Controller gọi Service, Service gọi Repository và Mapper

Tạo Repository -> Tạo Service -> Tạo DTO (Req và Resp) -> Tạo Mapper -> Tạo Controller

Thứ tự:

Entity ->dto (request, response), repository -> mapper, (passwordEncoder) -> service -> controller

ANOTATION

1. @Value() => Dùng để đọc biến từ file yaml

TOKEN

Để tạo 1 token cần

+ 1 header

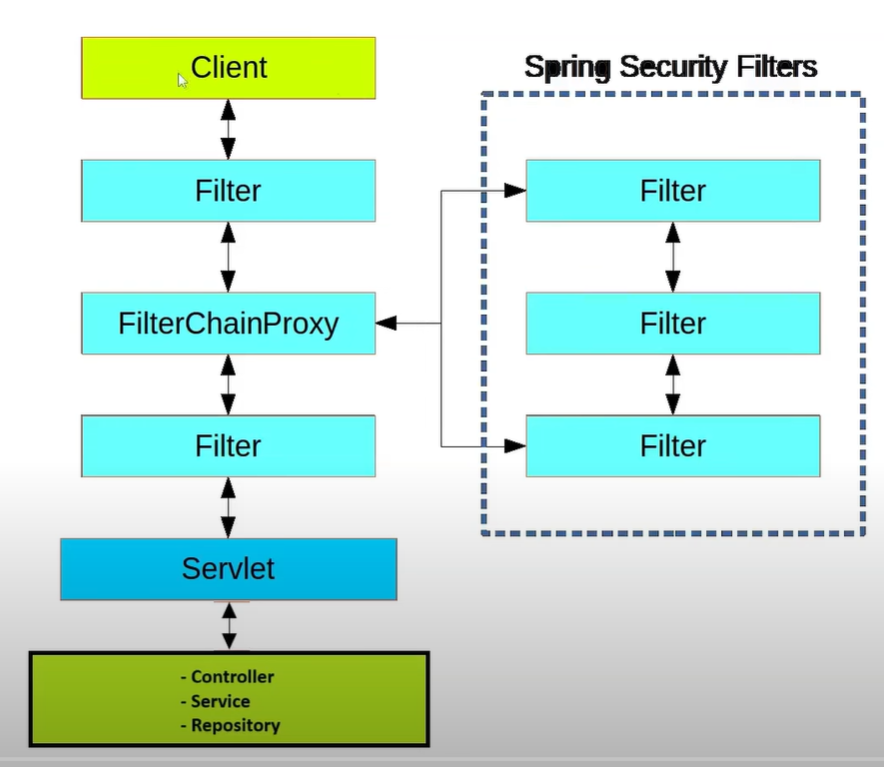
+ body => Nội dung gởi đi kèm trong token (username, issuer -> Định danh token do ai issuer, issuerTime)

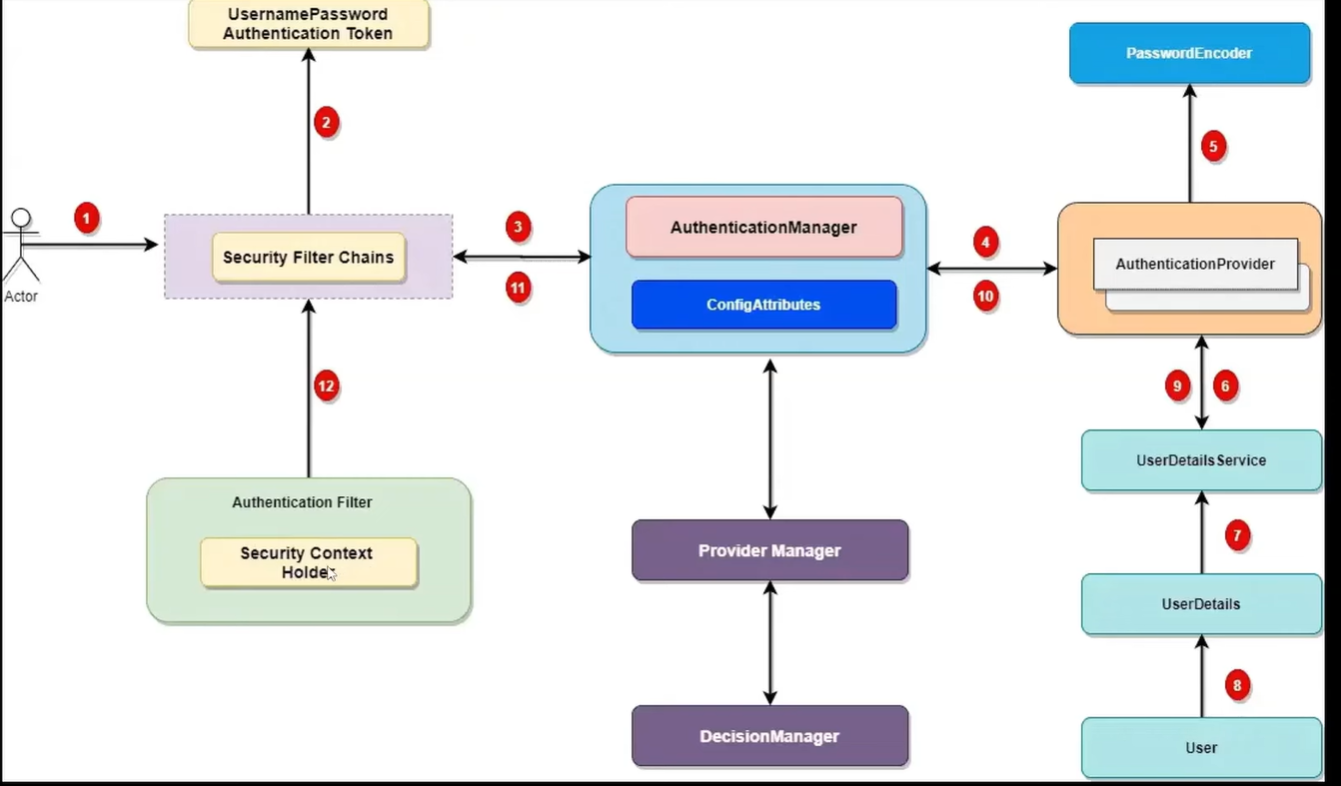
+ payload

* Sign token

Spring Security Architecture

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-oauth2-resource-server</artifactId>  
</dependency>





**@RequiredArgsConstructor  
@FieldDefaults**(**level** = AccessLevel.*PRIVATE*, **makeFinal** = true) *// Thay the AutoWide*

**@Slf4j => Log**

**SPRING SECURITY**

* Chỉ lấy được thông tin của chính mình

**@PostAuthorize**("returnObject.username == authentication.name") *// Goi method truoc va sau do kiem tra xem co phai Role mong muon hay khong moi tra ket qua?, authentication.name la USER hien tai*public UserResponse getUser(String id){  
 *log*.info("In method gerUsersByID");  
 return userMapper.toUserResponse(userRepository.findById(id). *// Nếu tìm thấy thì trả về* orElseThrow(() -> new AppException(ErrorCode.*USER\_NOT\_EXISTED*))); *// Còn không tìm thấy thì báo lỗi*}

Trong springSecurity khi mà User được xác thực thành công để đăng nhập thì thông tin của User đó được lưu trữ trong SecurityContextHolder

GraalVM

Bài 14: Validation bằng custom anotation

*Vấn đề: Tạo 1 custom anotation để validation DOB*

1. Tạo Package validator

1.1 Interface Anotation

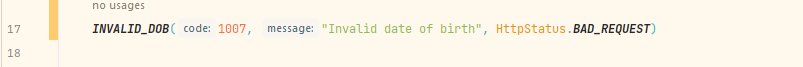
1.2 Class Validator

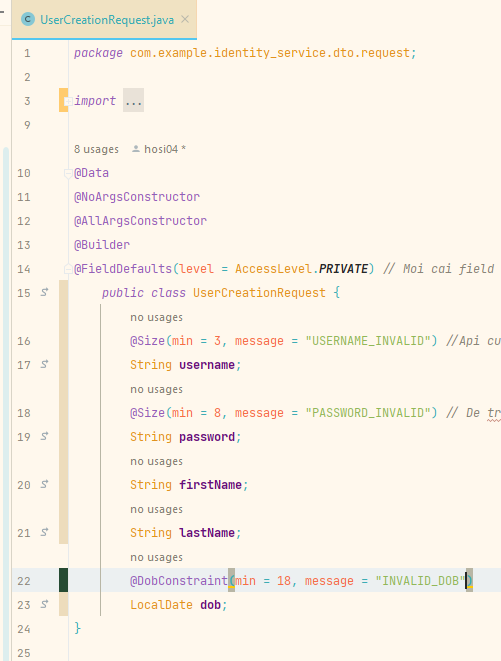
1. Click nào 1 cái anotaion nào đó có sẵn để xem cấu trúc khai báo
2. Target: tức là cái anotaion này sẽ được apply ở đâu? (ở 1 biến => field)
3. Retention: anotation này sẽ được xử lý lúc nào?
4. Constraint: Class sẽ chịu trách nhiệm validation cho anotation này

- Cấu trúc cở bản của 1 anotation









ChronoUnit: Có nhiều phép toán hữu ích trong ngày tháng

Bài 14: Validation bằng custom anotation (ADVANCE)

Cải tiến trong GlobalExceptionHandler

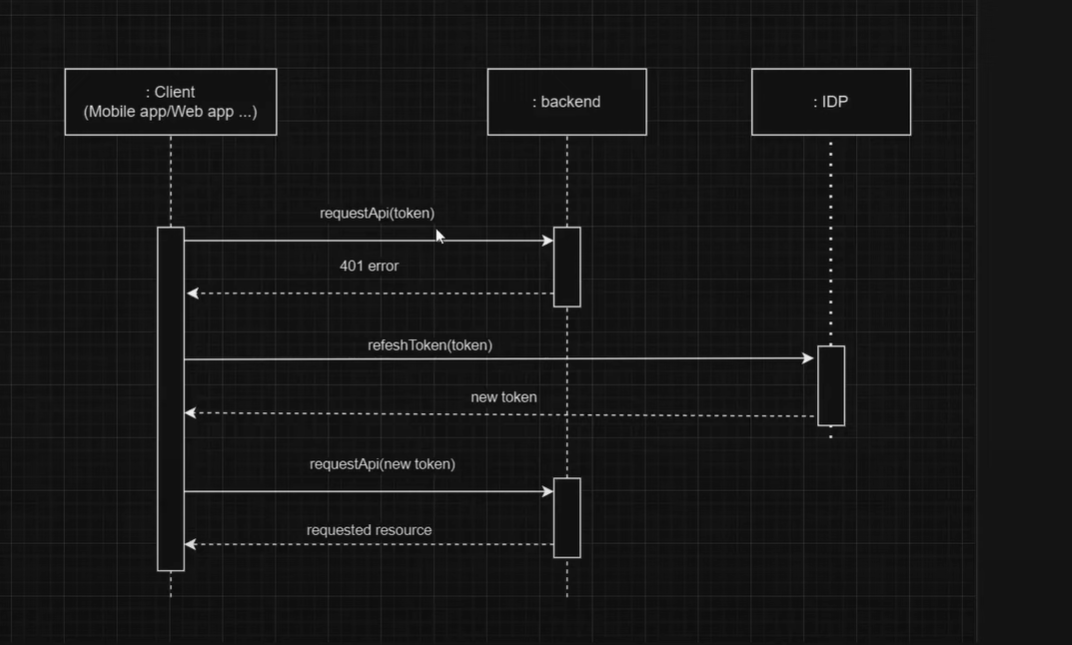
BÀI 15: logout with JWT token

* Thêm jwtID vào generateToken
* Thêm 1 entity InvalidatedToken (id, expiryTime)
* Tạo 1 repository InvalidatedTokenRepository
* Tạo 1 hàm Logout trong AuthenticationService để thực hiện chức năng logout

refresh JWT token

* Tạo 1 function refreshToken trong AuthenticationService(refrestRequest req)
* Tạo 1 req nữa
* Tạo 1 controller trong AuthenticationController

refresh JWT token (phần 2) align với Oauth2



Unit Test

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
</dependency>

<dependency>  
 <groupId>com.fasterxml.jackson.datatype</groupId>  
 <artifactId>jackson-datatype-jsr310</artifactId>  
</dependency>

* 1 cái req đi vào thi đầu tiên nó sẽ qua Controller sau đó quá Service sau đó đến repository (Gần như là lớp cuối cùng chứa req)
* Tập trung viết Unit Test cho Controller và Service

Ví dụ tạo Test Controller mà trong Controler có gọi đến function của Service thì chúng ta sẽ cần Mock cái service đấy. mục đích kiểu như là nó sẽ bỏ qua cái function của Service và chỉ test cái Controller

METHOD Builder là của Lombok