JWT 인증과 권한을 이용한 웹기반 채팅

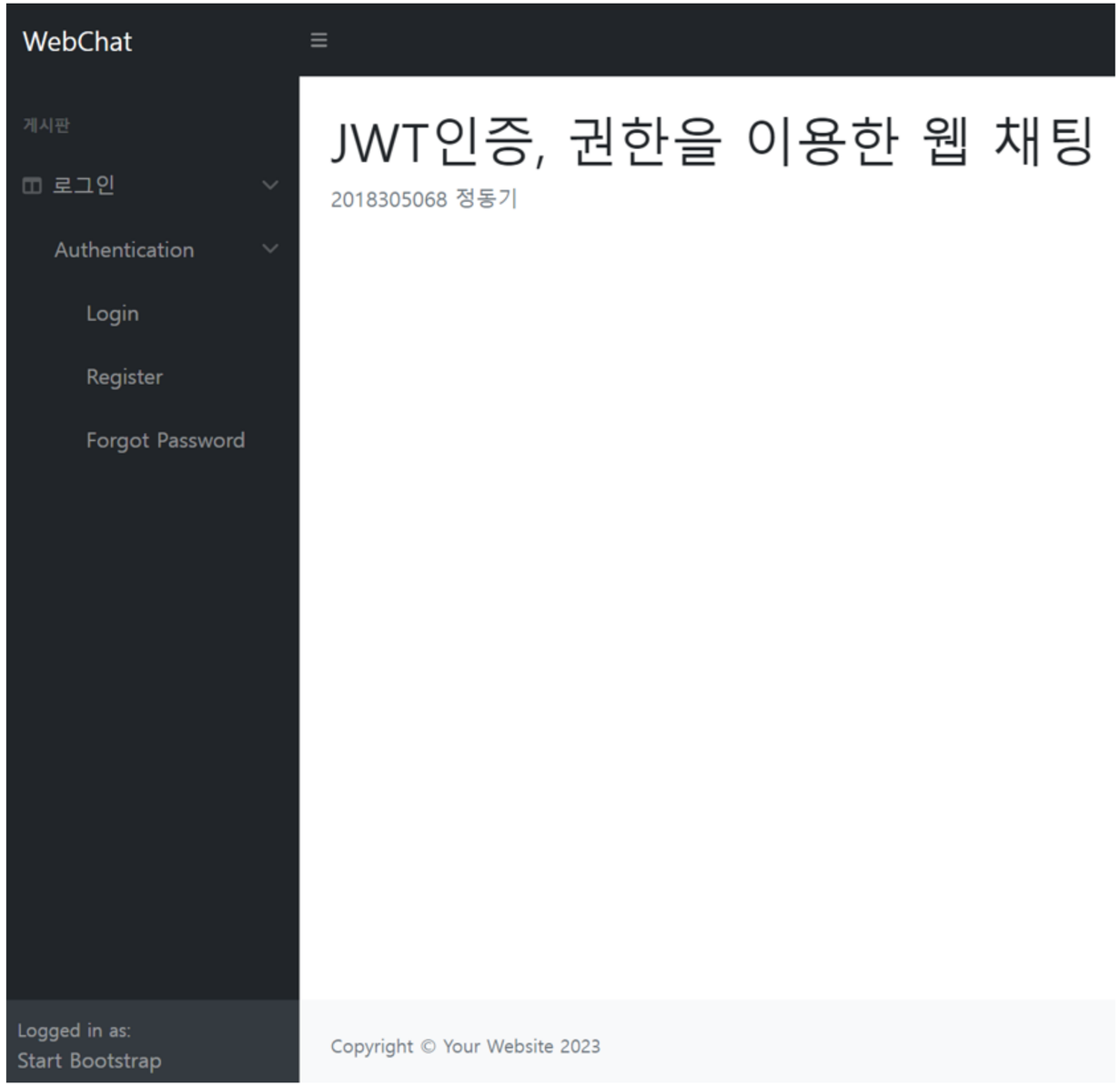
2018305068 정동기

작품 설명

JWT (JSON Web Token) 인증을 통한 게시판과 채팅 기능을 사용할 수 있는 사이트입니다. 사용자는 JWT를 가지고 HTTP 통신 시에 인증에 성공하면 사이트에 접근이 가능하게 됩니다. 또한 인증 후 자신의 권한에 따라 접근할 수 있는 기능이 달라집니다.

목표

JWT를 사용하여 데이터베이스와의 통신을 최소화 시켜 부하를 줄일 수 있으며, 사용자 인증 정보의 보안을 강화합니다. 인증 정보에 있는 권한에 따라서 사용자의 사이트 접근 범위가 달라지며, 권한이 높은 사용자는 권한이 낮은 사용자를 관리하는 방법을 통해 보안을 강화합니다.



개발환경

Front-End

- Jquery
- AJAX
- Socket.js
- Stomp.js

Back-End

- Spring Boot
  - JPA
  - security
  - JWT
  - WebSocket
  - Stomp
  - mail
  - QueryDSL
  - MySQL
  - Thymeleaf

JWT

JSON 웹 토큰이란 무엇입니까?

JWT(JSON 웹 토큰)는 당사자 간에 정보를 JSON 개체로 안전하게 전송하기 위한 간결하고 독립적인 방법을 정의하는 개방형 표준(RFC 7519)입니다. 이 정보는 디지털 서명이 되어 있으므로 확인하고 신뢰할 수 있습니다. JWT는 비밀(**HMAC 알고리즘 사용)** 또는 **RSA** 또는 **ECDSA**를 사용하는 공개/개인 키 쌍을 사용하여 서명할 수 있습니다 .

JWT를 암호화하여 당사자 간 보안을 제공할 수도 있지만 우리는 *서명된* 토큰에 중점을 둘 것입니다. 서명된 토큰은 그 안에 포함된 청구의 *무결성을* 확인할 수 있는 반면, 암호

JWT – Header, Payload, Signature

Header

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

alg : 암호화 알고리즘

typ: 토큰의 타입

Payload

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

정보
- 등록된 클레임
sub : id
Name : 이름
- 개인 클레임
admin: 권한

Signature

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

Header
payload
Secret : 시크릿 키

JWT – Header, Payload, Signature

JWT

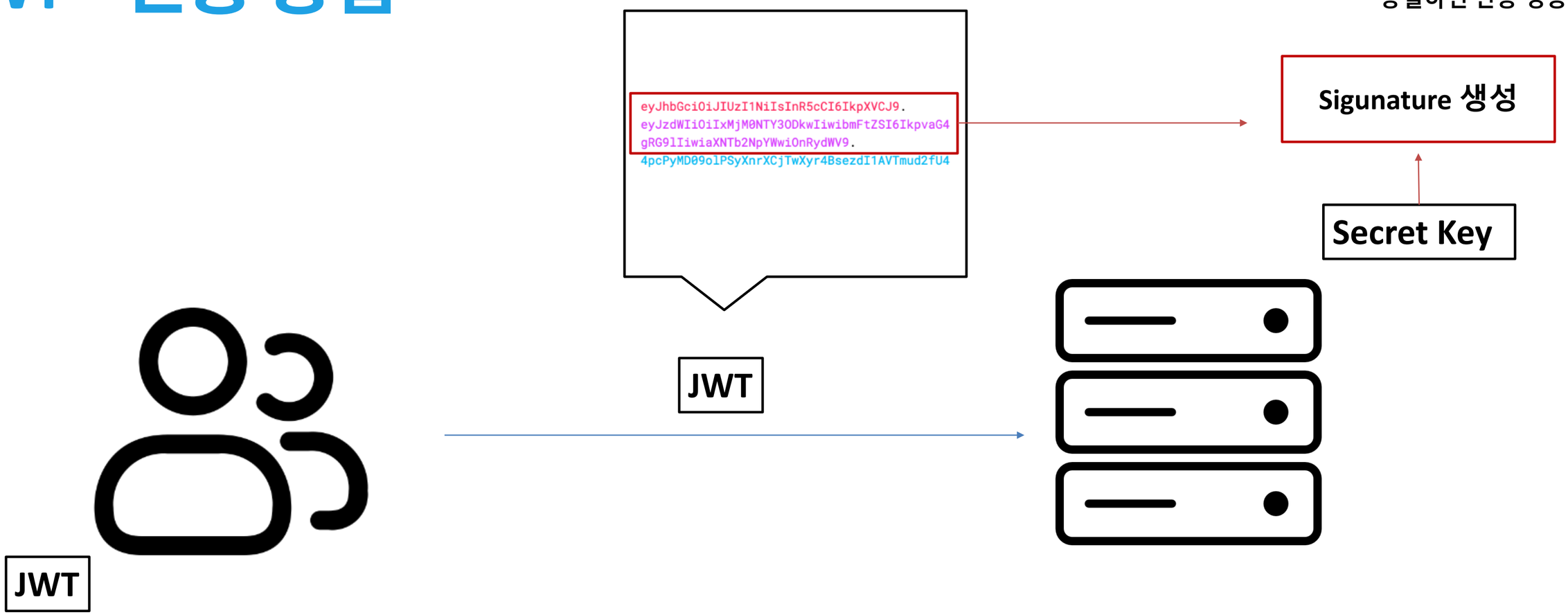
Header: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.

Payload: eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaXNTb2NpYWwiOnRydWV9.

Signature: 4pcPyMD09olPSyXnrXCjTwXyr4BsezdI1AVTmud2fU4

JWT – 인증 방법

동일하면 인증 성공



JWT – 장점

01

인증 정보의 상태를 유지, 갱신할 필요가 없다.

DB부하를 막는다.

02

Secretkey만 있다면 어디서든 접근이 가능하다.

서버에서 secretkey만 가지고 있으면 서버 확장을 쉽게 할 수 있다.

03

HTTP Body값이 탈취되어도 Token이 없으면 사용 못한다.

accessToken이 없다면 인증이 불가능하다.

04

서명으로 토큰 변경 여부를 알 수 있다.

서명은 헤더와 페이로드로 생성하기 때문에 변경 여부를 바로 알 수 있다.

JWT - 단점 및 보완

토큰 탈취

01

HttpOnly 속성 추가

JS를 통한 접근을 막는다.

02

accessToken의 유효기간을 줄였다.

유효기간을 줄임으로써 탈취 당하더라도 사용을 오래 못하도록 하였다.

03

refreshToken을 이용하여 accessToken을 발급 받는다.

accessToken이 없을 때 refreshToken으로 발급 받는다.

04

Rotating Refresh Token

RefreshToken이 탈취 당할 수 도 있기 때문에 사용하면 재발급 받는다.

주요 기능

로그인

회원가입
로그인
비밀번호 찾기

JWT 인증

HTTP 통신 시 JWT 인증

사용자 관리

사용자 권한 부여 및
삭제
내 정보
상대방 정보

게시판

게시글 작성
게시글 수정
게시글 페이징
게시글 찾기
댓글-대댓글(수정, 삭제)

파일

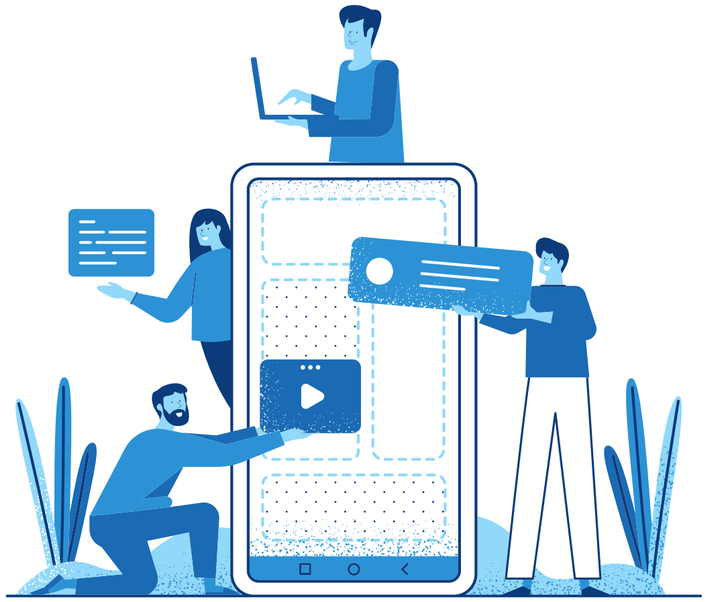
파일 저장
파일 삭제

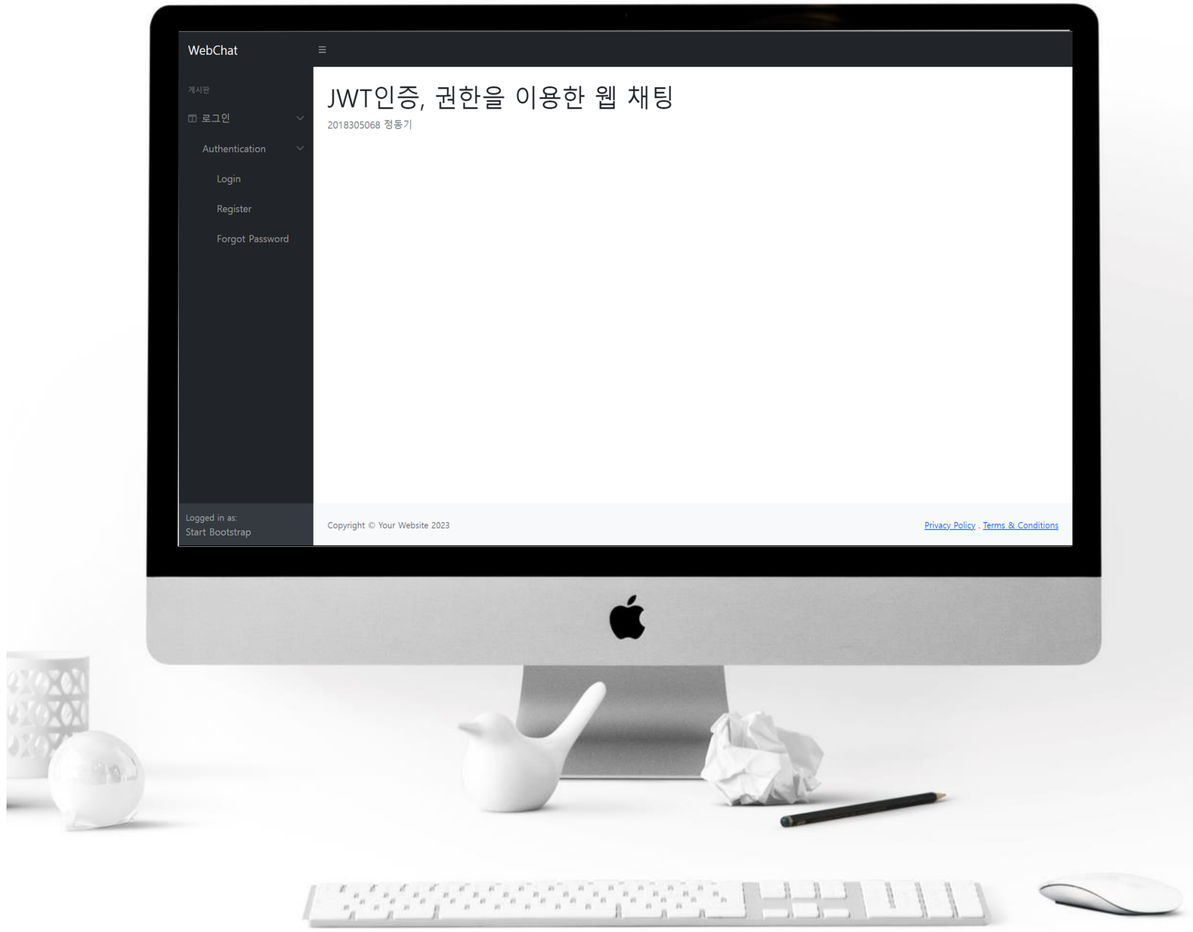
채팅

공개 채팅
개인 채팅

로그인

- 회원가입
- 로그인
- 비밀번호 찾기





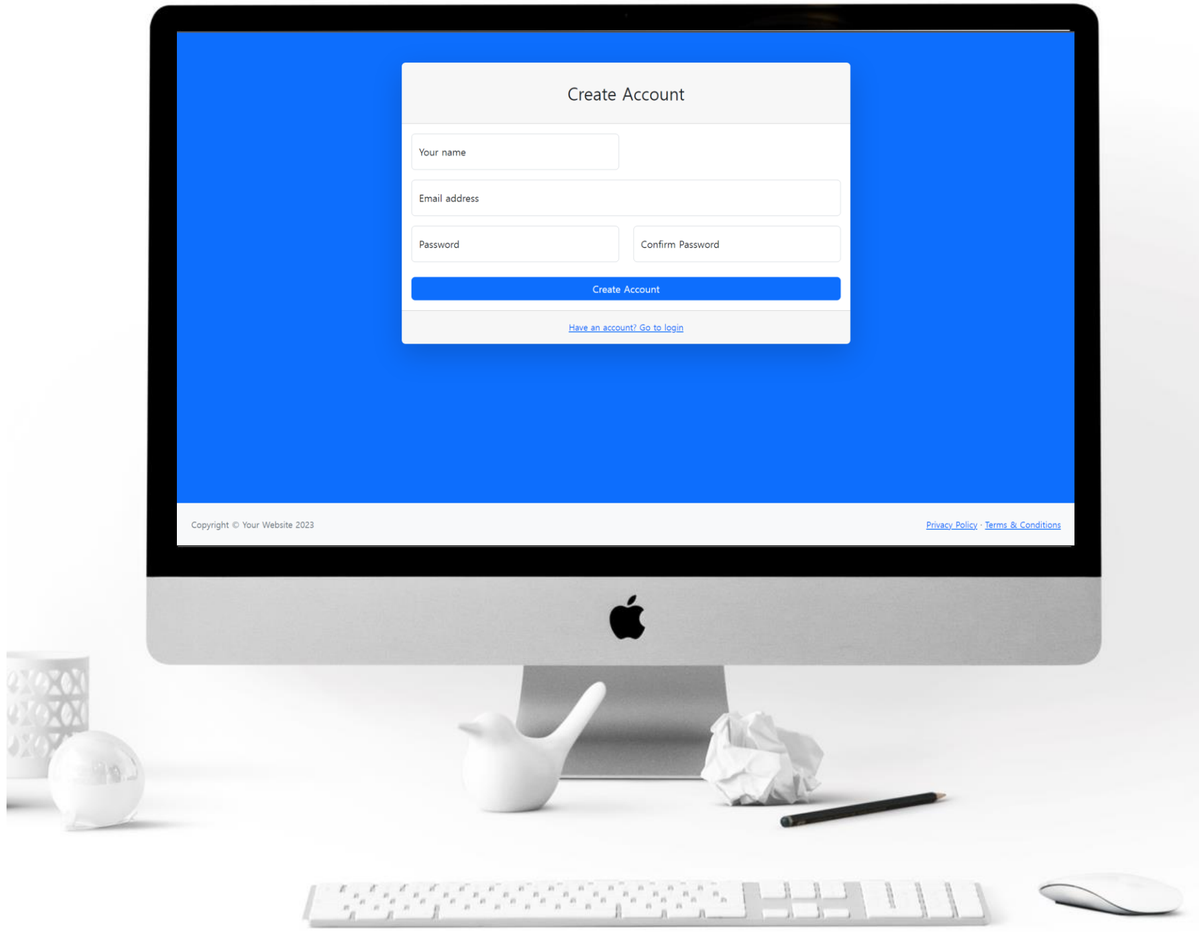
로그인 - 회원가입

1. 비밀번호 인코딩 - bcrypt

- Bcrypt : SHA-256 해시 알고리즘 기반으로 “salt”를 추가하여 같은 비밀번호라도 다른 결과를 생성한다.

장점: 복호화 불가능 – 해킹당하더라도 안전

- 첫 회원가입 시에는 권한이 GUEST



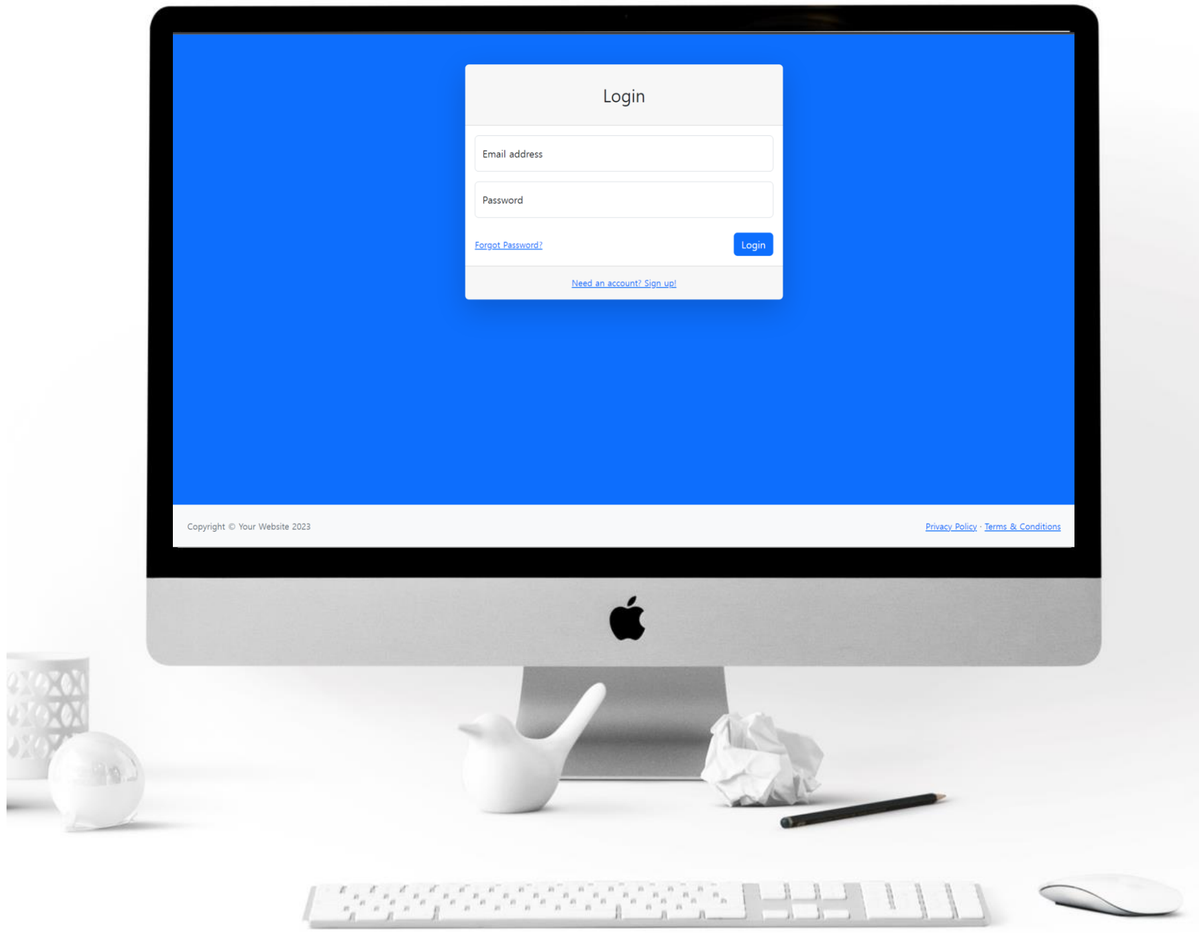
로그인 - 로그인

1. POST방식으로 들어오는 /login 요청 처리

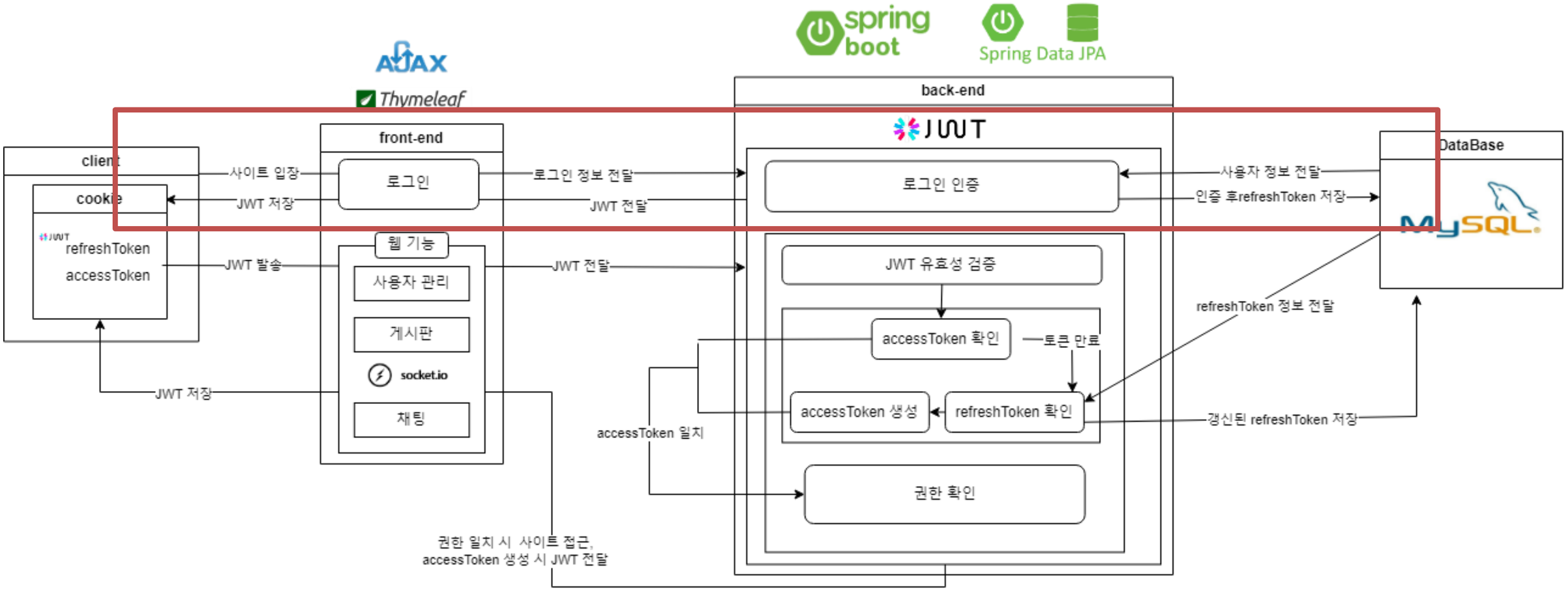
1) AbstractAuthenticationProcessingFilter 확장
2) 이메일과 비밀번호 추출 후 사용자 인증
   - 성공시 SuccessHanlder, 실패시 FailureHandler

2. JWT 생성 및 권한 설정

- SuccessHandler에 접근 시 JWT 생성 후 클라이언트에게 보내주기 [accessToken, refreshToken]



로그인

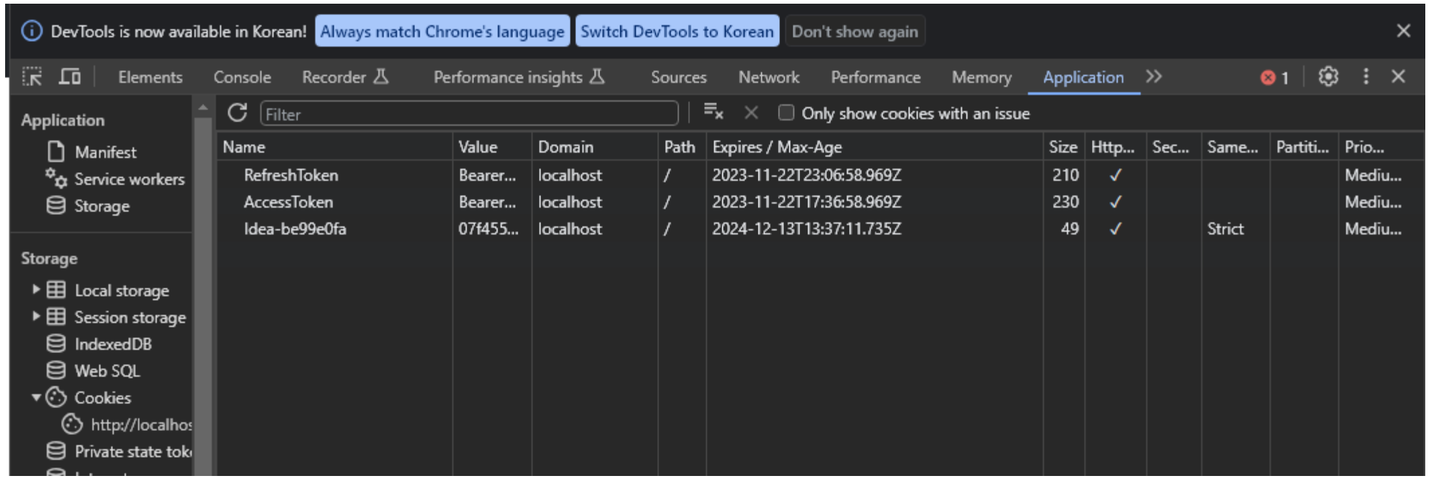

AJAX
Thymeleaf
spring boot
Spring Data JPA
back-end
JWT
front-end
client
cookie
refreshToken
accessToken
로그인
웹 기능
사용자 관리
게시판
socket.io
채팅
로그인 인증
JWT 유효성 검증
accessToken 확인
토큰 만료
accessToken 생성
refreshToken 확인
권한 확인
DataBase
MySQL
사이트 입장
JWT 저장
로그인 정보 전달
JWT 전달
JWT 발송
JWT 전달
JWT 저장
사용자 정보 전달
인증 후refreshToken 저장
refreshToken 정보 전달
갱신된 refreshToken 저장
accessToken 일치
권한 일치 시 사이트 접근,
accessToken 생성 시 JWT 전달

로그인

MySQL

createdDate	lastModifiedDate	user_id	nickname	password	refreshToken	role	username
2023-11-1...	2023-11-23 01:...	1	정동기	{bcrypt}$...	Bearer eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJSZWZyZXNoVG9rZW4iLCJleHAiOjE3MDA2OTIyMzYsInVzZXJJZCI6MX0.CGJrP...	ADMIN	dkrhank45@gmail.com
2023-11-1...	2023-11-23 02:...	2	사람인	{bcrypt}$...	Bearer eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJSZWZyZXNoVG9rZW4iLCJleHAiOjE3MDA2OTQ0MTgsInVzZXJJZCI6Mn0.ALzTe...	MANAGER	test2@test.com
2023-11-1...	2023-11-22 11:...	3	에잉	{bcrypt}$...	Bearer eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJSZWZyZXNoVG9rZW4iLCJleHAiOjE3MDA2NDA1NTAsInVzZXJJZCI6M30.xXTHt...	USER	test3@test.com
2023-11-1...	2023-11-22 18:...	4	as	{bcrypt}$...	Bearer eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJSZWZyZXNoVG9rZW4iLCJleHAiOjE3MDA2NDA1NjQsInVzZXJJZCI6NH0.0euJA...	SEMIUSER	test4@test.com
2023-11-2...	2023-11-22 11:...	5	안녕	{bcrypt}$...	Bearer eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJSZWZyZXNoVG9rZW4iLCJleHAiOjE3MDA2NDA2NDgsInVzZXJJZCI6NX0._1SaG...	GUEST	test@test.com
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Cookie



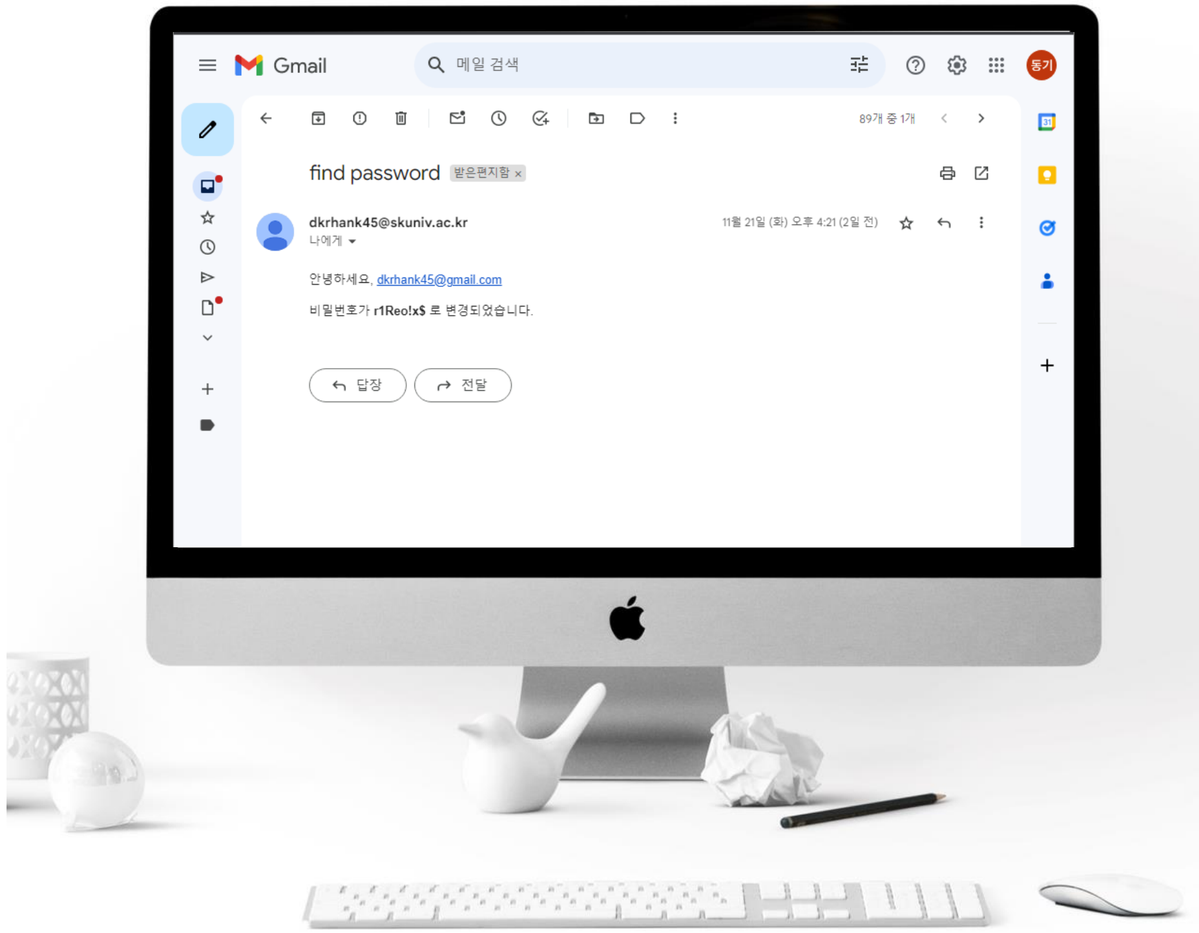
로그인 – 비밀번호 찾기

1. 정규식 사용

1) 비밀번호

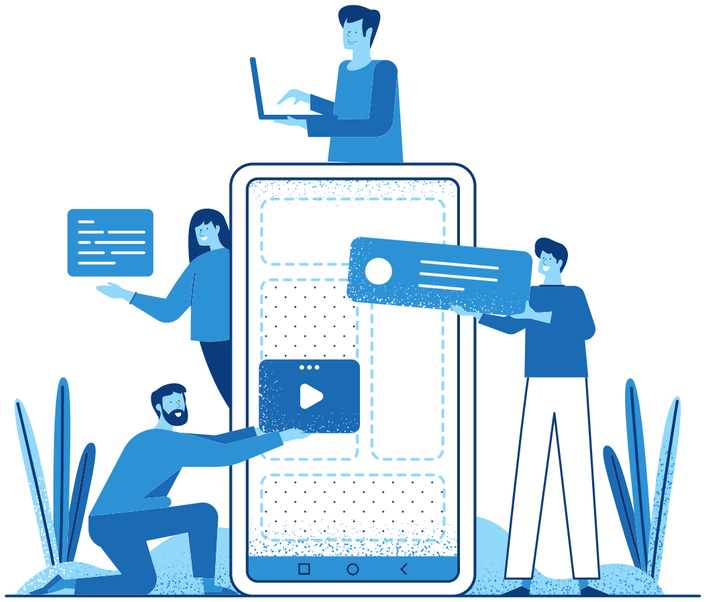
2. 이메일 수신

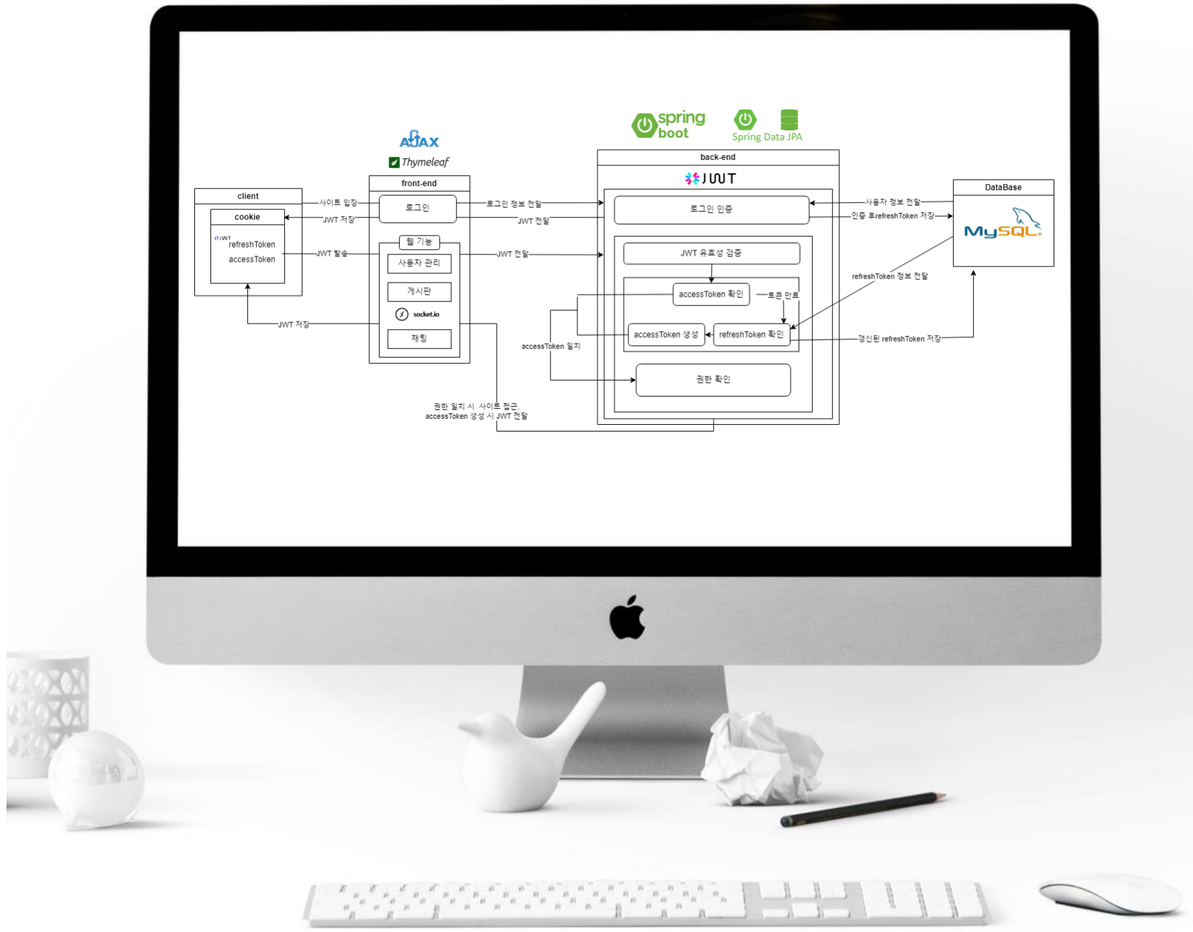
1) 비밀번호 복호화가 불가능 하므로 정규식에 맞는 랜덤한 비밀번호 생성 후 보내주기



JWT 인증

- JWT 토큰 발급 및 인증





JWT 인증

1. **접근 시 인증이 필요없는 URL과 정적 리소스 인증 건너뛰기**

2. **엑세스 토큰이 있는 경우 (리프레시 토큰도 있는 경우 포함)**

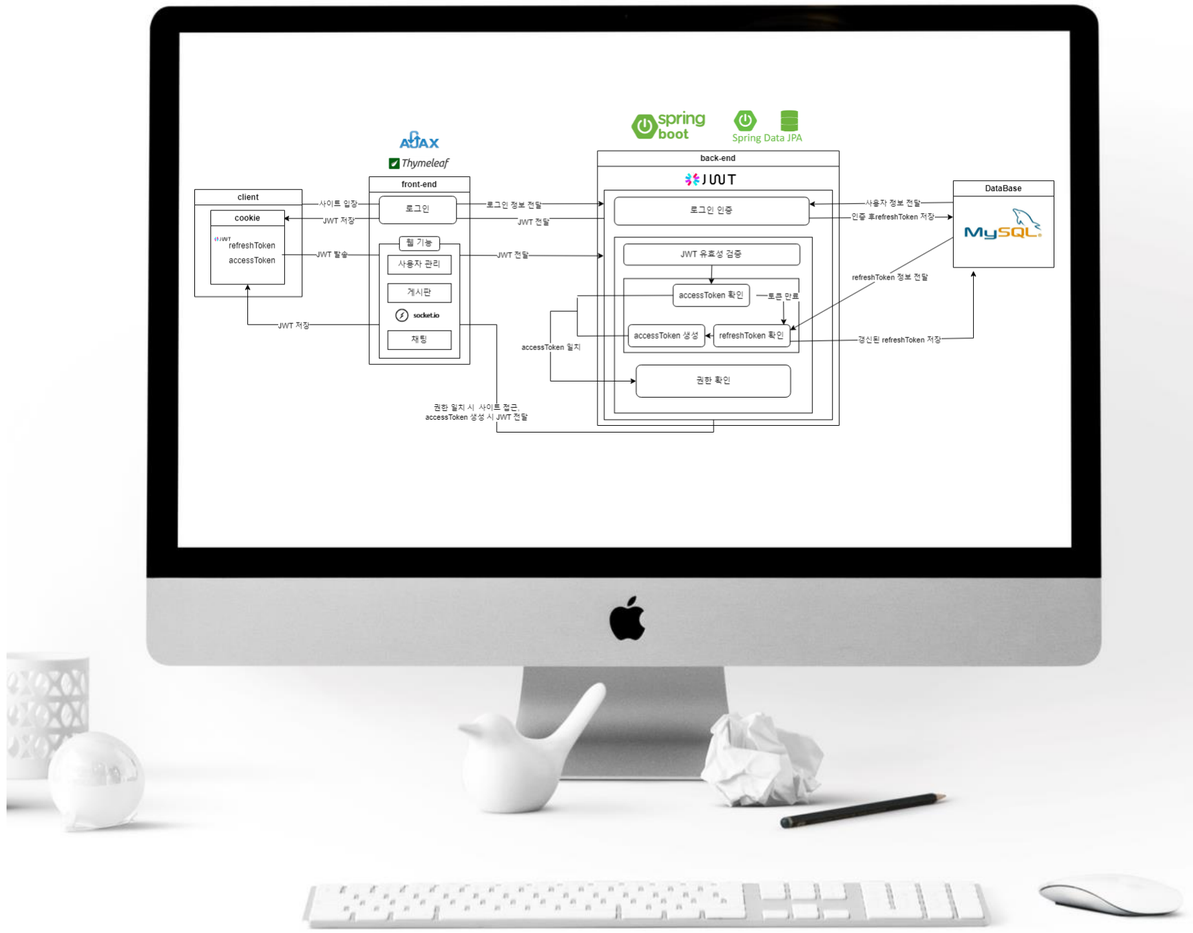
    1) 엑세스 토큰이 맞는지 확인 후 인증

3. **리프레시 토큰만 있는경우**

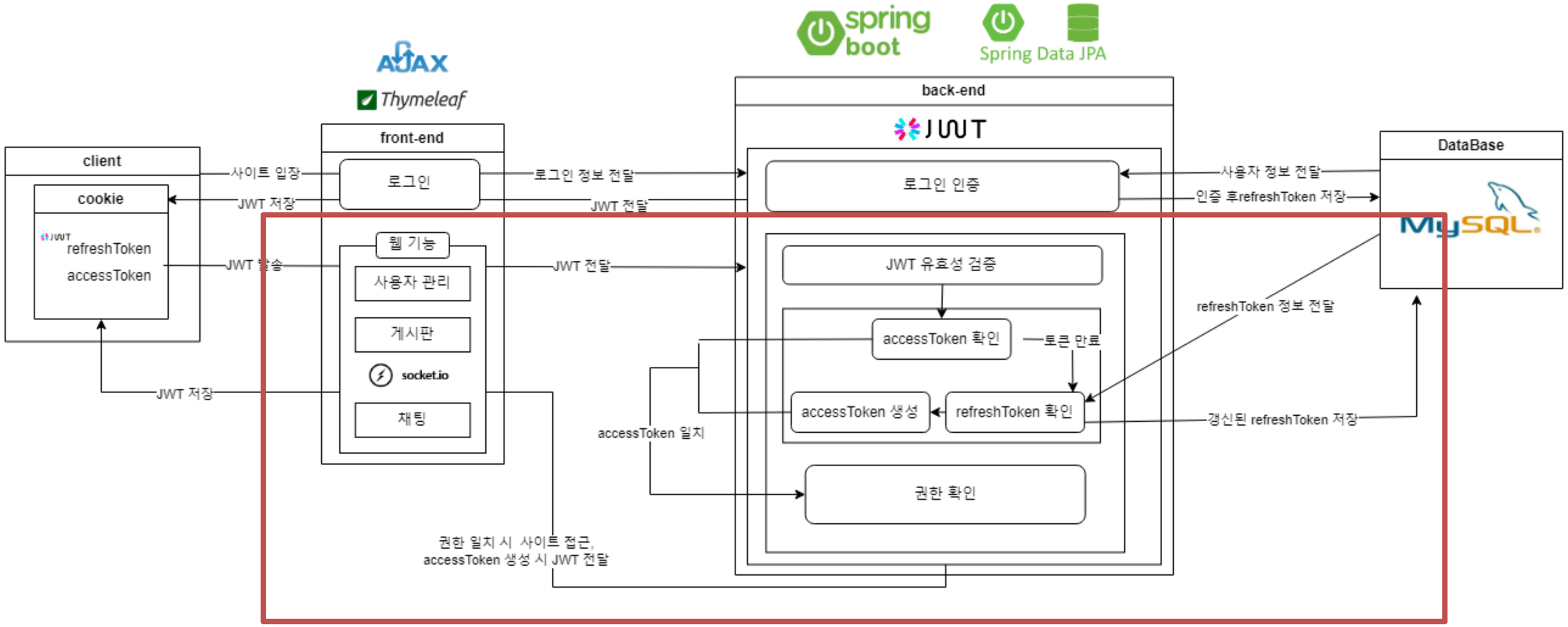
    1) 리프레시 토큰이 유효하다면
    2) 엑세스 토큰과 리프레시 토큰을 생성 후
    3) 리프레시 토큰을 DB에 저장
    4) 클라이언트에게 엑세스 토큰과 리프레시 토큰 보내주기

4. **둘 다 없는 경우**

    - 엑세스 거부

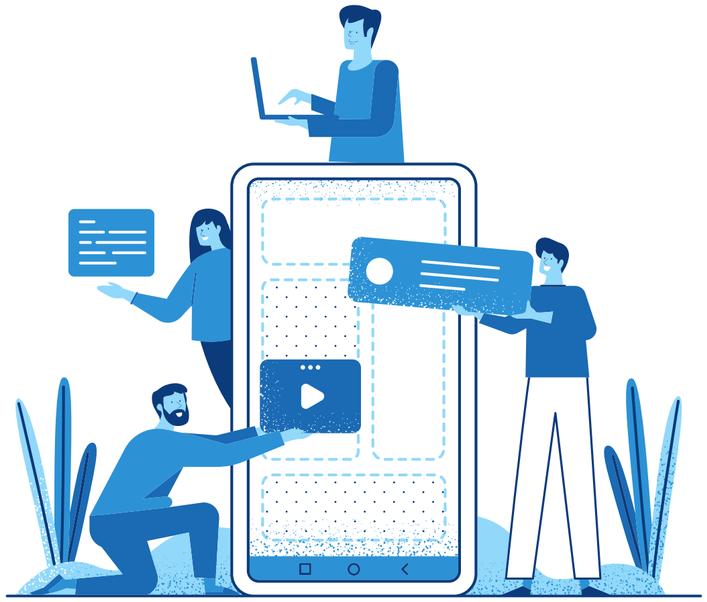


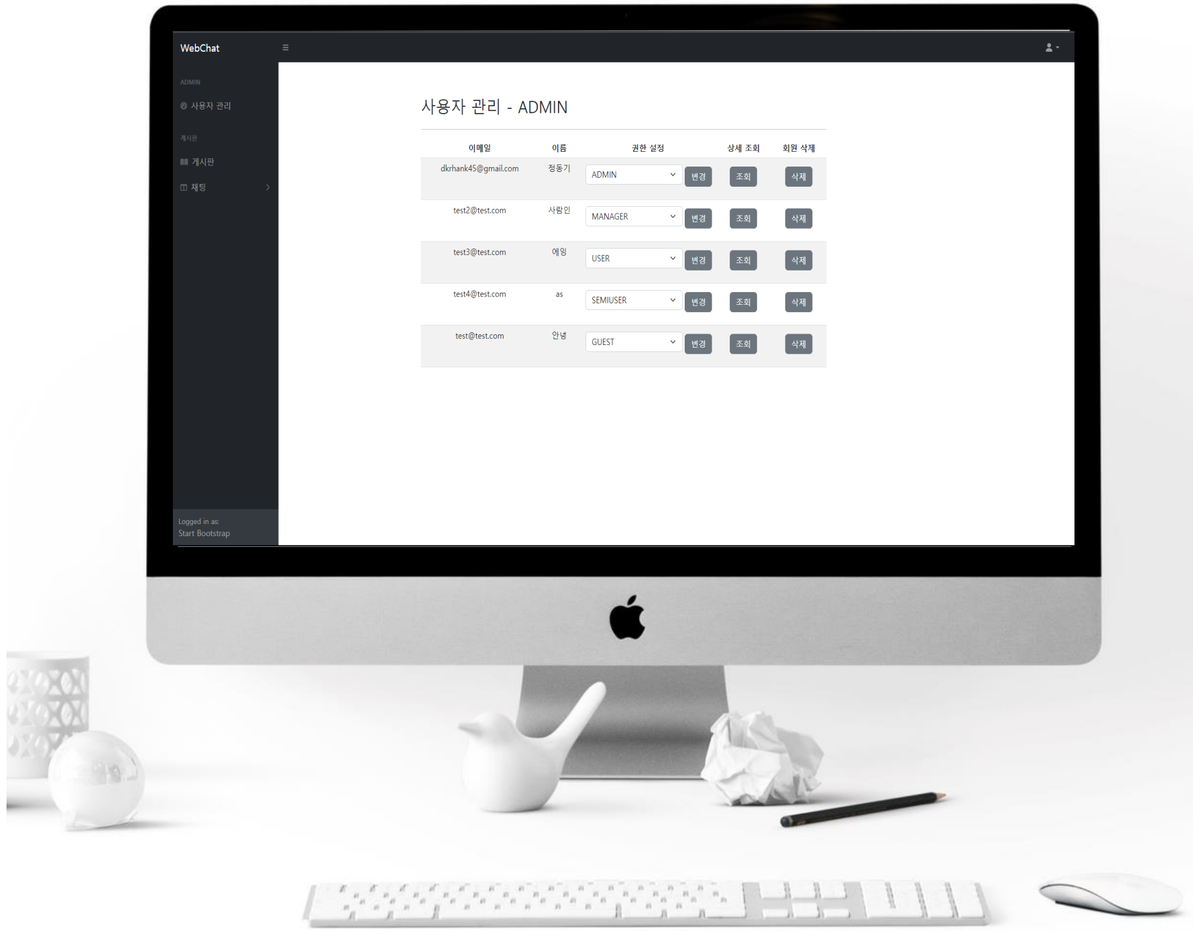
JWT 인증


AJAX
Thymeleaf
spring boot
Spring Data JPA
front-end
back-end
JWT
client
cookie
refreshToken
accessToken
사이트 입장
JWT 저장
JWT 발송
로그인
로그인 정보 전달
JWT 전달
웹 기능
사용자 관리
게시판
socket.io
채팅
로그인 인증
JWT 유효성 검증
accessToken 확인
토큰 만료
accessToken 생성
refreshToken 확인
권한 확인
accessToken 일치
권한 일치 시 사이트 접근,
accessToken 생성 시 JWT 전달
DataBase
MySQL
사용자 정보 전달
인증 후refreshToken 저장
refreshToken 정보 전달
갱신된 refreshToken 저장

사용자 관리

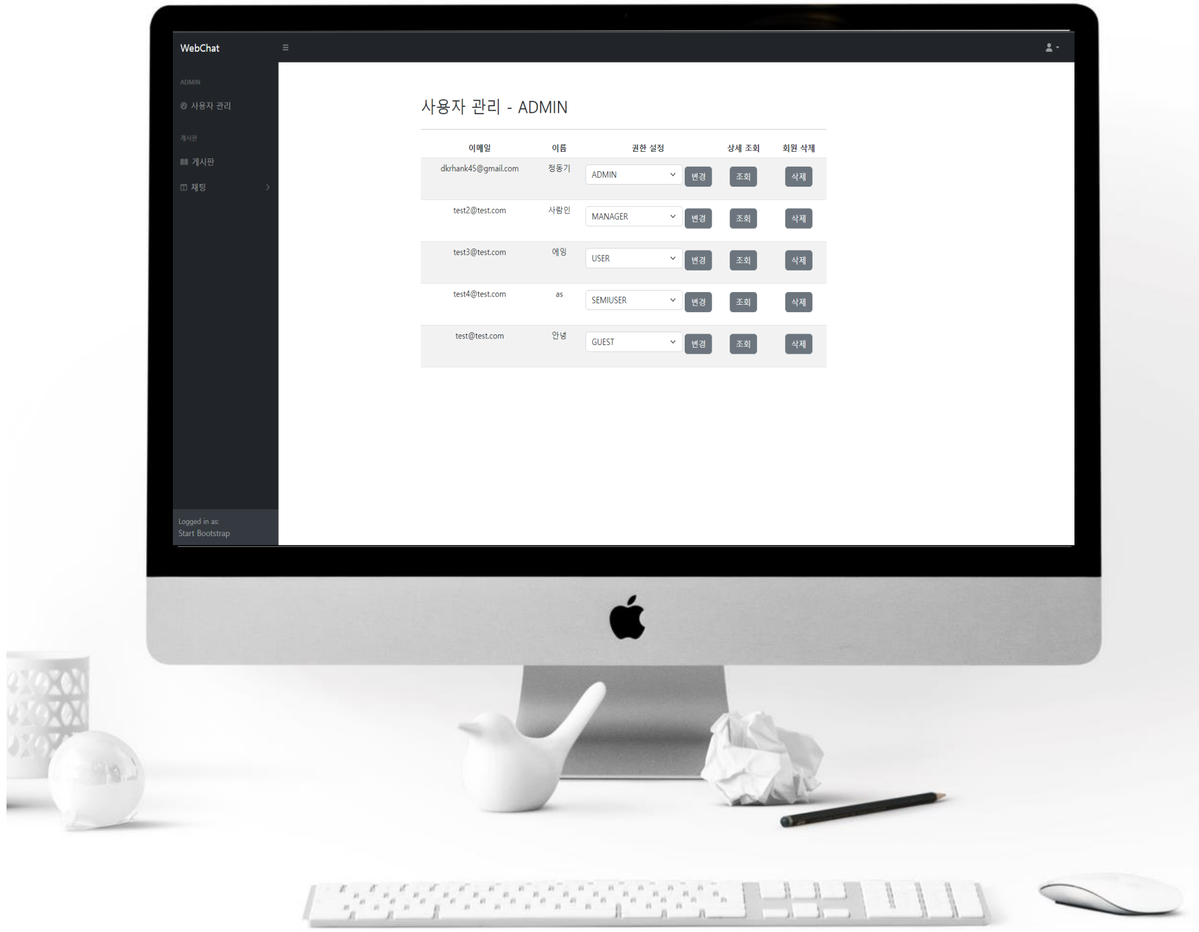
- ADMIN
- MANAGER
- 내 정보
- 사용자 정보





사용자 관리 - ADMIN

- 사용자 권한 설정
- 회원 삭제



사용자 관리 - MANAGER

- 사용자 권한 설정
- ADMIN 포함 안함

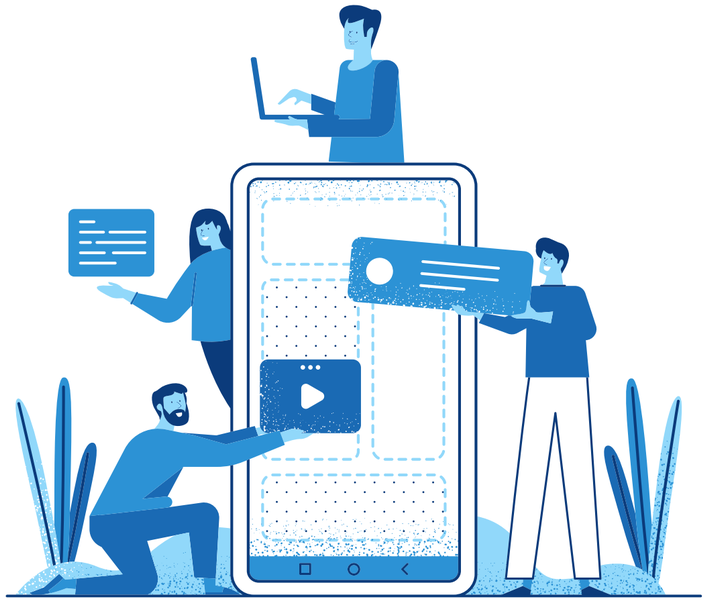
사용자 관리 – 내 정보

- **사용자 정보**

1. **비밀번호 변경**
   1) 현재 비밀번호
   2) 변경하고 싶은 비밀번호
2. **이름 변경**
   1) 변경하고 싶은 이름
3. **회원 탈퇴**
   1) 현재 비밀번호
4. **어드민 변경**
   1) 어드민 키

채팅

- Socket
- Stomp



```java
@Configuration
@EnableWebSocketMessageBroker
@RequiredArgsConstructor
public class WebSocketConfig implements WebSocketMessageBrokerConfigurer {

    1 usage
    @Override
    public void configureMessageBroker(MessageBrokerRegistry registry){
        registry.enableSimpleBroker( ...destinationPrefixes: "queue","/sub"); // 받는 사람
        registry.setApplicationDestinationPrefixes("/pub"); // 보내는 사람
    }

    1 usage
    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry){
        registry.addEndpoint( ...paths: "/ws")
                .setAllowedOrigins("http://localhost:8080")
                .withSockJS();
    }

    // STOMP에서 64KB 이상의 데이터 전송을 못하는 문제 해결
    1 usage
    @Override
    public void configureWebSocketTransport(WebSocketTransportRegistration registry) {
        registry.setMessageSizeLimit(160 * 64 * 1024);
        registry.setSendTimeLimit(100 * 10000);
        registry.setSendBufferSizeLimit(3 * 512 * 1024);
    }
}
```

채팅 – STOMP

```java
@Configuration
@EnableWebSocketMessageBroker
@RequiredArgsConstructor
public class WebSocketConfig implements WebSocketMessageBrokerConfigurer {

    1 usage
    @Override
    public void configureMessageBroker(MessageBrokerRegistry registry){
        registry.enableSimpleBroker( ...destinationPrefixes: "queue","/sub"); // 받는 사람
        registry.setApplicationDestinationPrefixes("/pub"); // 보내는 사람
    }

    1 usage
    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry){
        registry.addEndpoint( ...paths: "/ws")
                .setAllowedOrigins("http://localhost:8080")
                .withSockJS();
    }

    // STOMP에서 64KB 이상의 데이터 전송을 못하는 문제 해결
    1 usage
    @Override
    public void configureWebSocketTransport(WebSocketTransportRegistration registry) {
        registry.setMessageSizeLimit(160 * 64 * 1024);
        registry.setSendTimeLimit(100 * 10000);
        registry.setSendBufferSizeLimit(3 * 512 * 1024);
    }
}
```

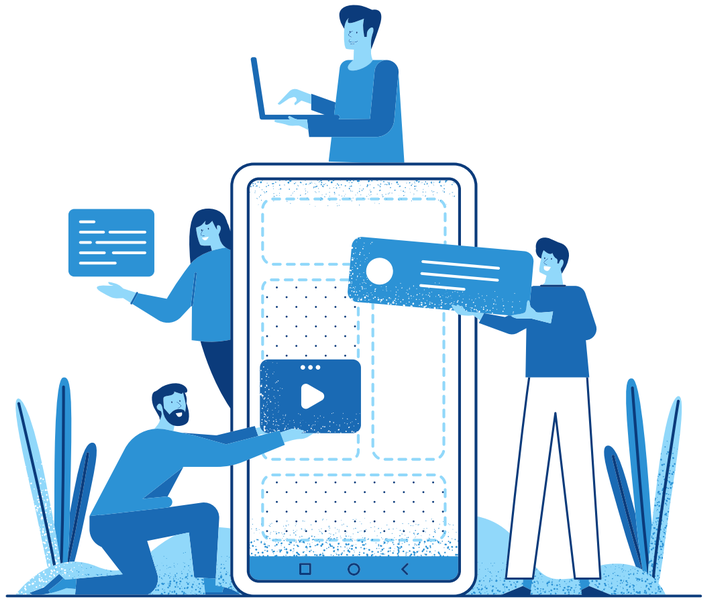
Queue:, Sub: 서버가 클라이언트에게
Pub: 클라이언트가 서버에게

Ws: 경로를 통해 웹소켓 서버에 연결
http://localhost:8080: 접근 가능한 클라이언트 출처
withSockJS : 웹소켓을 지원하지 않는 브라우저에서도 유사한 기능 사용 가능

메시지 최대 크기 지정

권한 설정

- 게시판 및 파일, 댓글
- 채팅



```
@Bean
public SecurityFilterChain filterChain(HttpSecurity http) throws Exception{
    http
            .exceptionHandling(exp ->
                    exp.accessDeniedPage( accessDeniedUrl: "/error"))
            .formLogin(login -> login.disable())
            .httpBasic(basic -> basic.disable())
            .csrf(csrf -> csrf.disable())
            .sessionManagement(session ->
                    session.sessionCreationPolicy(SessionCreationPolicy.STATELESS))
            .authorizeHttpRequests(auth -> // URL별 권한 관리 옵션
                    auth
                            .requestMatchers( ...patterns: "/css/**", "/js/**").permitAll()
                            .requestMatchers( ...patterns: "/", "/signup", "/loginForm", "/password" ,"/favicon.ico", "/error").permitAll()
                            .requestMatchers( ...patterns: "/login", "/signUp").permitAll()
                            .requestMatchers( ...patterns: "/files/**").hasAnyRole( ...roles: "USER", "ADMIN", "MANAGER")
                            .requestMatchers( ...patterns: "/board/**", "/chat/**").hasAnyRole( ...roles: "SEMIUSER","USER", "ADMIN", "MANAGER")
                            .requestMatchers( ...patterns: "/updateauthorization").hasAnyRole( ...roles: "ADMIN", "MANAGER")
                            .requestMatchers( ...patterns: "/user/updateAuth").hasAnyRole( ...roles: "ADMIN", "MANAGER")
                            .anyRequest().authenticated() // 인증된 사용자만 접근 가능
            )
            .addFilterAfter(customUsernamePasswordAuthenticationFilter(), LogoutFilter.class)
            .addFilterBefore(jwtAuthenticationProcessingFilter(), CustomUsernamePasswordAuthenticationFilter.class); // 로그인 후에는 모든 요청이 JWT 필터를 거치게 된다

    return http.build();
}
```

권한 설정

```
@Bean
public SecurityFilterChain filterChain(HttpSecurity http) throws Exception{
    http
            .exceptionHandling(exp ->
                    exp.accessDeniedPage( accessDeniedUrl: "/error"))
            .formLogin(login -> login.disable())
            .httpBasic(basic -> basic.disable())
            .csrf(csrf -> csrf.disable())
            .sessionManagement(session ->
                    session.sessionCreationPolicy(SessionCreationPolicy.STATELESS))
            .authorizeHttpRequests(auth -> // URL별 권한 관리 옵션
                    auth
                            .requestMatchers( ...patterns: "/css/**", "/js/**").permitAll()
                            .requestMatchers( ...patterns: "/", "/signup", "/loginForm", "/password" ,"/favicon.ico", "/error").permitAll()
                            .requestMatchers( ...patterns: "/login", "/signUp").permitAll()
                            .requestMatchers( ...patterns: "/files/**").hasAnyRole( ...roles: "USER", "ADMIN", "MANAGER")
                            .requestMatchers( ...patterns: "/board/**", "/chat/**").hasAnyRole( ...roles: "SEMIUSER","USER", "ADMIN", "MANAGER")
                            .requestMatchers( ...patterns: "/updateauthorization").hasAnyRole( ...roles: "ADMIN", "MANAGER")
                            .requestMatchers( ...patterns: "/user/updateAuth").hasAnyRole( ...roles: "ADMIN", "MANAGER")
                            .anyRequest().authenticated() // 인증된 사용자만 접근 가능
            )
            .addFilterAfter(customUsernamePasswordAuthenticationFilter(), LogoutFilter.class)
            .addFilterBefore(jwtAuthenticationProcessingFilter(), CustomUsernamePasswordAuthenticationFilter.class); // 로그인 후에는 모든 요청이 JWT 필터를 거치게 된다

    return http.build();
}
```

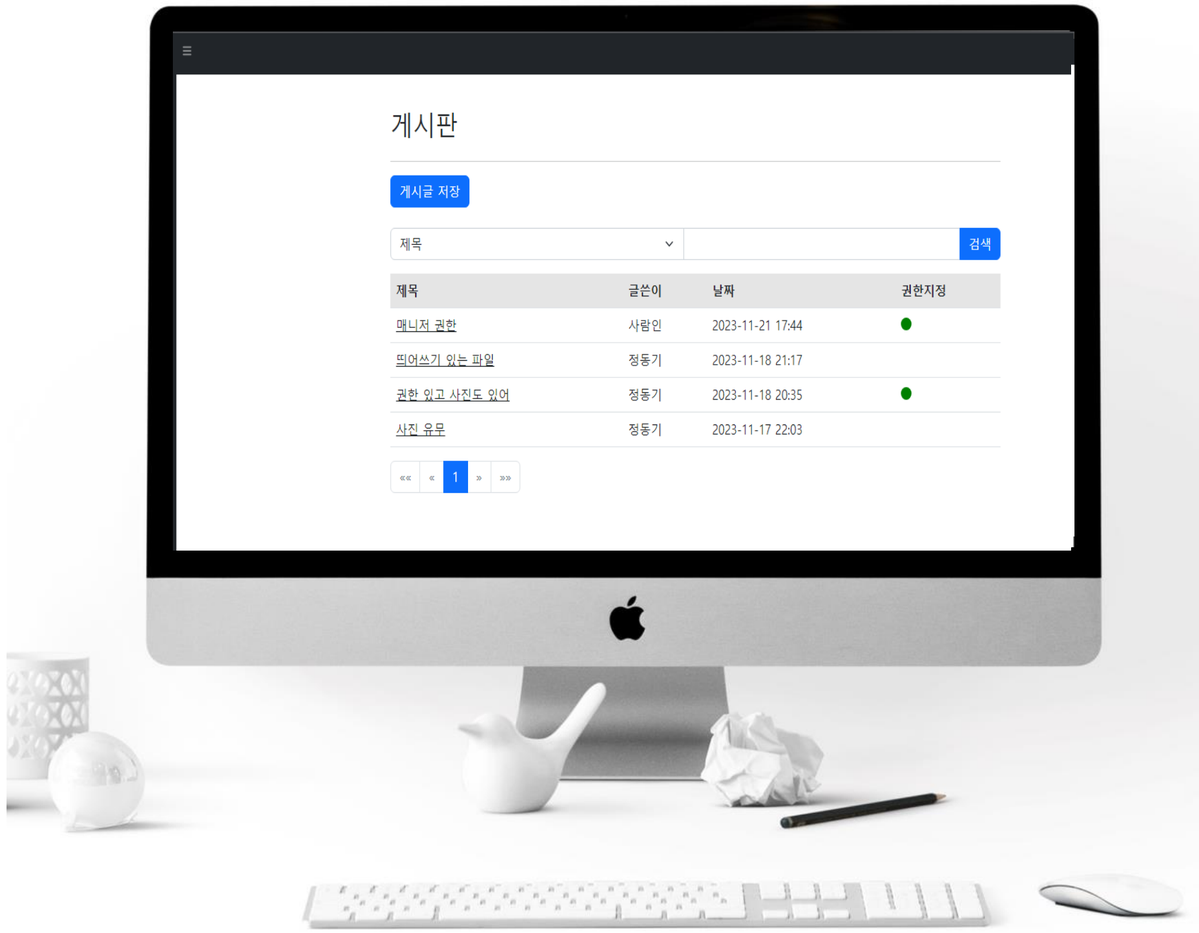
PermitAll() : 인증 불필요 [정적 리소스, 기본화면, 회원가입, 로그인, 비밀번호 찾기, 에러 표시
hasAnyRole : 인증 필요 및 권한 설정 [파일, 게시판, 채팅, 권한 설정]
Authenticated() : 인증 필요 [나머지 URL]

권한 설정 – 게시판 및 파일, 댓글

1. **GUEST는 사용 불가**

2. **게시글 저장**
   1) ADMIN, MANAGER : 자신의 권한 이상만 게시글 읽기 가능
3. **게시글 수정 및 삭제**
   1) 글쓴이만 가능
4. **게시글 읽기**
   1) SEMIUSER는 파일 접근 금지
   2) 댓글과 대댓글(수정 및 삭제) 는 SEMIUSER도 가능
   3) 읽기 권한이 있다면 작성자 이상의 권한만 읽기 가능
5. **게시글 페이징**

6. **게시글 서치**
   1) QueryDSL을 이용하여 제목, 내용, 제목+내용 으로 검색 가능

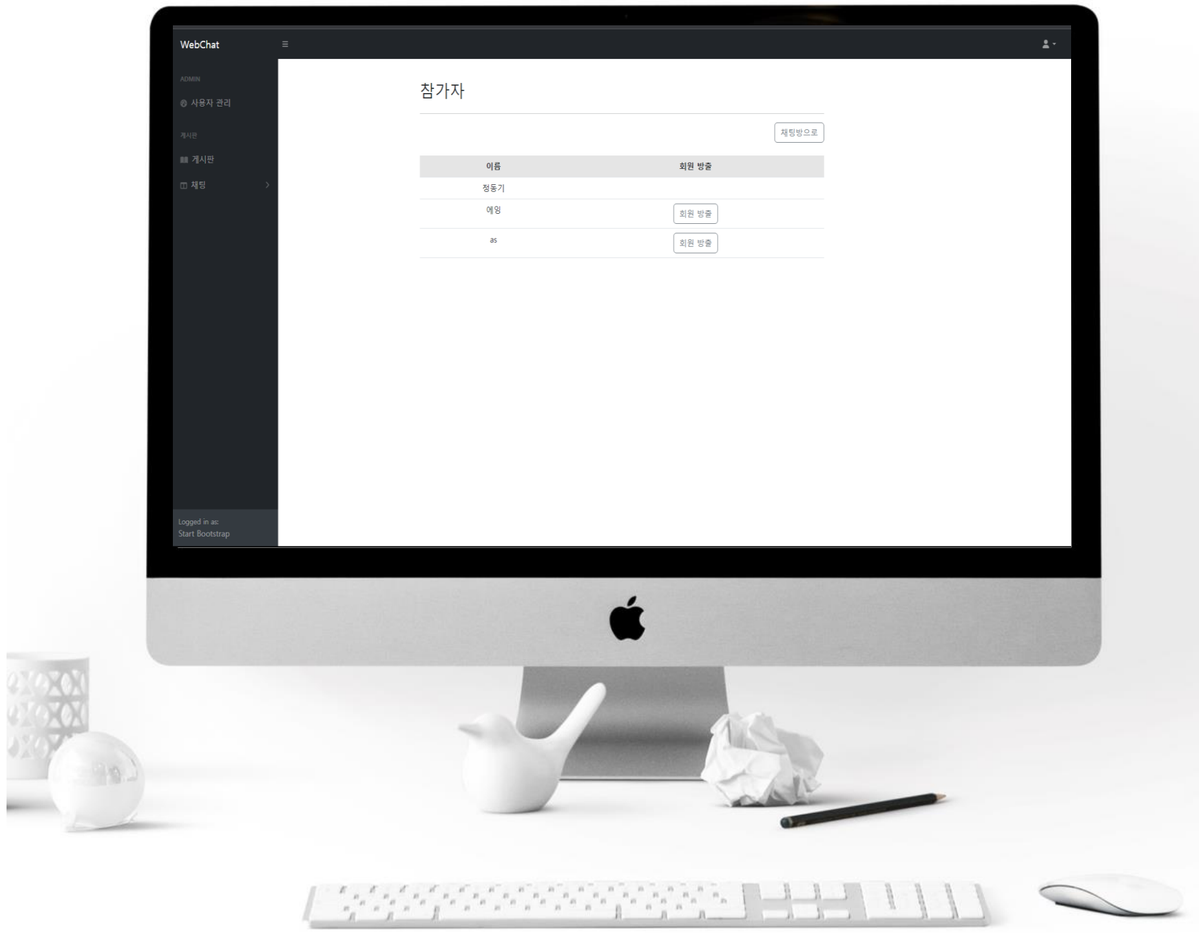


권한 설정 – 채팅

1. GUEST는 사용 불가

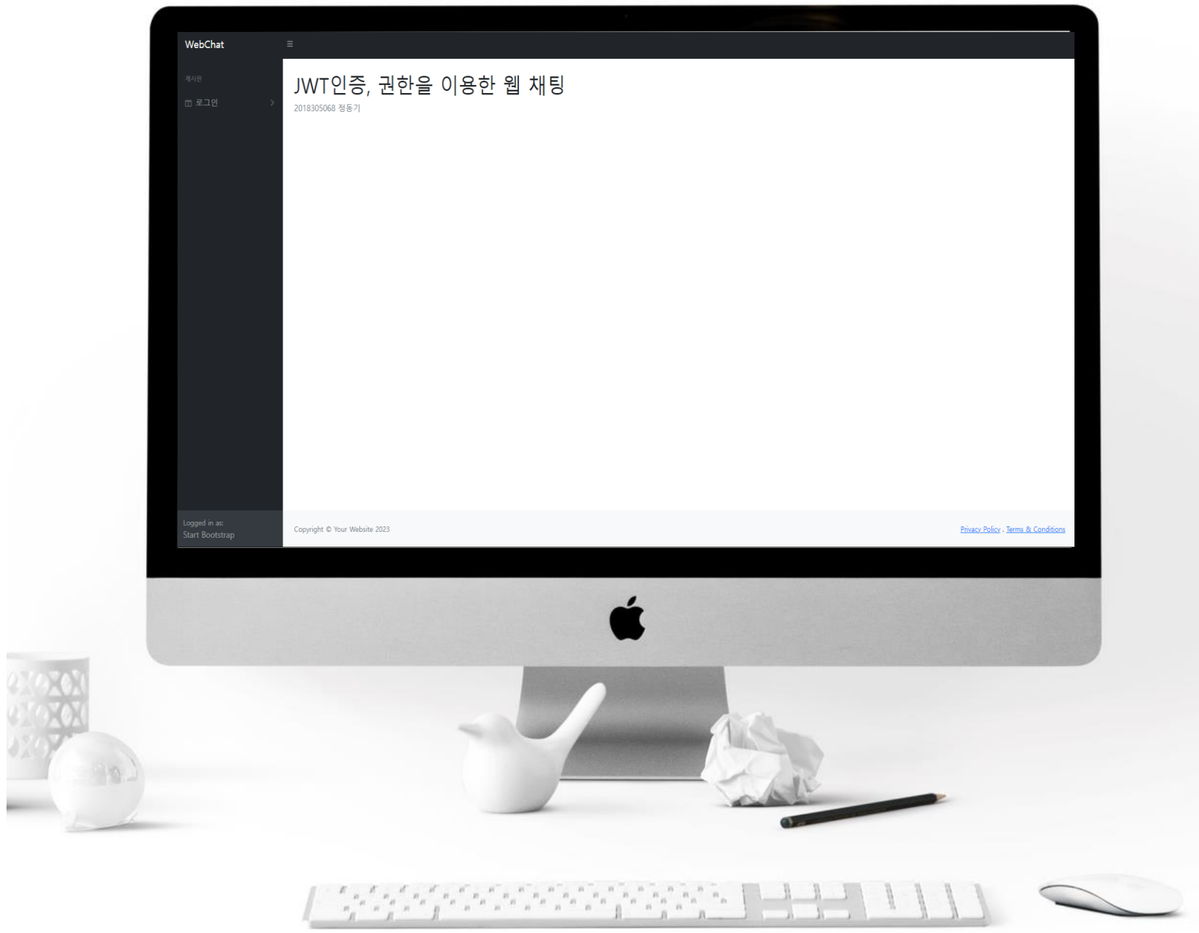
2. 공개 채팅과 비공개 채팅

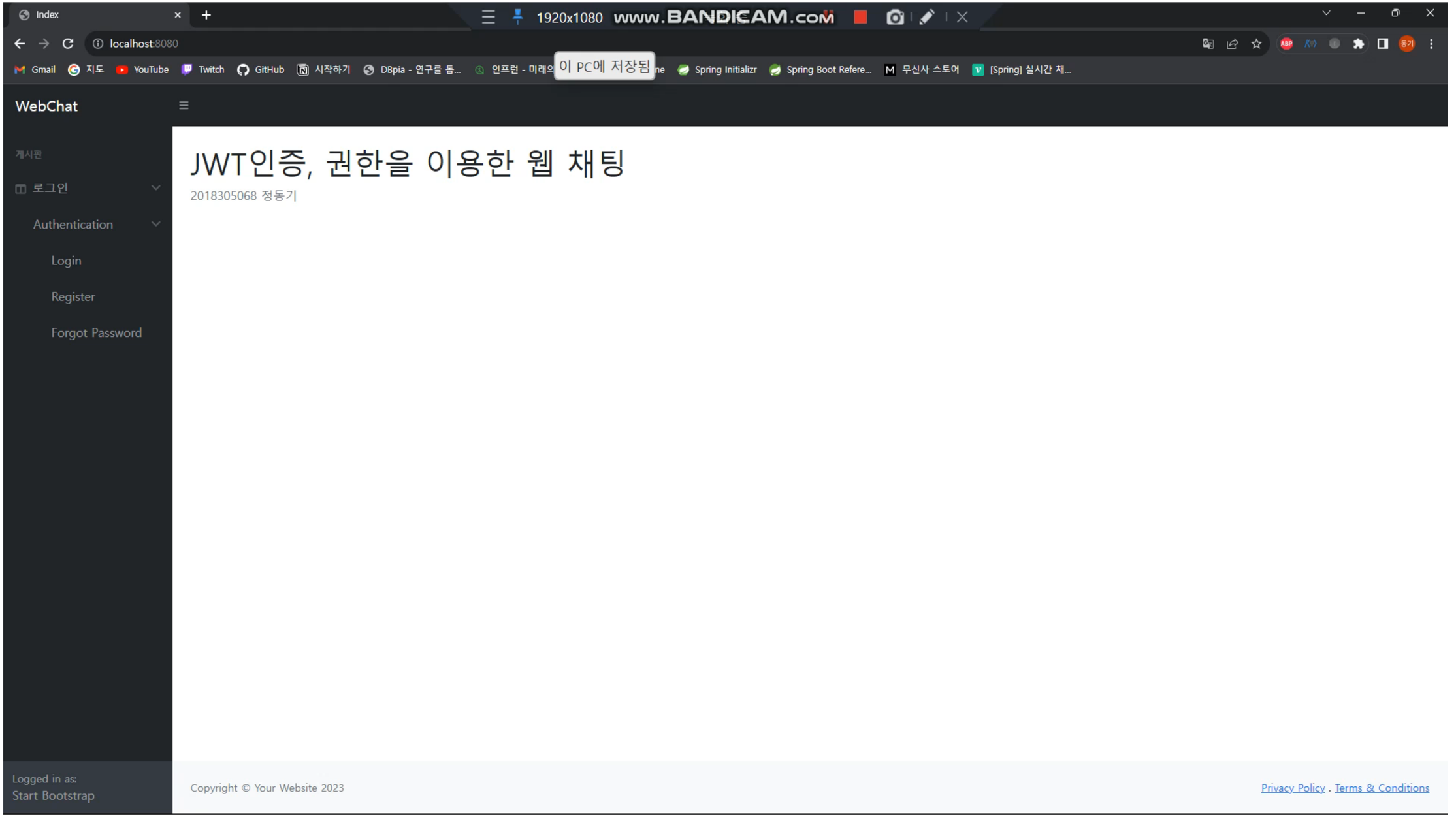
3. 사용자 방출
   1) ADMIN, MANAGER, ROOM CREATER 방출 가능
   2) 방출 시 queue [개인 채팅] 으로 메시지를 보내줌



시연 순서

1. 로그인 후 JWT 확인
2. refreshToken으로 accessToken 가져오기
3. 사용자 권한 – ADMIN , MANAGER
4. 게시판 - 내 아래 권한 접근 금지, SEMIUSER 파일 접근 금지
5. 채팅 추방 – 비공개 채팅




Index
1920x1080 www.BANDICAM.com
localhost:8080
이 PC에 저장됨
Gmail
지도
YouTube
Twitch
GitHub
시작하기
DBpia - 연구를 돕...
Spring Initializr
Spring Boot Refere...
무신사 스토어
[Spring] 실시간 채...
WebChat
게시판
로그인
Authentication
Login
Register
Forgot Password
JWT인증, 권한을 이용한 웹 채팅
2018305068 정동기
Logged in as:
Start Bootstrap
Copyright © Your Website 2023
Privacy Policy . Terms & Conditions

감사합니다