Day 19 Spring Security – JWT & OAuth

Day 18 Spring Security Intro

We added Spring Security Dependency to our existing ARS spring boot application.

Authentication – Is a process of verifying the Identity (Credentials) [Username/password, mobile(or)email with OTP] {Form based login}

Token Based Authentication – (Server will generate a token and send it to client after successful authentication (verifying credentials))

Every time, the client will add this security token to access all other resources from server.

Authorization – Ensuring the user has access to the specified resource or not . (Role based Access – RBA)

After adding Spring Security Dependency

1. Spring boot will enable security to all the end-points.
2. Spring Boot will add two custom end-points namely /login & /logout
3. Spring Boot Generates a default user with username as “user” and a dynamic password that will be displayed in the console when starting/re-starting the application. (This is only recommended for dev-environment)
4. Spring boot also adds to forms for /login & /logout end-points (UI – Thymeleaf frontend)
5. To disable auto password generation add following properties in application.properties file

“spring.security.user.name= admin

spring.security.user.password= admin123#”

1. To enable access to few end-points for all the users needs to add “SecurityConfig” class. – This enable security by-pass for few end-points namely (h2-console, swagger-ui, login page)

@Bean

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

http.authorizeHttpRequests(

auth -> auth.requestMatchers("/swagger-ui/\*\*", "/v3/api-docs/\*\*", "/h2-console/\*\*")

.authenticated()

// Swagger H2 Also Require Login

.anyRequest().authenticated())

.httpBasic(Customizer.*withDefaults*()) // Basic auth

.formLogin(Customizer.*withDefaults*()) // Form login also works

.csrf(csrf -> csrf.ignoringRequestMatchers("/h2-console/\*\*")) // Allow H2 console

.headers(headers -> headers.~~frameOptions~~().sameOrigin()); // For H2 console frames

**return** http.build();

}

Form based Authentication – Spring Security

Token Based - Authentication & Authorization

@PreAuthorize.hasRole(“Admin”) – Controller

JWT – JSON Web Token – It’s a Standard Security Mechanism

Jwt.io

JWT contains 3 parts (series of alpha numeric data)

1. Header (Algorithm used for hashing + token type)
2. Payload ( actual user details – non-sensitive info only [username/email/mobile])
3. Footer (Hashed header + hashed payload + security key]

Username=djh4ij87234nudhkfg (HAS, RSA, PSA – 256,384,512)

<https://www.geeksforgeeks.org/springboot/spring-boot-3-0-jwt-authentication-with-spring-security-using-mysql-database/>

*<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>  
</dependency>  
<dependency>  
<groupId>io.jsonwebtoken</groupId>  
<artifactId>jjwt-jackson</artifactId>  
<version>0.11.5</version>  
</dependency>*

<https://medium.com/@tericcabrel/implement-jwt-authentication-in-a-spring-boot-3-application-5839e4fd8fac>