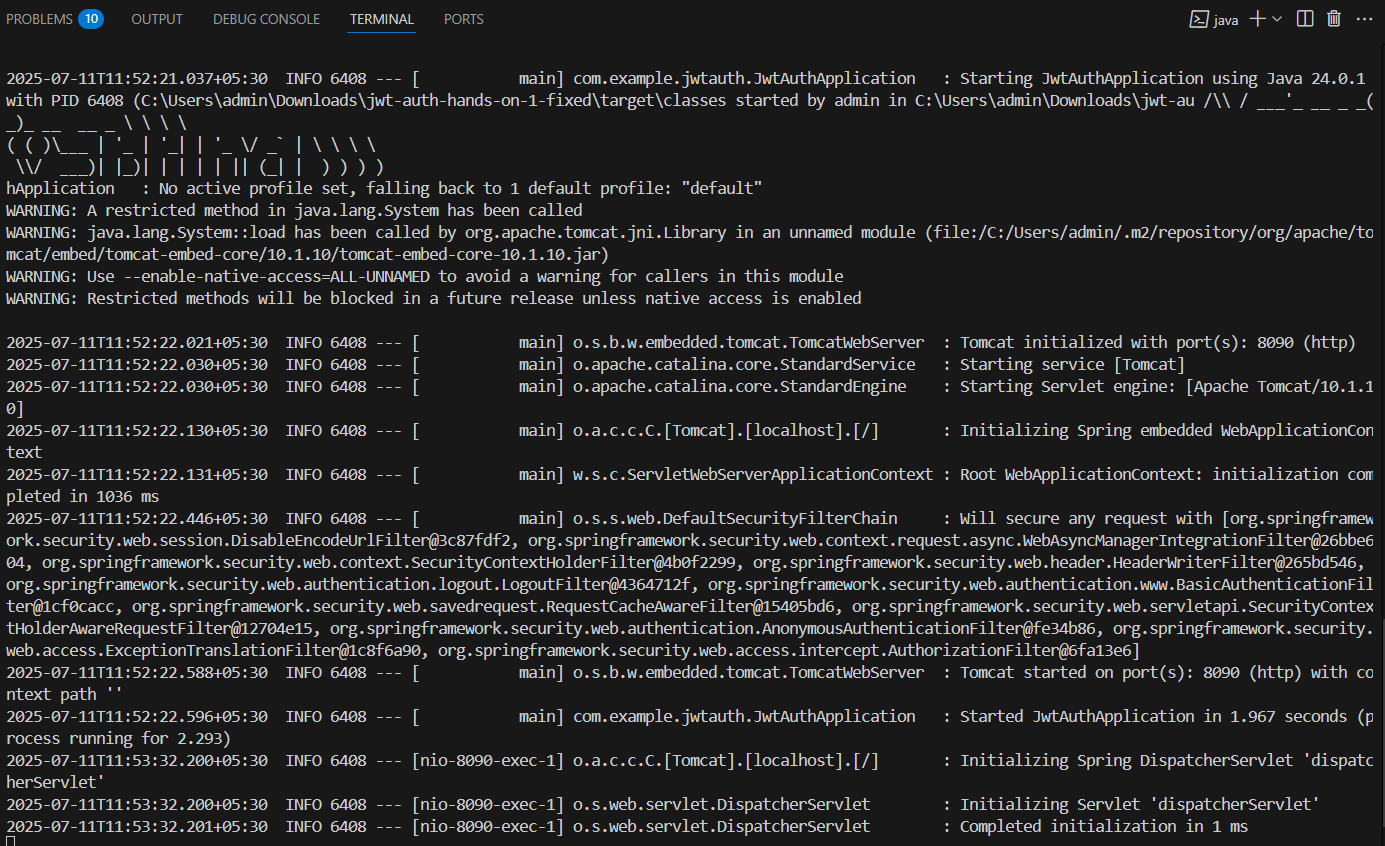
return new InMemoryUserDetailsManager(user);

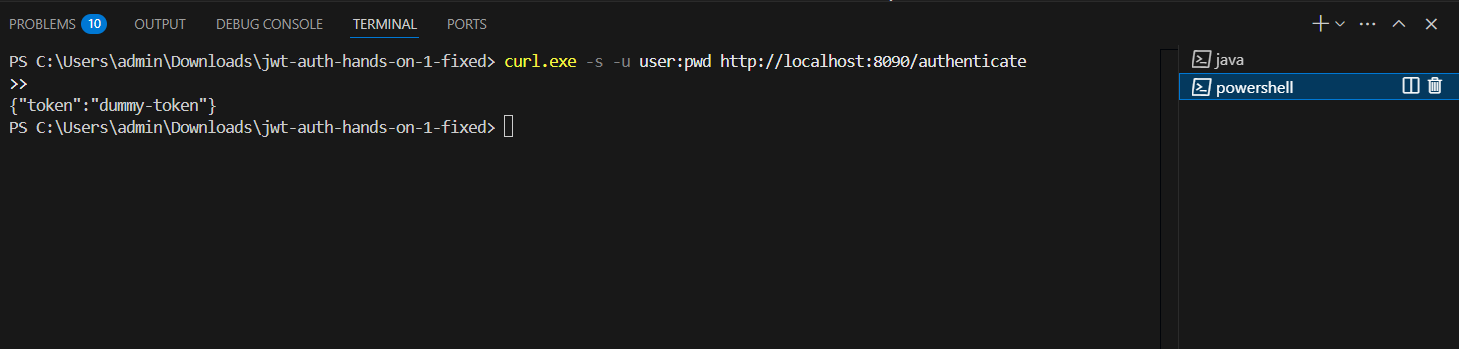
}

}

**OUTPUT:**

****

****

****