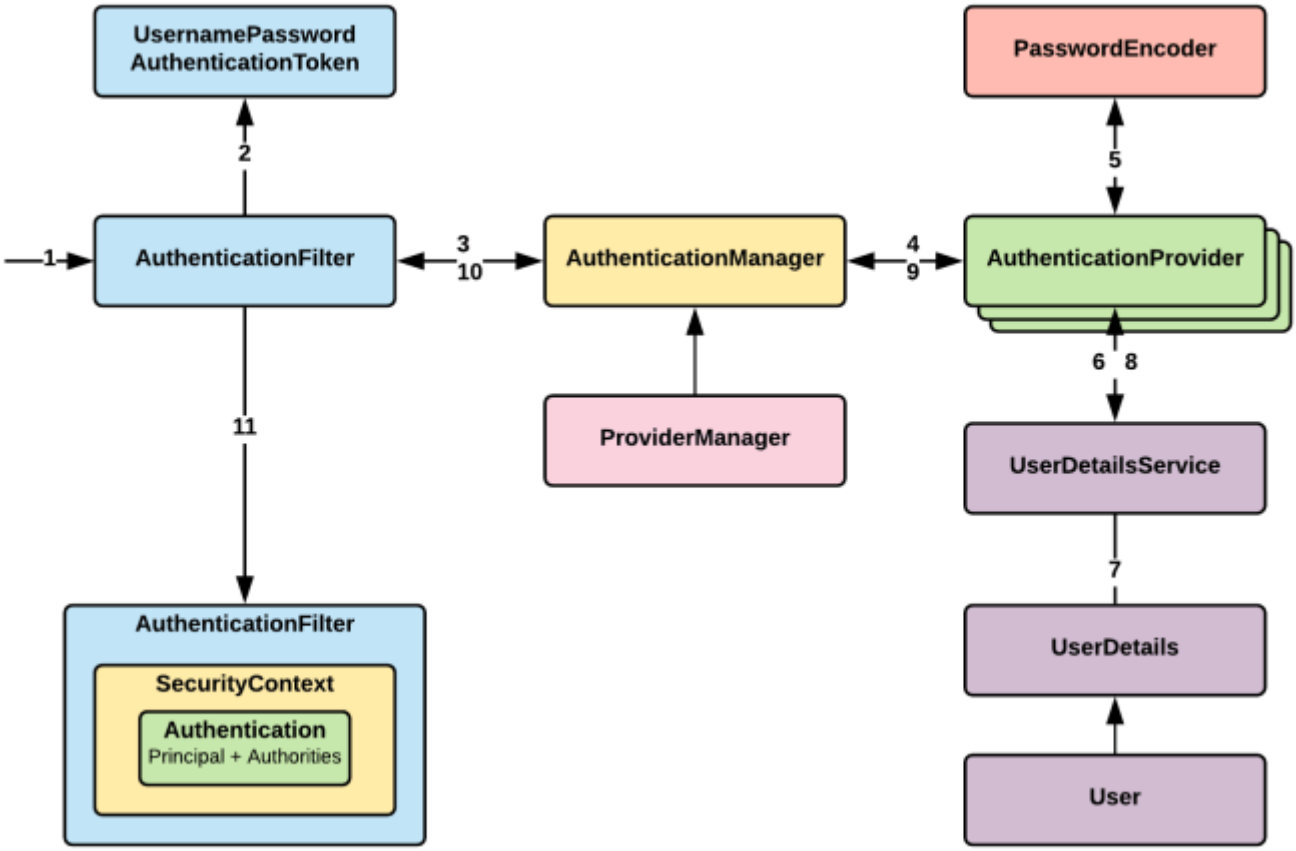
# Spring Security - Authentication and Authorization

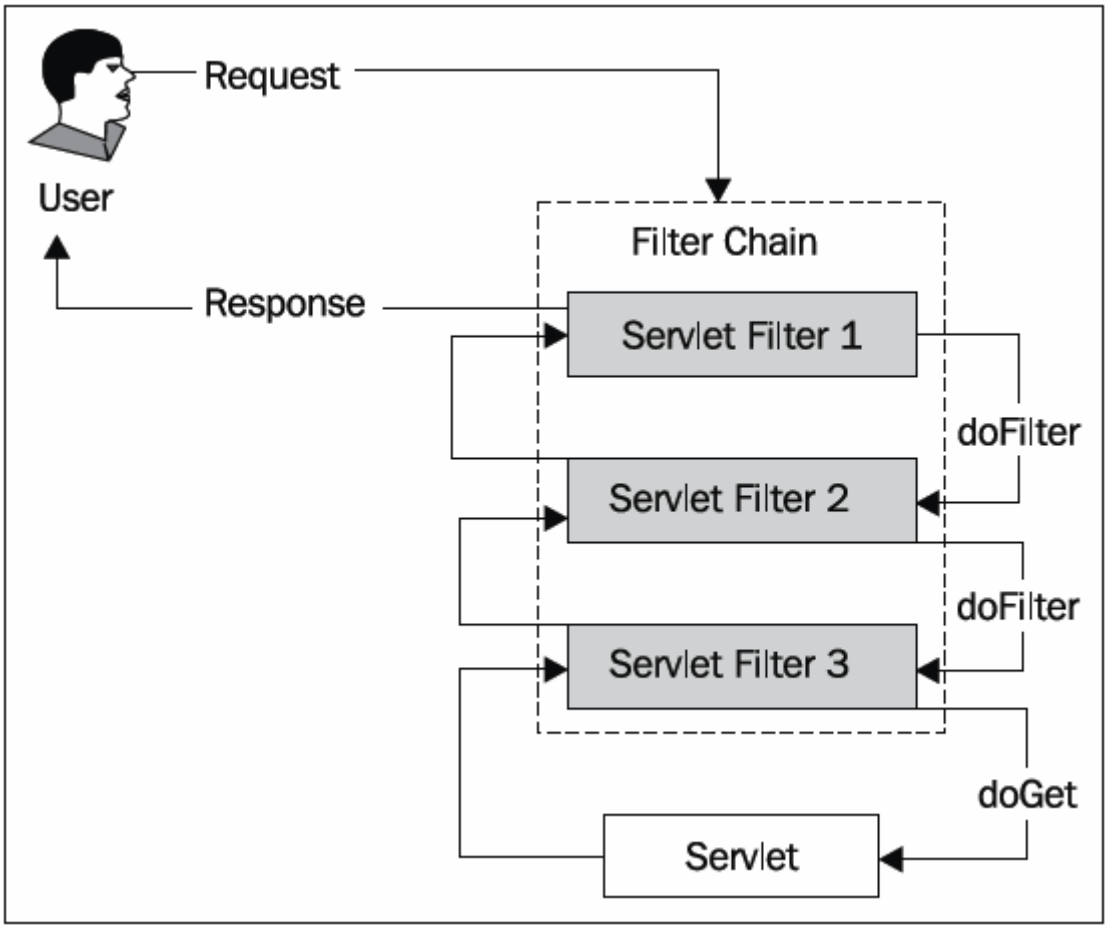
> API에 권한 기능이 없으면, 아무나 회원 정보를 조회하고 수정하고 삭제할 수 있다. 따라서 이를 막기 위해 인증된 유저만 API를 사용할 수 있도록 해야하는데, 이때 사용할 수 있는 해결책 중 하나가 Spring Security다.

스프링 프레임워크에서는 인증 및 권한 부여로 리소스 사용을 컨트롤 할 수 있는 `Spring Security`를 제공한다. 이 프레임워크를 사용하면, 보안 처리를 자체적으로 구현하지 않아도 쉽게 필요한 기능을 구현할 수 있다.




Spring Security는 스프링의 `DispatcherServlet` 앞단에 Filter 형태로 위치한다. Dispatcher로 넘어가기 전에 이 Filter가 요청을 가로채서, 클라이언트의 리소스 접근 권한을 확인하고, 없는 경우에는 인증 요청 화면으로 자동 리다이렉트한다.

## Spring Security Filter




Filter의 종류는 상당히 많다. 위에서 예시로 든 클라이언트가 리소스에 대한 접근 권한이 없을 때 처리를 담당하는 필터는 `UsernamePasswordAuthenticationFilter`다.

인증 권한이 없을 때 오류를 JSON으로 내려주기 위해 해당 필터가 실행되기 전 처리가 필요할 것이다.

API 인증 및 권한 부여를 위한 작업 순서는 아래와 같이 구성할 수 있다.

1. 회원 가입, 로그인 API 구현
2. 리소스 접근 가능한 ROLE_USER 권한을 가입 회원에게 부여
3. Spring Security 설정에서 ROLE_USER 권한을 가지면 접근 가능하도록 세팅
4. 권한이 있는 회원이 로그인 성공하면 리소스 접근 가능한 JWT 토큰 발급
5. 해당 회원은 권한이 필요한 API 접근 시 JWT 보안 토큰을 사용

이처럼 접근 제한이 필요한 API에는 보안 토큰을 통해서 이 유저가 권한이 있는지 여부를 Spring Security를 통해 체크하고 리소스를 요청할 수 있도록 구성할 수 있다.

## Spring Security Configuration

서버에 보안을 설정하기 위해 Configuration을 만든다. 기존 예시처럼, USER에 대한 권한을 설정하기 위한 작업도 여기서 진행된다.

```
@Override
    protected void configure(HttpSecurity http) throws Exception {
        http
                .httpBasic().disable() // rest api 이므로 기본설정 사용안함. 기본설
정은 비인증시 로그인폼 화면으로 리다이렉트
                .cors().configurationSource(corsConfigurationSource())
                .and()
                .csrf().disable() // rest api이므로 csrf 보안이 필요없으므로 disable
처리.

.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS) // jwt
token으로 인증하므로 세션은 필요없으므로 생성안함.
                .and()
                .authorizeRequests() // 다음 리퀘스트에 대한 사용권한 체크
                .antMatchers("/*/signin", "/*/signin/**", "/*/signup",
"/*/signup/**", "/social/**").permitAll() // 가입 및 인증 주소는 누구나 접근가능
                .antMatchers(HttpMethod.GET, "home/**").permitAll() // home으로 시
작하는 GET요청 리소스는 누구나 접근가능
                .anyRequest().hasRole("USER") // 그외 나머지 요청은 모두 인증된 회원
만 접근 가능
                .and()
                .addFilterBefore(new JwtAuthenticationFilter(jwtTokenProvider),
UsernamePasswordAuthenticationFilter.class); // jwt token 필터를 id/password 인증
필터 전에 넣는다

    }
```

**[참고 자료]**

- 링크
- 링크
- 링크