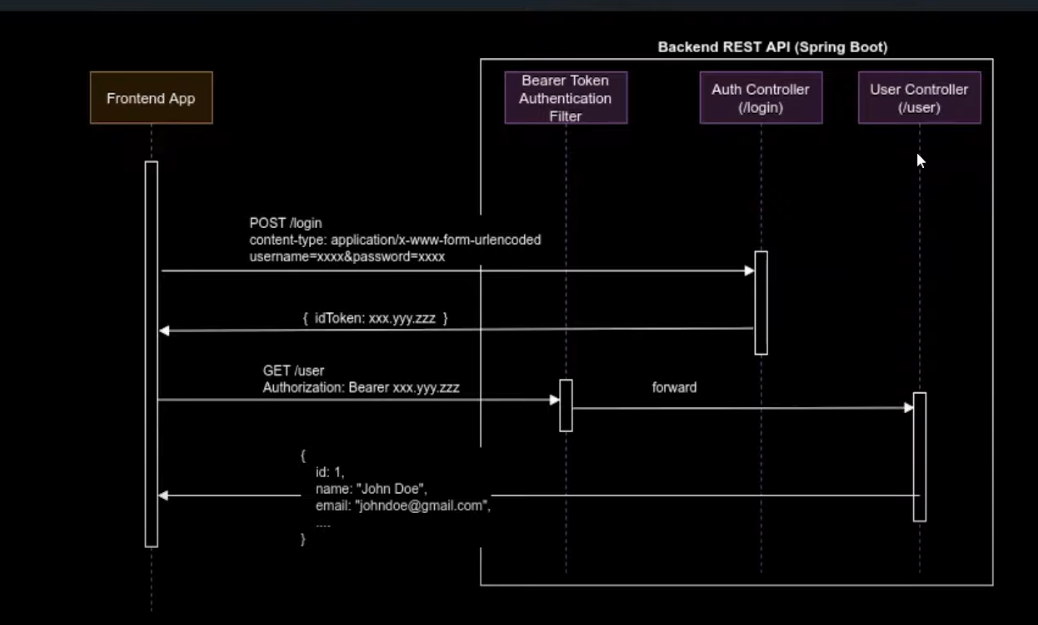
**WorkFlow**

B1: Đầu tiên khi gửi 1 request đến server , request đó sẽ chạy qua tất cả các filterChain đã được cấu hình trong dự án.

B2: Sau đó nó sẽ được xác thực nếu mà request đó chưa xác thực -> chưa có token thì sẽ trả về trạng thái 401 || 403 và sẽ chuyển đến trang đăng nhập.

B3: Sau khi đăng nhập gửi lại token về cho client và lưu trong localStorage -> mỗi request sau này sẽ gửi kèm token này trong phần header Authorization để server có thể sử lý và trả ra thông tin của enpoint truy cập.

B4: Khi truy cập vào 1 enpoint server sẽ kiểm tra xem token đó còn hạn hay không nếu hết hạn -> đăng nhập lại để lấy token mới.



**Các lớp cần biết trong Spring security**

-**SimepleGrantedAuthority**: Là 1 lớp được triển khai từ interface GrantedAuthority, lớp này đại diện cho quyền của 1 người dùng.

+) -> Nó có phương thức để lấy quyền ra như là getAuthority();

-**UserDetails**: Là 1 interface đại diện cho 1 thông tin chi tiết của người dùng đã được xác thực và nó cung cấp các method để lấy thông tin chi tiết từ 1 người dùng đã xác thực.

->method:Username,Password,Authorities,Account Non-Expired,Account Non-Locked,Credentials Non-Expired,Enabled.

-Tác dụng:

+)Khi đăng nhập người dùng Security sẽ sử dụng username và password từ Usertails để xác thực.

+)Dùng để cung cấp thông tin cần thiết cho ủy quyền như là xem người dùng này có được truy cập vào tài nguyên này hay không.

-Cách lấy thông tin UserDetails:

-Spring security cung cấp 1 interface là UserDetailsService -> có 1 method là loadUserByUserName dùng để lấy UserDetails từ database.

-**UsernamePasswordAuthenticationToken:** là 1 lớp dùng để xác thực người dùng dựa trên username và password.-> Nó cần những tham số như là username,password,authorities để khởi tạo nó.

-Tác dụng:

+) Chứa thông tin xác thực của người dùng như username và password.

+)Nó được dùng để gửi tới AuthenticationManager bằng cách gọi phương thức authenticate để thực hiện xác thực người dùng -> Nó xác thực bằng cách gọi AuthenticationProvider để đối chiếu với dữ liệu trong csdl -> AuthenticationProvider gọi đến loadUserbyUserName của UserDetailsService để lấy người dùng và gọi đến PasswordEncoder để mã hóa mật khẩu của người dùng ở trong cơ sở dữ liệu -> Sau khi xác thực thành công thì nó trả ra 1 interface Authentication hoàn chỉnh.

-Việc setAuthentication cho SecurityContextHolder là vì workflow của nó, khi có 1 request đến thì nó sẽ thực hiện chạy vào phương thức doFilterInternal trước sau đó mới chạy vào controller để lấy giữ liệu -> Tại sao nó chạy chạy như vậy ? tại vì khi ta config SecurityFilterChain ta đã sử dụng addFilterBefore và ta đã cho jwtAuthentication lên trước UsernamePasswordAuthenticationFilter.class -> ta phải setAuthentication cho SecurityContextHolder trước sau đó nó sẽ thực thi việc so sánh xem trong securityContextHolder có dữ liệu của token không thì nó sẽ nhả giữ liệu từ controller ra.