AuthController – Get Profile

In AuthController.java -> add this below code

```
@GetMapping(value = "/profile", produces = "application/json")
    @Operation(summary = "View Profile")
    @ApiResponse(responseCode = "200", description = "List of users")
    @ApiResponse(responseCode = "401", description = "Token missing")
    @ApiResponse(responseCode = "403", description = "Token error")
    @SecurityRequirement(name = "studyeasy-demo-api")
    public ProfileDTO profile(Authentication authentication){
        String email = authentication.getName();
        Optional<Account> optionalAccount = accountService.findByEmail(email);
        if(optionalAccount.isPresent()){
            Account account = optionalAccount.get();
            ProfileDTO profileDTO = new ProfileDTO(account.getId(),
account.getEmail(), account.getAuthorities());
            return profileDTO;
        }
        return null;
    }
```

Create a DTO -> in payload -> ProfileDTO.java

```
package org.studyeasy.SpringRestdemo.payload.auth;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.Setter;

@Setter
@Getter
@AllArgsConstructor
public class ProfileDTO {
    private long id;
    private String email;
    private String authorities;
}
```

In AccountService.java -> add this below code

```
public Optional<Account> findByEmail(String email){
        return accountRepository.findByEmail(email);
    }
```

In SecurityConfig.java -> add below line

```
.requestMatchers("/auth/profile").authenticated()
```

Updated SecurityConfig.java

```
package org.studyeasy.SpringRestdemo.security;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.authentication.ProviderManager;
import org.springframework.security.authentication.dao.DaoAuthenticationProvider;
import org.springframework.security.config.Customizer;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import
org.springframework.security.config.annotation.web.configuration.EnableWebSecurit
y;
import org.springframework.security.config.http.SessionCreationPolicy;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.oauth2.jwt.JwtDecoder;
import org.springframework.security.oauth2.jwt.JwtEncoder;
import org.springframework.security.oauth2.jwt.NimbusJwtDecoder;
import org.springframework.security.oauth2.jwt.NimbusJwtEncoder;
import org.springframework.security.web.SecurityFilterChain;

import com.nimbusds.jose.JOSEException;
import com.nimbusds.jose.jwk.JWKSet;
import com.nimbusds.jose.jwk.RSAKey;
import com.nimbusds.jose.jwk.source.JWKSource;
import com.nimbusds.jose.proc.SecurityContext;

@Configuration
@EnableWebSecurity
public class SecurityConfig {
    private RSAKey rsaKey;

    @Bean
    public JWKSource<SecurityContext> jwkSource() {
        rsaKey = Jwks.generateRsa();
        JWKSet jwkSet = new JWKSet(rsaKey);
        return (jwkSelector, securityContext) -> jwkSelector.select(jwkSet);
    }

    @Bean
    public static PasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder();
    }

    // @Bean
    // public InMemoryUserDetailsManager users() {
    // return new InMemoryUserDetailsManager(
    // User.withUsername("chaand")
    // .password("{noop}password")
    // .authorities("read")
    // .build());
    // }

    @Bean
    public AuthenticationManager authManager(UserDetailsService
userDetailsService) {
        var authProvider = new DaoAuthenticationProvider();
        authProvider.setPasswordEncoder(passwordEncoder());
        authProvider.setUserDetailsService(userDetailsService);
        return new ProviderManager(authProvider);
    }

    @Bean
    JwtEncoder jwtEncoder(JWKSource<SecurityContext> jwks) {
        return new NimbusJwtEncoder(jwks);
    }

    @Bean
    JwtDecoder jwtDecoder() throws JOSEException {
        return NimbusJwtDecoder.withPublicKey(rsaKey.toRSAPublicKey()).build();
    }

    @Bean
```

```java
    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").hasAnyAuthority("SCOPE_ADMIN")
                        .requestMatchers("/auth/profile").authenticated()
                        .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();
    }

}
```

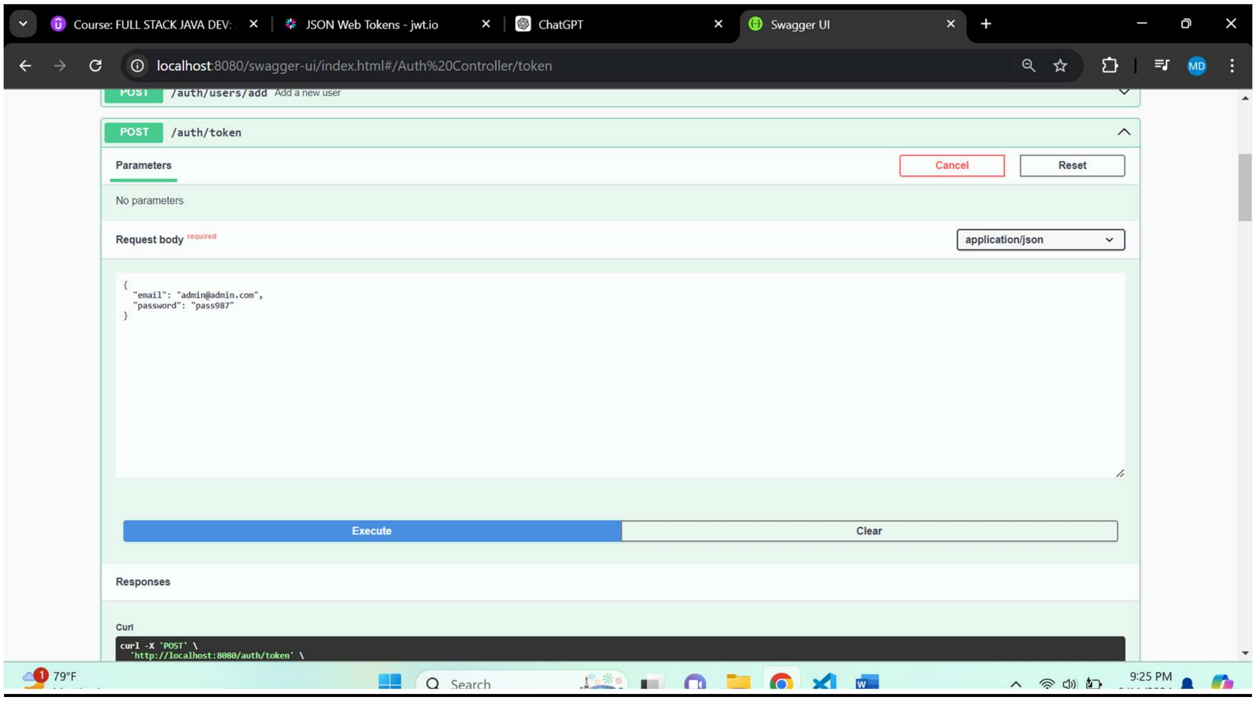
Output

If you try to access list of users or view profile directly, you get error (401).

If you try generating token and then access as Admin, you get list of data and profile.

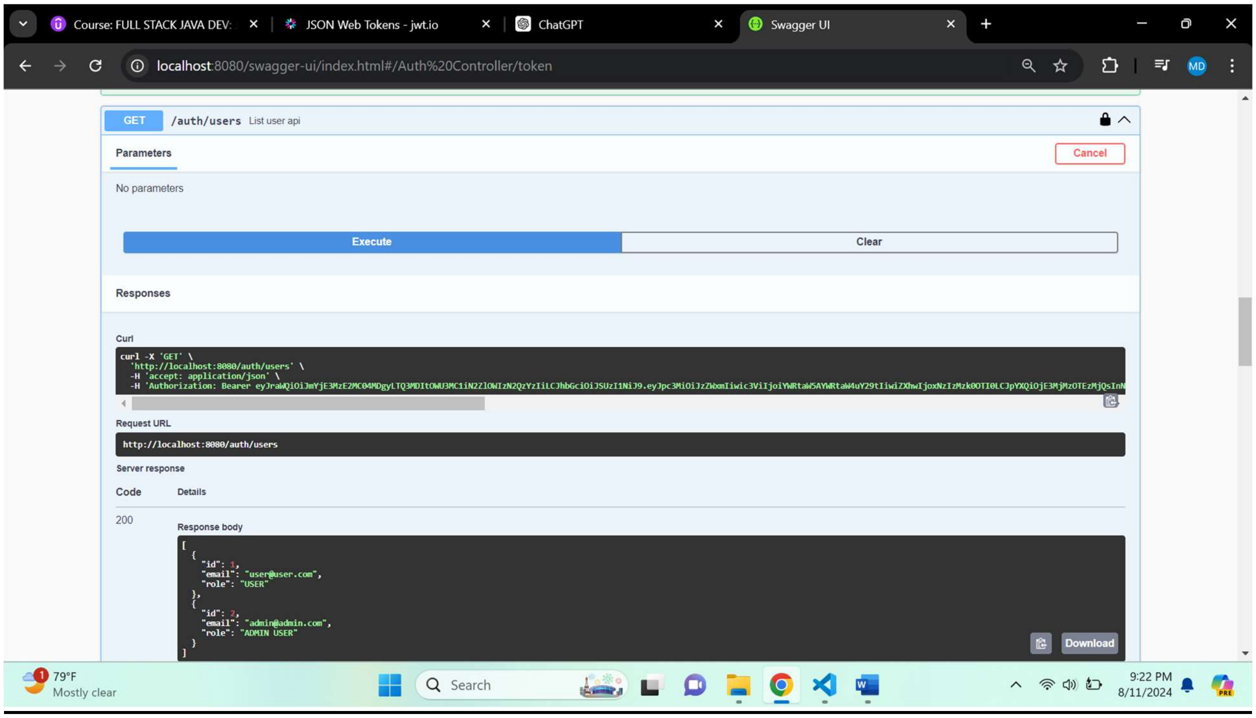
If you try generating token and then access as User, you will not get list of data, but you will get profile.

ADMIN



```
{
  "email": "admin@admin.com",
  "password": "pass987"
}
```

```
curl -X 'POST' \
  'http://localhost:8080/auth/token' \
```



/auth/users List user api

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8080/auth/users' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer eyJraWQiOiJmYjE3MzE2MC04MDgyLTQ3MDItOWU3MC1iN2ZlOWIzN2QzYzIiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoiYWRtaW5AYWRtaW4uY29tIiwiZXhwIjoxNzIzMzk0OTI0LCJpYXQiOjE3MjMzOTEzMjQsInN
```

Request URL

```
http://localhost:8080/auth/users
```

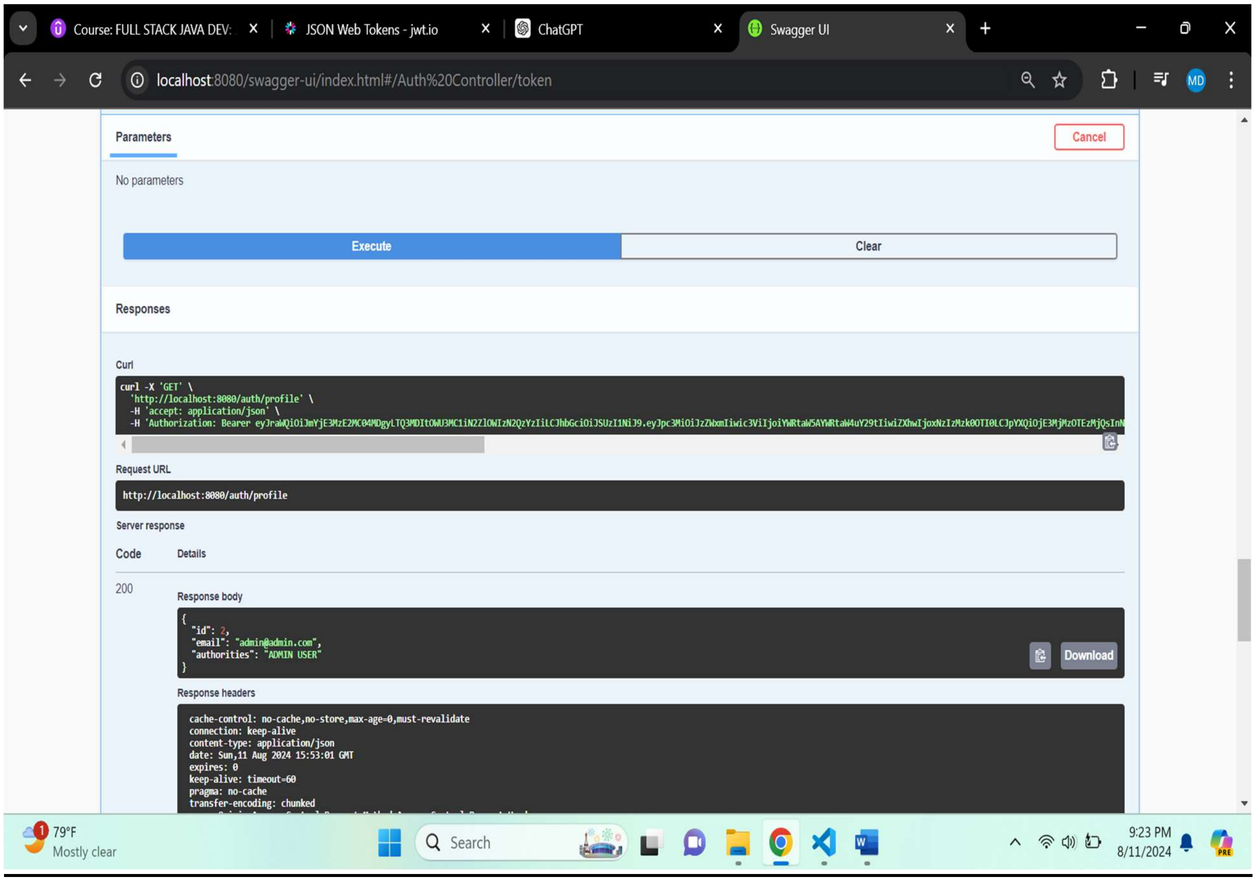
Server response

Code Details

200

Response body

```
[
  {
    "id": 1,
    "email": "user@user.com",
    "role": "USER"
  },
  {
    "id": 2,
    "email": "admin@admin.com",
    "role": "ADMIN USER"
  }
]
```



Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8080/auth/profile' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer eyJraWQiOiJmYjE3MzE2MC04MDgyLTQ3MDItOWU3MC1iN2ZlOWIzN2QzYzIiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoiYWRtaW5AYWRtaW4uY29tIiwiZXhwIjoxNzIzMzk0OTI0LCJpYXQiOjE3MjMzOTEzMjQsInN
```

Request URL

```
http://localhost:8080/auth/profile
```

Server response

Code Details

200

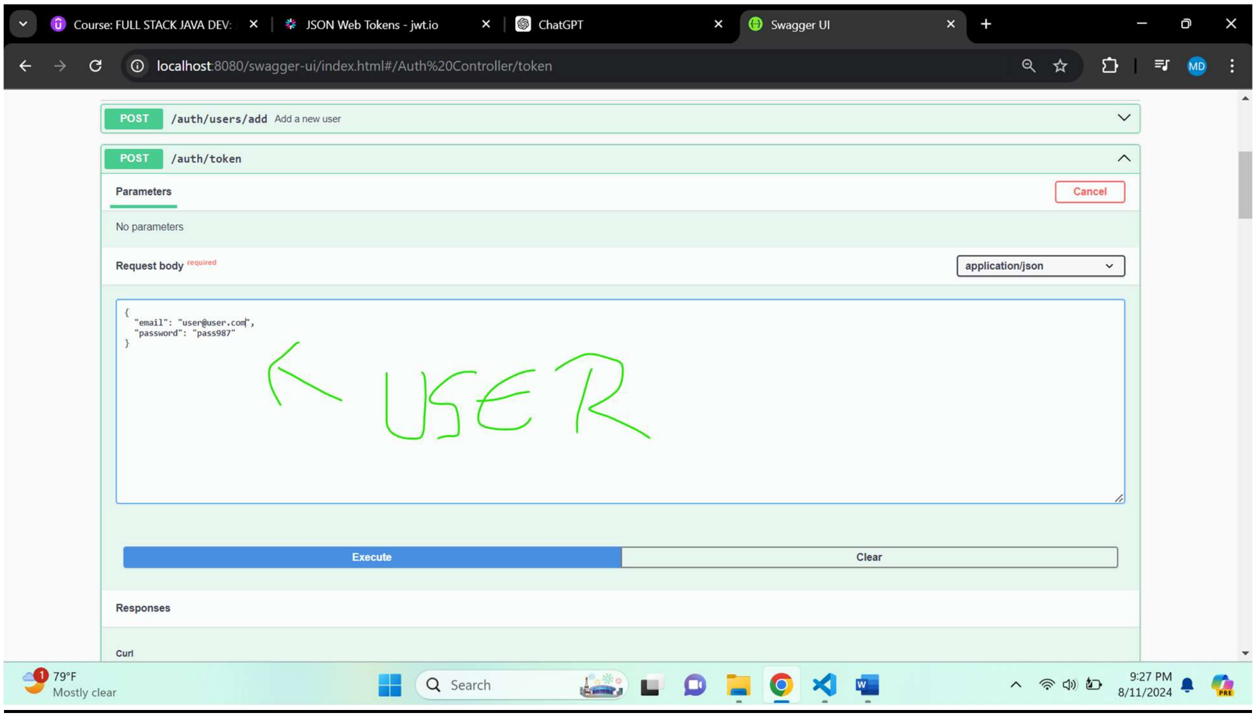
Response body

```
{
  "id": 2,
  "email": "admin@admin.com",
  "authorities": "ADMIN USER"
}
```

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-type: application/json
date: Sun,11 Aug 2024 15:53:01 GMT
expires: 0
keep-alive: timeout=60
pragma: no-cache
transfer-encoding: chunked
```

USER

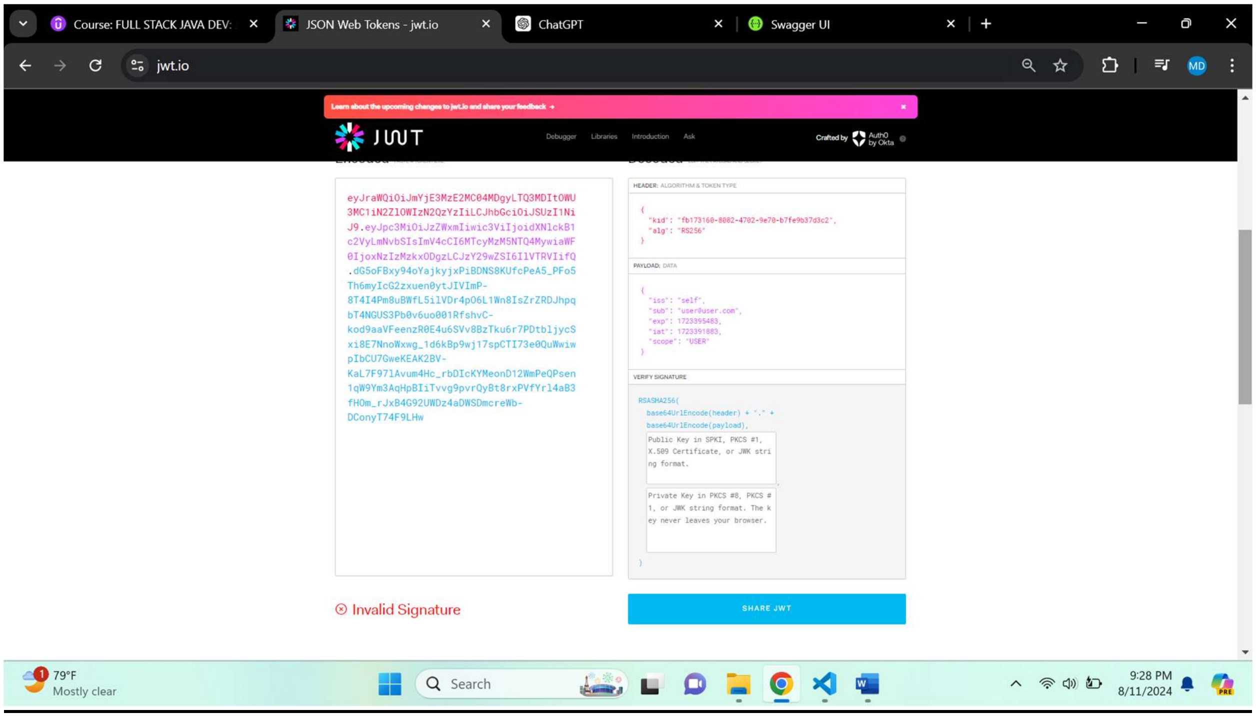


No parameters

Request body required

```
{
  "email": "user@user.com",
  "password": "pass987"
}
```

← USER



```
eyJraWQiOiJmYjE3MzE2MC04MDgyLTQ3MDItOWU
3MC1iN2Z1OWIzN2QzYzIiLCJhbGciOiJSUzI1Ni
J9.eyJpc3MiOiJzZWxmIiwic3ViIjoidXNlckB1
c2VyLmNvbSIsImV4cCI6MTcyMzM5NTQ4MywiaWF
0IjoxNzIzMzkxODgzLCJzY29wZSI6IlVTRVIifQ
.dG5oFBxy94oYajkyjxPiBDNS8KUfcPeA5_PFo5
Th6myIcG2zxuen0ytJIVImP-
8T4I4Pm8uBWfL5ilVDr4pO6L1Wn8IsZrZRDJhpq
bT4NGUS3Pb0v6uo001RfshvC-
kod9aaVFeenzR0E4u6SVv8BzTku6r7PDtbljycS
xi8E7NnoWxwg_1d6kBp9wj17spCTI73e0QuWwiw
pIbCU7GweKEAK2BV-
KaL7F97lAvum4Hc_rbDIcKYMeonD12WmPeQPsen
1qW9Ym3AqHpBIiTvvg9pvrQyBt8rxPVfYrl4aB3
fHOm_rJxB4G92UWDz4aDWSDmcreWb-
DConyT74F9LHw
```

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "kid": "fb173160-8082-4702-9e70-b7fe9b37d3c2",
  "alg": "RS256"
}
```

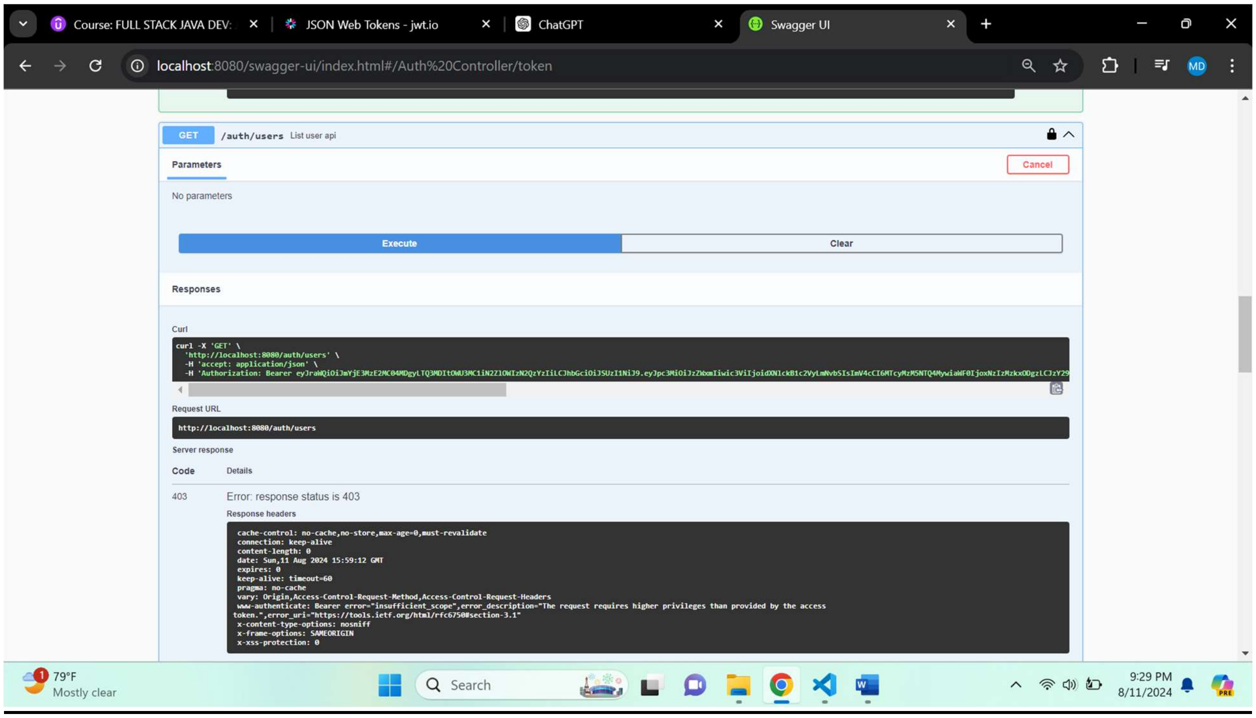
PAYLOAD: DATA

```
{
  "iss": "self",
  "sub": "user@user.com",
  "exp": 1723395483,
  "iat": 1723391883,
  "scope": "USER"
}
```

VERIFY SIGNATURE

```
RSASHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  Public Key in SPKI, PKCS #1, X.509 Certificate, or JWK string format.
  Private Key in PKCS #8, PKCS #1, or JWK string format. The key never leaves your browser.
)
```

⊗ Invalid Signature


Course: FULL STACK JAVA DEV:
JSON Web Tokens - jwt.io
ChatGPT
Swagger UI
localhost:8080/swagger-ui/index.html#/Auth%20Controller/token
MD
79°F
Mostly clear
Search
9:29 PM
8/11/2024

GET /auth/users List user api

Parameters

Cancel

No parameters

Execute | Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8080/auth/users' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer eyJraWQiOiJmYjE3MzE2MC04MDgyLTQ3MDItOWU3MC1iN2Z1OWIzN2QzYzIiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoidXNlckB1c2VyLmNvbSIsImV4cCI6MTcyMzM5NTQ4MywiaWF0IjoxNzIzMzkxODgzLCJzY29
```

Request URL

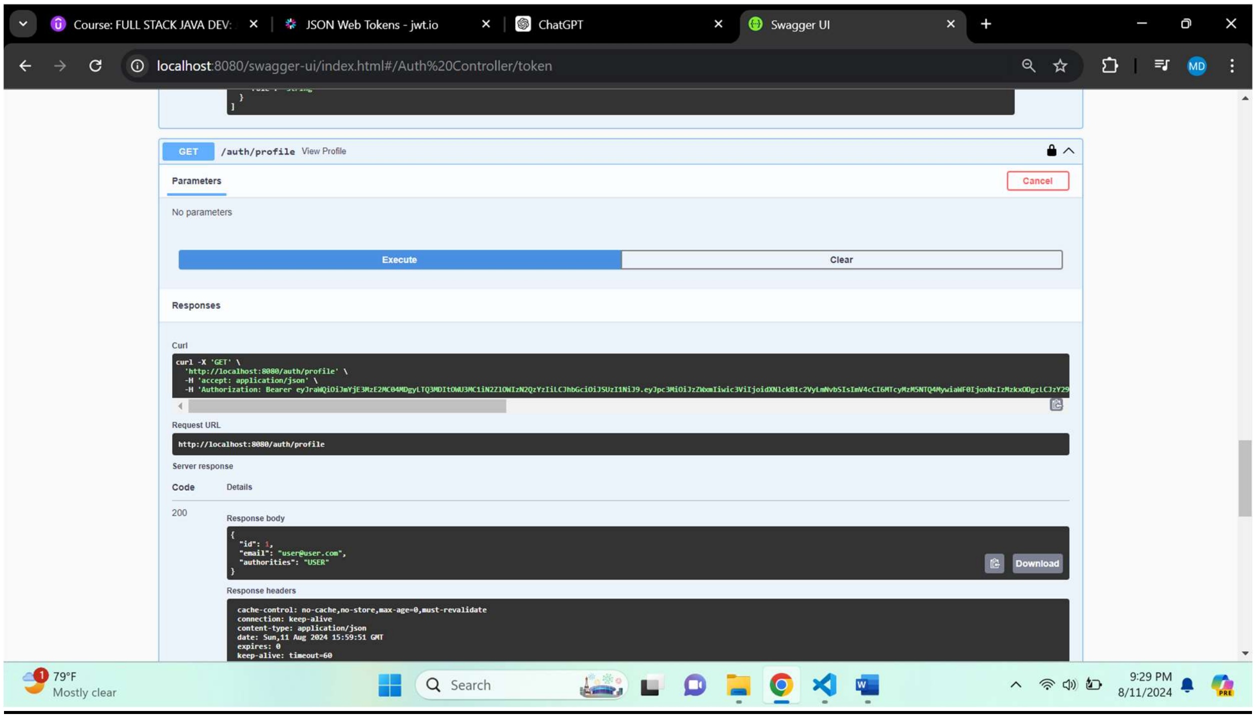
```
http://localhost:8080/auth/users
```

Server response

Code	Details
403	Error: response status is 403

Response headers

```
 cache-control: no-cache,no-store,max-age=0,must-revalidate
 connection: keep-alive
 content-length: 0
 date: Sun,11 Aug 2024 15:59:12 GMT
 expires: 0
 keep-alive: timeout=60
 pragma: no-cache
 vary: Origin,Access-Control-Request-Method,Access-Control-Request-Headers
 www-authenticate: Bearer error="insufficient_scope",error_description="The request requires higher privileges than provided by the access
token.",error_uri="https://tools.ietf.org/html/rfc6750#section-3.1"
 x-content-type-options: nosniff
 x-frame-options: SAMEORIGIN
 x-xss-protection: 0
```


Course: FULL STACK JAVA DEV:
JSON Web Tokens - jwt.io
ChatGPT
Swagger UI
localhost:8080/swagger-ui/index.html#/Auth%20Controller/token
MD
79°F
Mostly clear
Search
9:29 PM
8/11/2024

GET /auth/profile View Profile

Parameters

Cancel

No parameters

Execute | Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8080/auth/profile' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer eyJraWQiOiJmYjE3MzE2MC04MDgyLTQ3MDItOWU3MC1iN2Z1OWIzN2QzYzIiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoidXNlckB1c2VyLmNvbSIsImV4cCI6MTcyMzM5NTQ4MywiaWF0IjoxNzIzMzkxODgzLCJzY29
```

Request URL

```
http://localhost:8080/auth/profile
```

Server response

Code	Details
200	Response body

```
{
  "id": 1,
  "email": "user@user.com",
  "authorities": "USER"
}
```

Download

Response headers

```
 cache-control: no-cache,no-store,max-age=0,must-revalidate
 connection: keep-alive
 content-type: application/json
 date: Sun,11 Aug 2024 15:59:51 GMT
 expires: 0
 keep-alive: timeout=60
```