

AuthController - ListUsers AddSecurity

Remove below lines in SecurityConfig.java

```
.requestMatchers("/**").permitAll()
```

```
.requestMatchers("/db-console/**").permitAll()
```

Update below lines in SecurityConfig.java

```
.requestMatchers("/token").permitAll()
```

To

```
.requestMatchers("/auth/token").permitAll()
```

Add below line in SecurityConfig.java

```
.requestMatchers("/auth/Users/add").permitAll()
```

```
.requestMatchers("/auth/users").hasRole("USER")
```

Updated securityFilterChain method

```
@Bean
public SecurityFilterChain securityFilterChain(HttpSecurity http) throws
Exception {
    http
        // CSRF configuration
        .csrf(csrf -> csrf
            .disable()) // Disable CSRF for stateless JWT
        authentication
        // Frame options for H2 console
        .headers(headers -> headers
            .frameOptions(frameOptions -> frameOptions.sameOrigin()))
        // Authorization configuration
        .authorizeHttpRequests(authorize -> authorize
            .requestMatchers("/auth/token").permitAll())
```

```

        .requestMatchers("/auth/users/add").permitAll()
        .requestMatchers("/auth/users").hasRole("USER")
        .requestMatchers("/swagger-ui/**").permitAll()
        .requestMatchers("/v3/api-docs/**").permitAll()
        .requestMatchers("/test").authenticated() // `/test`
requires authentication
    )
    // JWT-based authentication
    .oauth2ResourceServer(oauth2 -> oauth2
        .jwt(Customizer.withDefaults()))
    // Stateless session management
    .sessionManagement(session -> session
        .sessionCreationPolicy(SessionCreationPolicy.STATELESS))
    .headers(headers -> headers
        .frameOptions(frameOptions ->
            frameOptions.sameOrigin())); // Required for H2 console

    return http.build();
}

```

Output

Trying to list of users without authorizing

The screenshot shows a web browser window with the Swagger UI interface. The URL bar indicates the endpoint is `localhost:8080/swagger-ui/index.html#/Auth%20Controller/Users`. The interface shows the `GET /auth/users` endpoint with the description "List user api". Below the endpoint details, there is an "Execute" button. The "Responses" section shows a 401 status code with the message "Error: response status is 401". The response headers are displayed in a dark box:

```

Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Connection: keep-alive
Content-Length: 0
Date: Sun, 11 Aug 2024 02:46:34 GMT
Expires: 0
Keep-Alive: timeout=60
Pragma: no-cache
Vary: Origin,Access-Control-Request-Method,Access-Control-Request-Headers
WWW-Authenticate: Bearer
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 0

```

The bottom of the image shows a Windows taskbar with the system clock at 7:17 AM on 8/11/2024.

Update UserLoginDTO.java

```

package org.studyeasy.SpringRestdemo.payload.auth;

import io.swagger.v3.oas.annotations.media.Schema;
import io.swagger.v3.oas.annotations.media.Schema.RequiredMode;
import jakarta.validation.constraints.Email;
import jakarta.validation.constraints.Size;
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class UserLoginDTO {
    @Email
    @Schema(description = "Email Address", example = "user@user.com",
requiredMode = RequiredMode.REQUIRED)
    private String email;

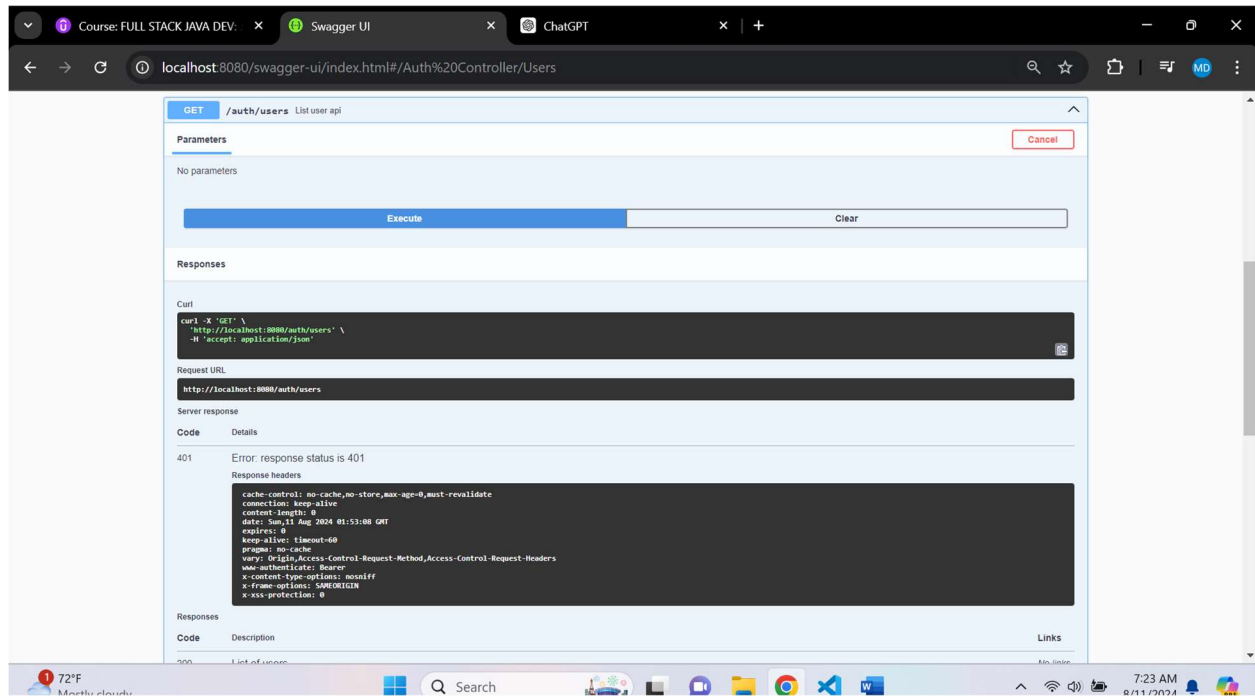
    @Size(min = 6, max = 20)
    @Schema(description = "Password", example = "pass987", requiredMode =
RequiredMode.REQUIRED, maxLength = 20, minLength = 6)
    private String password;
}

```

Output:- Token Generation

The screenshot displays the Swagger UI interface in a web browser. The URL bar shows `localhost:8080/swagger-ui/index.html#/Auth%20Controller/token`. The main content area shows the details of a successful `POST` request to `http://localhost:8080/auth/token`. The response status is `200`. The response body contains a JSON object with a `token` field, which holds a long, complex JWT token. The response headers include `cache-control: no-cache, no-store, max-age=0, must-revalidate`, `connection: keep-alive`, `content-type: application/json`, `date: Sun, 11 Aug 2024 01:51:23 GMT`, `expires: 0`, `keep-alive: timeout=60`, and `pragma: no-cache`. The browser's taskbar at the bottom shows the time as 7:21 AM on 8/11/2024.

And you still get error



Make changes in HomeController.java - commenting

```
package org.studyeasy.SpringRestdemo.controller;

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

// import io.swagger.v3.oas.annotations.security.SecurityRequirement;
// import io.swagger.v3.oas.annotations.tags.Tag;

@RestController
public class HomeController {

    @GetMapping("/")
    public String demo(){
        return "Hello World";
    }

    // @GetMapping("/test")
    // @Tag(name = "Test", description = "The Test API.")
    // @SecurityRequirement(name = "studyeasy-demo-api")
    // public String test(){
```

```
//      return "Test Api";
// }

}
```

In AuthController -> method -> Users -> add annotation -> SecurityRequirement

```
@SecurityRequirement(name = "studyeasy-demo-api")
```

Output – Insufficient scope

The screenshot shows the Swagger UI interface in a web browser. The endpoint being tested is `GET /auth/users` with the description "List user api". The "Execute" button has been clicked, and the response is displayed. The response status is 403 (Forbidden), and the message is "Error: response status is 403". The response headers are visible, showing various security-related headers like `cache-control`, `connection`, `content-length`, `date`, `expires`, `keep-alive`, `pragma`, `vary`, `www-authenticate`, `x-content-type-options`, `x-frame-options`, and `x-xss-protection`. The `www-authenticate` header indicates an "insufficient_scope" error, stating that the request requires higher privileges than provided by the access token.


Check it through -> jwt.io


It is valid scope and token; then why it is giving like this?

Course: FULL STACK JAVA DEV: x ChatGPT x Swagger UI x JSON Web Tokens - jwt.io x +

jwt.io

Learn about the upcoming changes to jwt.io and share your feedback →

 Debugger Libraries Introduction Ask

Crafted by  Auth0 by Okta

```
FhMC05ZjAyOWU2MDc1MjkiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoidXNlckB1c2VyLmNvbSIsImV4cCI6MTcyMzM0NTE3MywiaWF0IjoxNzIzNDQ1NTc5LjZyZW50IiwiaXNjb3VudCI6ImV4X6FvAN45i3yJCAN0jYr0NkWmzdZviRs5JtVXuHAq36VKP53gnjS3jd6xPnEBY0zHobVs0_ctvjJjczAgqcSjiMPnuK_wItG5fKsx8NdXqrJQvfrB-dxZipj0R-HPWYRXp2RrE5G7_gVI9b8QNACy00n-CQE055NwoBUdw-761ak7KKBVbPUIAsEVCgimjs3XGmGQAppjCmPhxirs80CWpCcxX_JIdmcSYBFxczYmR9vAnfLdkZnBkoyHxr040dRu_H1nubCE-6D3pqrFG03LvW-rSNHhFnR3Ahwvso0P6BZZJvvqW2i97PIgxQ0SbL
```

```
{  "kid": "c731d7fc-20a3-4569-81a0-9f029e607529",  "alg": "RS256"}
```

PAYLOAD: DATA

```
{  "iss": "self",  "sub": "user@user.com",  "exp": 1723345173,  "iat": 1723341573,  "scope": "ROLE_USER"}
```

VERIFY SIGNATURE

```
RSASHA256(  base64UrlEncode(header) + "." +
```

72°F Mostly cloudy 7:31 AM 8/11/2024

.hasRole() -> Used for monolithic application. So use has Authority()