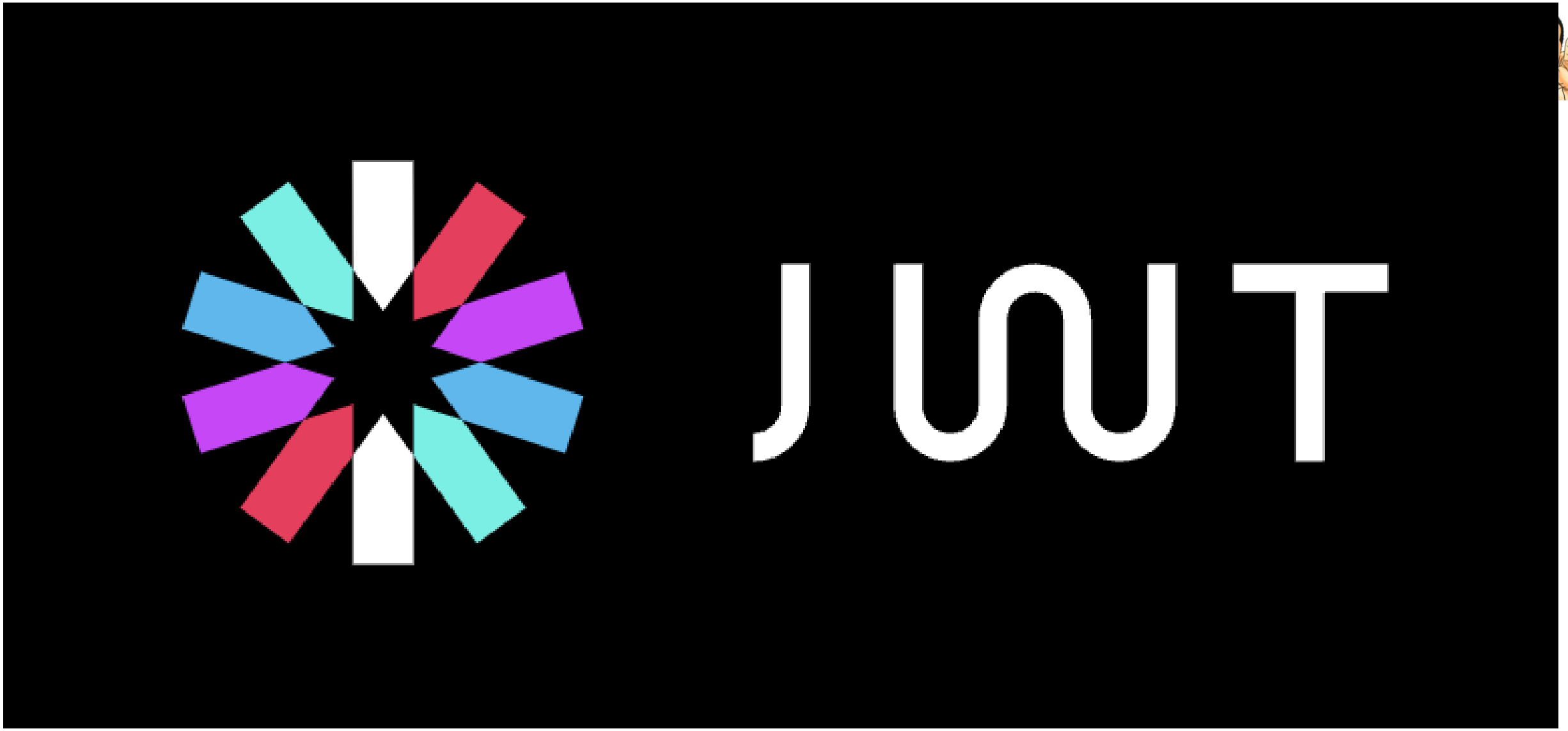
KB

It's Your Life

with


JUNIT

JWT 가 뭔지 아시는 분!?

Json Web Token 의 약자로

데이터를 JSON 형태로 저장하면서
해당 데이터를 암호화 한 토큰입니다!

JWT

(Json Web Token)


머리!
가슴!
몸통!
http://blog.naver.com/grandchyren

HEADER . PAYLOAD . SIGNATURE

헤더

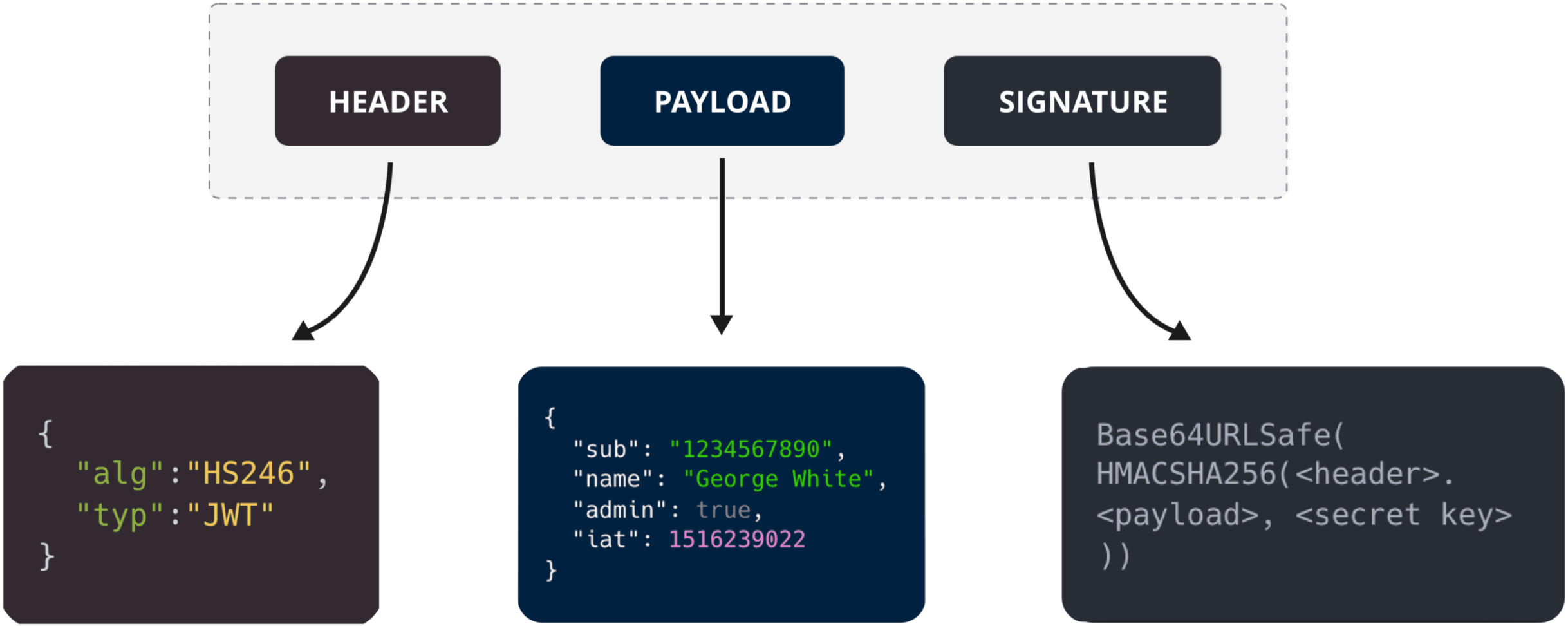
암호화 방법과
토큰의 타입 지정

내용

저장할 데이터

서명

암호화에 넣을 비밀 키와
만료 기간 설정


HEADER
PAYLOAD
SIGNATURE
{
"alg":"HS246",
"typ":"JWT"
}
{
"sub": "1234567890",
"name": "George White",
"admin": true,
"iat": 1516239022
}
Base64URLSafe(
HMACSHA256(<header>.
<payload>, <secret key>
))

https://jwt.io/

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

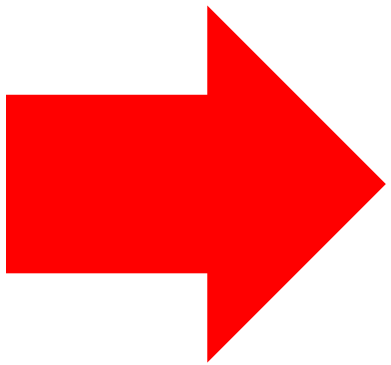
```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

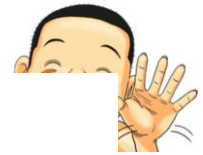
```
  "isOld": "true",
  "isMarried": "false"
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  It's decided by tetz
) secret base64 encoded
```



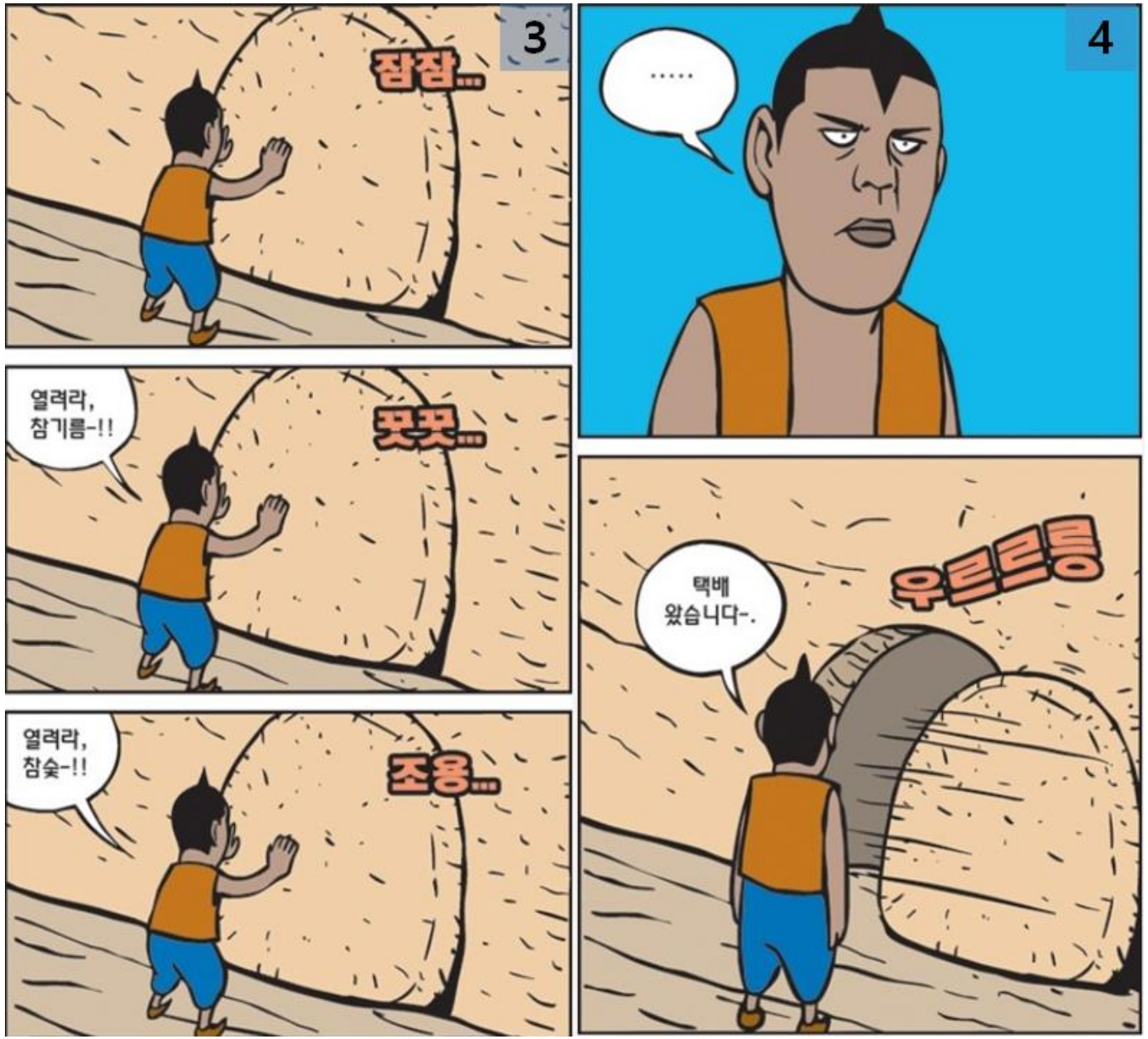
Encoded PASTE A TOKEN HERE



```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JuYW1lIjoidGV0eiIsImlzT2xkIjoidHJ1ZSIsI
mlzTWFycmllZCI6ImZhbHNlIn0.yhEOinS7zpkr
Hiudje-3tE17ZwiYMvm4AIL7fK2h1Nc
```


열려라 참깨!!


오뚜기
열려라
참깨라면
계란이 들어있어요!




```
    base64UrlEncode(header) + "." +
    base64UrlEncode(payload),
    It's decided by tetz
) secret base64 encoded
```

아무리 JWT 의 인코딩 방식과
암호화 방식을 알아도
요 SECRET KEY 를 알아야
해당 JWT 토큰을
JSON 으로 변환할 수 있습니다!


배트맨, 초능력도 없는 우린 뭘 할 수 있죠?
우린 쓸모가 없다.
팝콘이나 가져와라 로빈.
20

Oauth 2.0

(Open Authorization 2.0)

OAUTH2

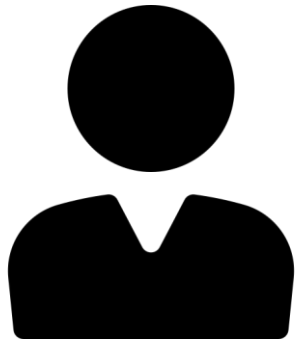

security


해줘
벅벅


인증을
부탁한다!
반드시
돌아올게
!!!
어!?

Oauth 2.0 의 과정

우리가 직접 보안 관리하고
ID / PW 관리하기 힘드니
카카오에 로그인하시면 그걸로 처리할게요



카카오
ID / PW 입력

로그인 완료

여러분의
서비스

회원 가입
처리

여러분의
DB

회원 정보 및
AccessToken,
RefreshToken
전달

카카오
서비스

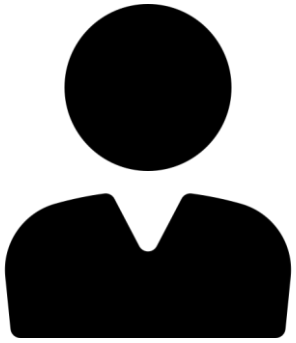
전달 받은
ID / PW 확인

회원 정보 및
AccessToken,
RefreshToken
(JWT) 발급

카카오
인증
서버

AccessToken 이 있으면
카카오는 해당 사용자가
로그인이 된 사용자임을
판단이 가능합니다!

이제 카카오 로그인 하셨으니
카카오 관련 데이터 요청이 가능합니다!



카카오 친구 목록 요청

카카오 친구 목록을
이용한 서비스 제공

여러분의
서비스

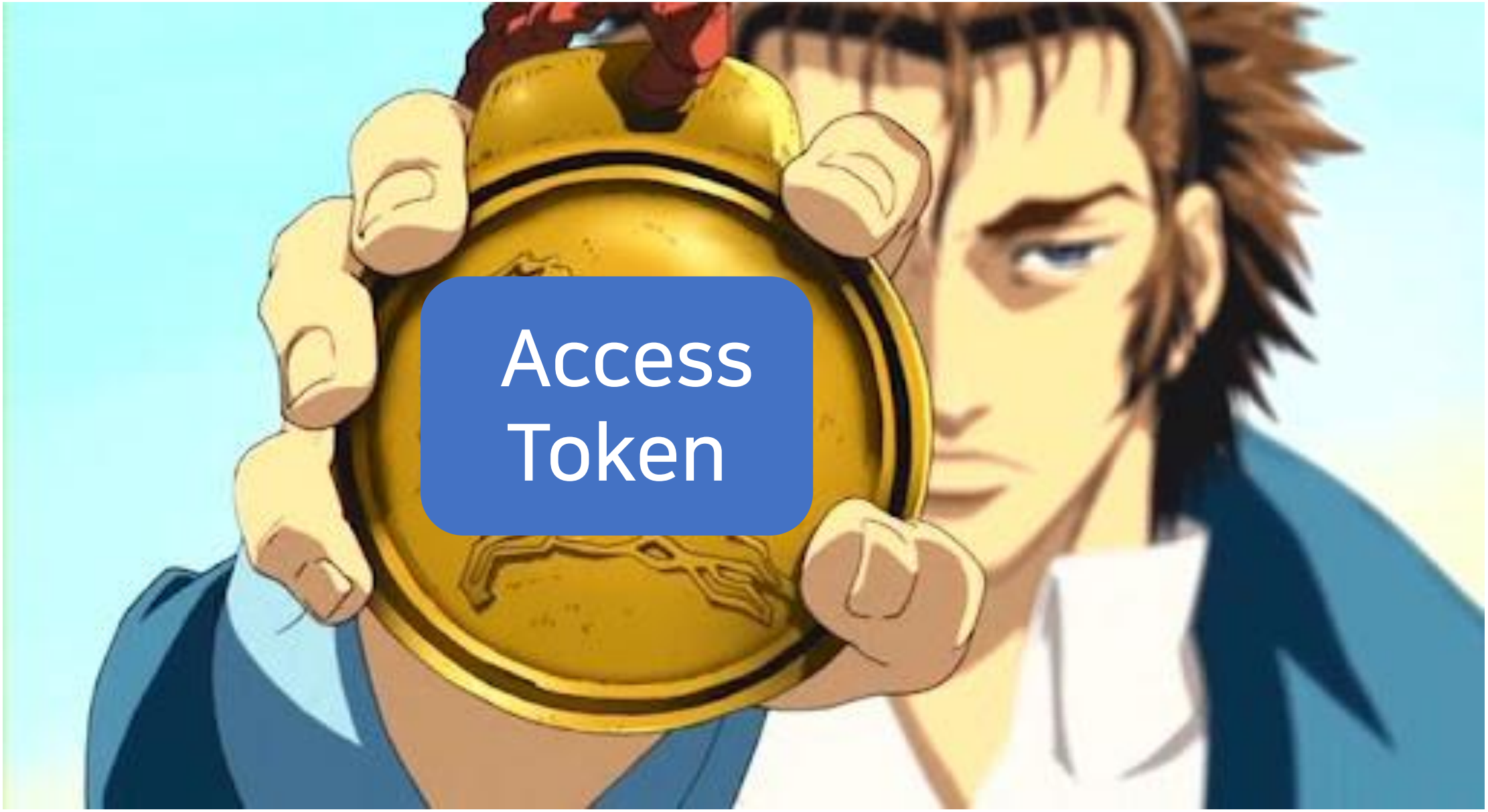
카카오 친구 목록 요청
Header에 AccessToken 삽입

카카오 친구 목록 응답

카카오
서비스

AccessToken 은 JWT 를 사용하므로
카카오가 발급한 SecretKey 를 모르면
해당 정보를 절대 알 수 없습니다!

따라서 해당 정보를 백엔드 서버나
클라이언트에 저장이 가능합니다!

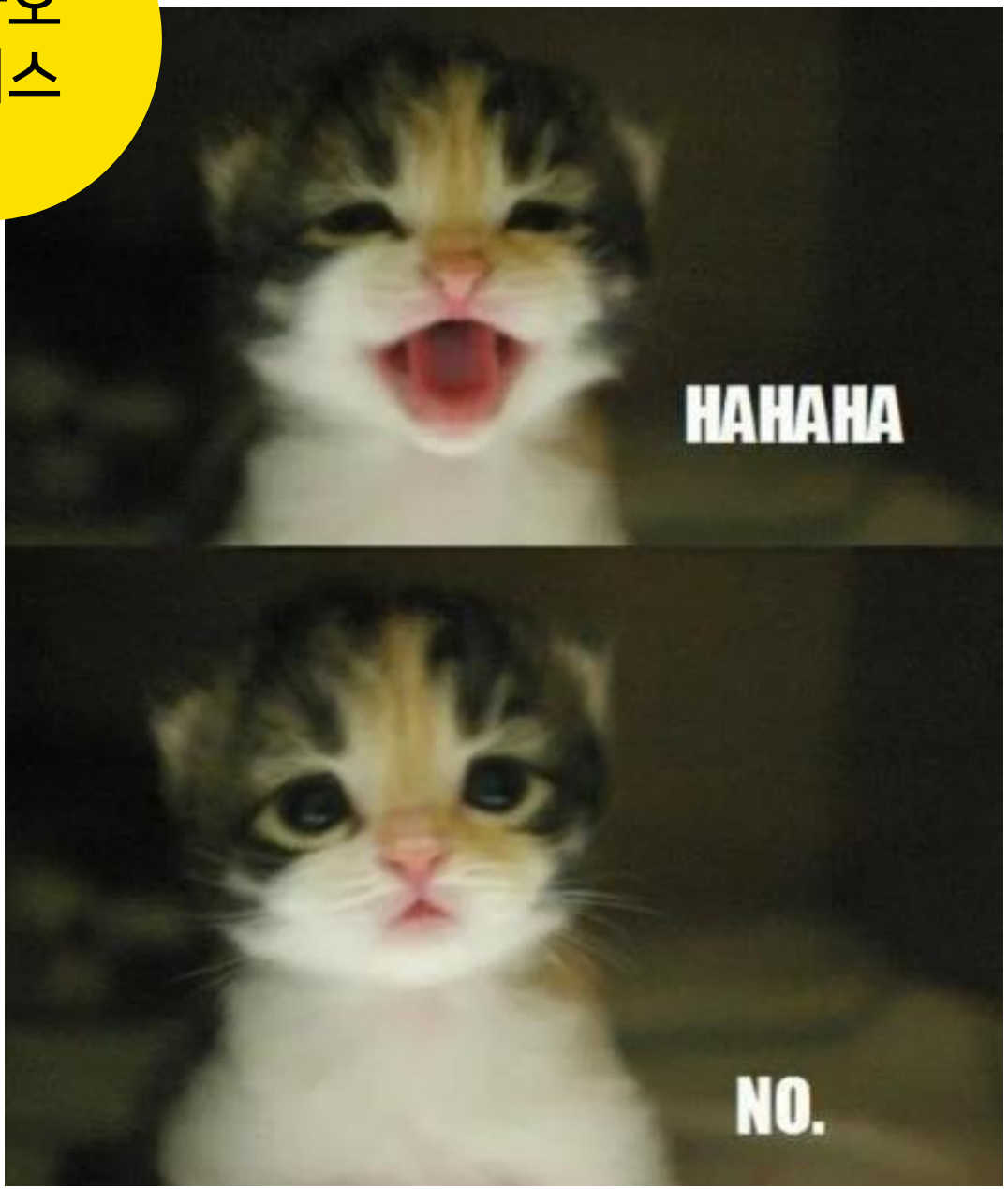

Access
Token

해커가 여러분의 AccessToken 을 탈취한다면!?
어떤 일이 발생 할까요?

카카오 서버 입장에서 해당 토큰을 가지고 있다는 것은
로그인한 사용자를 의미하므로 매우 위험합니다!

나쁜 X

카카오
서비스



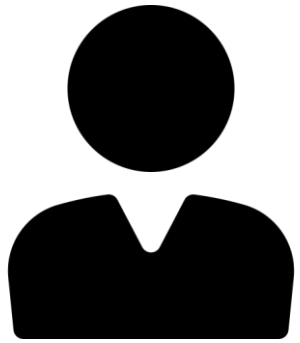
AccessToken 은 탈취의 가능성을 염두에 두고
만료 기간을 매우 짧게 가져 갑니다!

JWT 의 특성으로 만료 기간 변조는 불가능하기 때문에
훔친 AccessToken 으로는 인증을 뚫기가 매우 어렵습니다!

AccessToken 만료 기간이 짧으면
그럼 매번 로그인 다시 해줘야 하는거 아닌가요?

그래서 Refresh Token 이라는 것을 사용합니다!

우리가 직접 보안 관리하고
ID / PW 관리하기 힘드니
카카오에 로그인하시면 그걸로 처리할게요



카카오
ID / PW 입력

로그인 완료

여러분의
서비스

회원 가입
처리

여러분의
DB

회원 정보 및
AccessToken,
RefreshToken
전달

카카오
서비스

전달 받은
ID / PW 확인

회원 정보 및
AccessToken,
RefreshToken
(JWT) 발급

카카오
인증
서버

AccessToken 이 있으면
카카오는 해당 사용자가
로그인이 된 사용자임을
판단이 가능합니다!

여러분의
서비스

카카오 데이터 요청
Header 에 AccessToken 이 있는 상태

AccessToken 만료 응답

Header 에 RefreshToken 을 넣어서
AccessToken 요청

새로운 AccessToken 발급

카카오 데이터 요청
Header 에 새로 발급 받은 AccessToken

요청한 데이터 응답

카카오
서비스

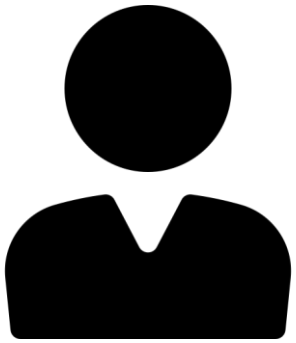
Refresh Token 이 탈취 당하면 어쩌쥬?

이럴 때를 대비해서 RefreshToken Rotation 이라는 방법을 사용합니다!

즉, AccessToken 이 재발급 되면 RefreshToken 도 변경을 해버리는 방법입니다!

→ 다만, 완벽한 보안은 아니기 때문에
Token 은 노출이 잘 안되도록 보관하는 것이 중요합니다!

우리가 직접 보안 관리하고
ID / PW 관리하기 힘드니
카카오에 로그인하시면 그걸로 처리할게요



카카오
ID / PW 입력

로그인 완료

여러분의
서비스

회원 가입
처리

여러분의
DB

회원 정보 및
AccessToken,
RefreshToken
전달

카카오
서비스

전달 받은
ID / PW 확인

회원 정보 및
AccessToken,
RefreshToken
(JWT) 발급

카카오
인증
서버

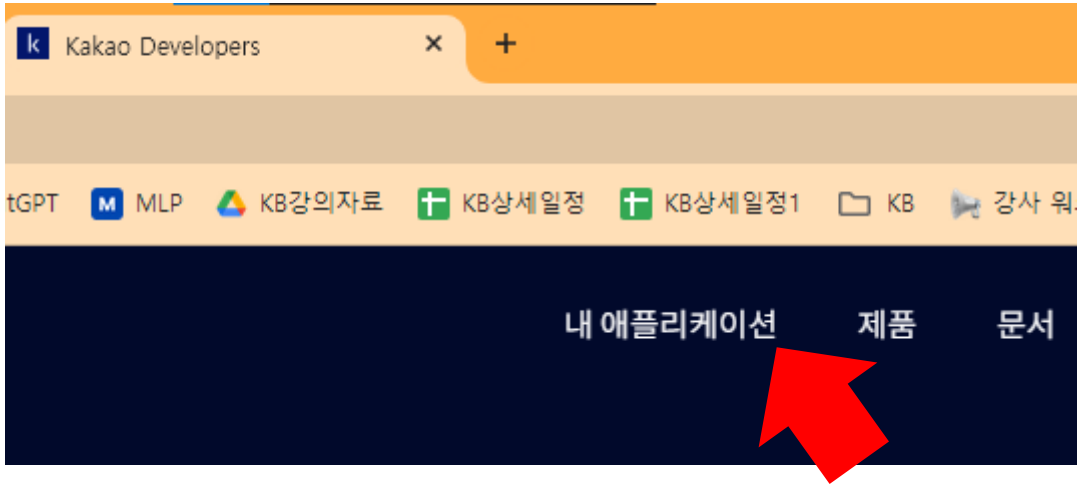

당장 하자

카카오 로그인

구현하기!

카카오 API

설정하기!


Kakao Developers
tGPT
MLP
KB강의자료
KB상세일정
KB상세일정1
KB
내 애플리케이션
제품
문서

kakao developers

내 애플리케이션 > 앱 설정 > 대시보드

앱 설정

대시보드

일반

비즈니스

앱 키

플랫

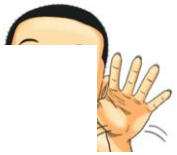
앱 권한 신청

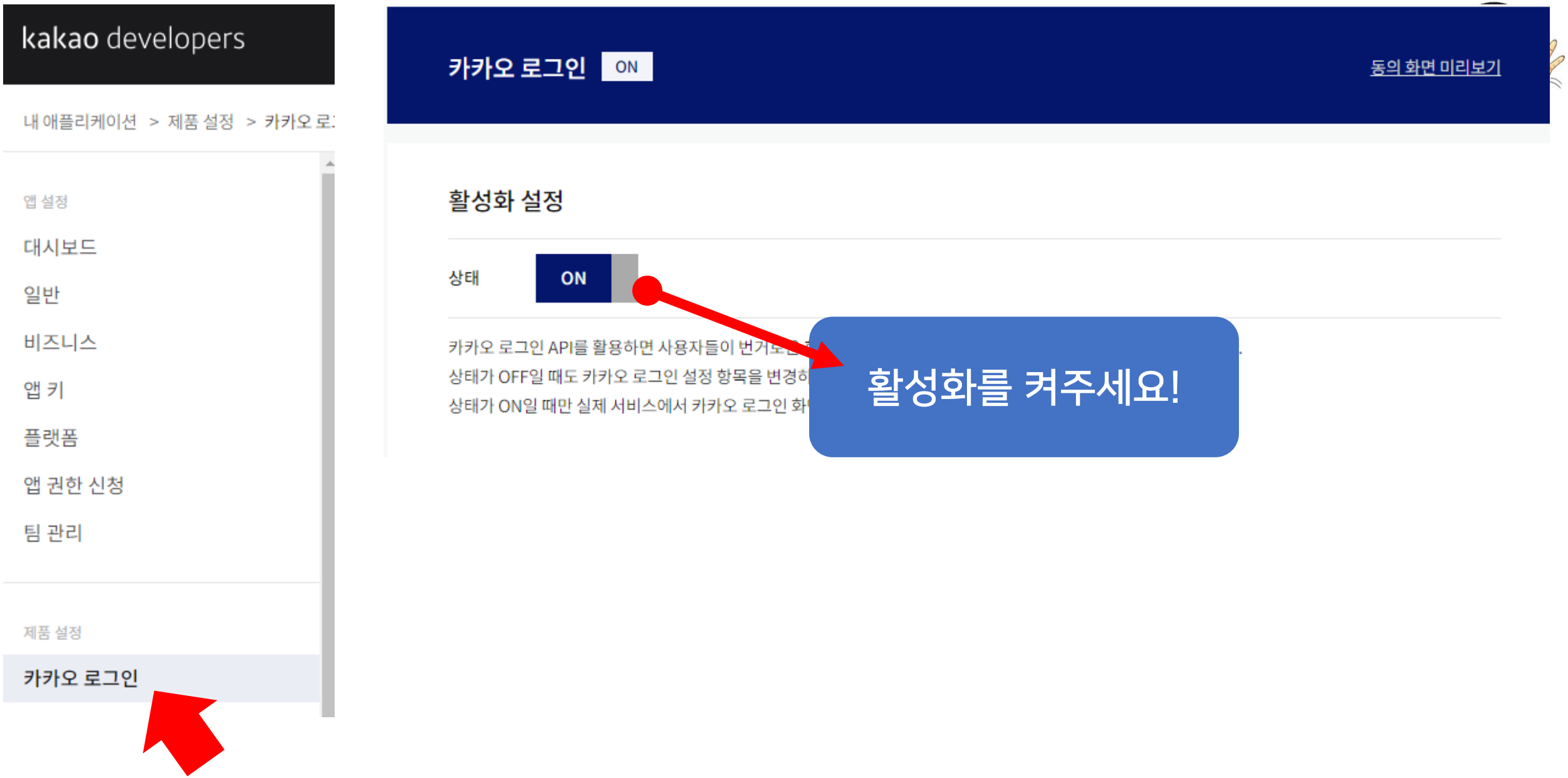
팀 관리

앱 키

플랫폼	앱 키		재발급
네이티브 앱 키		복사	재발급
REST API 키		복사	재발급
JavaScript 키		복사	재발급
Admin 키		복사	재발급

요 Key 들이 매우 중요합니다!!




kakao developers
내 애플리케이션 > 제품 설정 > 카카오 로
앱 설정
대시보드
일반
비즈니스
앱 키
플랫폼
앱 권한 신청
팀 관리
제품 설정
카카오 로그인
카카오 로그인 ON
동의 화면 미리보기
활성화 설정
상태
ON
카카오 로그인 API를 활용하면 사용자들이 번거로
상태가 OFF일 때도 카카오 로그인 설정 항목을 변경하
상태가 ON일 때만 실제 서비스에서 카카오 로그인 화
활성화를 켜주세요!

제품 설정

카카오 로그인

동의항목

간편가입

카카오톡 채널

개인정보 국외이전

연결 끊기

사용자 프로퍼티

Redirect URI

Redirect URI 등록

- 카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다. (최대 10개)
- REST API로 개발하는 경우 필수로 설정해

카카오에서 토큰을 보내줄 주소를
여기에서 설정하셔야 합니다!!

Redirect URI

Redirect URI

카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다.
여러개의 URI를 줄바꿈으로 추가해주세요. (최대 10개)
REST API로 개발하는 경우 필수로 설정해야 합니다.
예시: (O) https://example.com/oauth (X) https://www.example.com/oauth

http://localhost:8080/kakao/login

취소 | 저장

우리가 사용할 주소인
http://localhost:8080/kakao/login
으로 설정

Redirect URI

삭제 | 수정

Redirect URI	http://localhost:8080/kakao/login

- 카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다. (최대 10개)
- REST API로 개발하는 경우 필수로 설정해야 합니다.

kakao developers

내 애플리케이션 > 제품 설정 > 카카오 로

앱 설정

대시보드

일반

비즈니스

앱 키

플랫폼

앱 권한 신청

팀 관리

제품 설정

카카오 로그인

동의항목

카카오 로그인 ON

동의 화면 미리보기

동의항목

카카오 로그인으로 서비스를 시작할 때 동의받는
사업자 정보를 등록하고 비즈니스 인증을 완료하

개인정보 동의항목 심사 신청

닉네임을 받아와야 username
설정이 가능하므로 설정해줘야만 합니다!

개인정보

항목 이름	ID	상태	
닉네임	profile_nickname	● 사용 안 함	설정
프로필 사진	profile_image	● 사용 안 함	설정
카카오계정(이메일)	account_email	○ 권한 없음	
이름	name	○ 권한 없음	
성별	gender	○ 권한 없음	

동의 항목 설정

항목
닉네임 / profile_nickname

동의 단계

- 필수 동의
  카카오 로그인 시 사용자가 필수로 동의해야 합니다.
- 선택 동의
  사용자가 동의하지 않아도 카카오 로그인을 완료할 수 있습니다.
- 이용 중 동의
  카카오 로그인 시 동의를 받지 않고, 항목이 필요한 시점에 동의를 받습니다.
- 사용 안 함
  사용자에게 동의를 요청하지 않습니다.

동의 목적 [필수]

개발자 앱 동의 항목 관리 화면내에 입력하는 사실이 실제 서비스 내용과 다를 경우 API 서비스의 거부 사유가 될 수 있습니다.

취소 저장

당연히 카카오 사용자가 동의를 해야
가져올 수 있으니 로그인 과정에서
동의 항목에서 승인을 받아야만 합니다!

개인정보 동의항목 심사 신청

일단 닉네임과 프로필 사진을
받아 올 수 있도록 두 항목은
필수 동의로 변경을 합시다!

개인정보

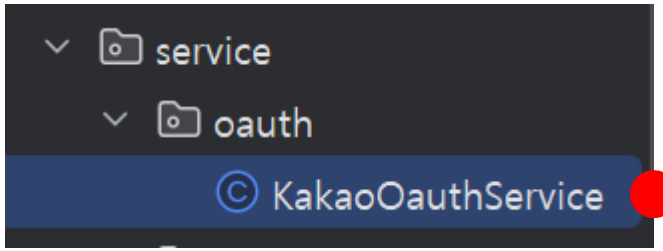
항목 이름	ID	상태	
닉네임	profile_nickname	● 필수 동의	설정
프로필 사진	profile_image	● 필수 동의	설정
카카오계정(이메일)	account_email	○ 권한 없음	
이름	name	○ 권한 없음	
성별	gender	○ 권한 없음	
연령대	age_range	○ 권한 없음	
생일	birthday	○ 권한 없음	
출생 연도	birthyear	○ 권한 없음	

더 민간한 개인정보는
좌측 상단의 개인정보
동의항목 심사 신청을 통해
비즈니스 앱으로 전환을
해야 요청이 가능합니다!

카카오 서비스

구현하기!

https://github.com/xenosign/spring-code-repo/blob/main/kakao/KakaoService.java



카카오 Oauth 를 처리할
KakaoOauthService 클래스를 만들어 봅시다

카카오에서 제공하는 REST API 키를
아래의 REST_API_KEY 에 넣어 주세요!

```
@Service  2 usages    kdtTetz *
public class KakaoOauthService {
    private String REST_API_KEY = "fe2ce67ae1aa8d5ab53a015eb2a03bea";
    private String REDIRECT_URI = "http://localhost:8080/kakao/login";
```

앱 키

플랫폼	앱 키
네이티브 앱 키	de84448e209d8754ae0a3a29373f1e60
REST API 키	fe2ce67ae1aa8d5ab53a015eb2a03bea
JavaScript 키	562bf94127c2578e84824d0dc7e20e01
Admin 키	d0651b7da8c91d9994febde6fed904e7

우리가 설정한 Redirect uri 를 넣어주세요
http://localhost:8080/kakao/login

```java
@Service  2 usages    kdtTetz *
public class KakaoOauthService {
    private String REST_API_KEY = "fe2ce67ae1aa8d5ab53a015eb2a03bea";  1
    private String REDIRECT_URI = "http://localhost:8080/kakao/login";
```

Access Token 받기

https://developers.kakao.com/docs/latest/ko/kakaologin/rest-api

토큰 받기

기본 정보

메서드	URL	인증 방식
POST	https://kauth.kakao.com/oauth/token	-

권한	사전 설정	카카오 로그인	동의항목
-	플랫폼 등록 카카오 로그인 활성화 Redirect URI 등록 동의항목 OpenID Connect 활성화(선택)	필요	필요: 필수 동의항목

카카오에서 발급하는 Access Token 을
발급 받는 메서드!

```
public String getAccessToken(String authorize_code) {  1 usage
    String access_Token = "";
    String refresh_Token = "";
    String reqURL = "https://kauth.kakao.com/oauth/token";

    try {
        URL url = new URL(reqURL);
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();

        conn.setRequestMethod("POST");
        conn.setDoOutput(true);
```

Access Token 발급은
해당 API 주소로 요청을
보내면 됩니다!

특정 주소에 다양한
파라미터 옵션을 붙여서 보내는 방식을 적용

```
BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()
StringBuilder sb = new StringBuilder();
sb.append("grant_type=authorization_code");
sb.append("&client_id=").append(REST_API_KEY);
sb.append("&redirect_uri=").append(REDIRECT_URI);
sb.append("&code=").append(authorize_code);
bw.write(sb.toString());
bw.flush();

System.out.println("ACCESS TOKEN 요청 URL : " + sb);
```

카카오 API 문서가 요구하는 방식대로
파라미터를 맞춰서 전달!

최종 요청 URL 확인하는 코드

```
ACCESS TOKEN REQUEST URL : grant_type=authorization_code&client_id=fe2ce67ae1aa8d5ab53a015eb2a03bea&redirect_uri
 =http://localhost:8080/kakao/login&code
 =cznEe1Ay4G1XCkNm7ItQRHAS62E_azss7xNQK_oh1XATTRqDte31GAAAAAQKPXOaAAABkZbX3FUq3eF1vjqPRg
```

방금 요런 주소로 우리는 요청을 날린 겁니다!

카카오 API 서버가 보낸 STATUS 확인

```java
int responseCode = conn.getResponseCode();
System.out.println("ACCESS TOKEN 응답 코드 : " + responseCode);

BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream()));
String line = "";
StringBuilder result = new StringBuilder();

while ((line = br.readLine()) != null) {
    result.append(line);
}
System.out.println("카카오 응답 body 의 내용 : " + result);
```

카카오가 보낸 응답을 객체로 만들어서 출력

ACCESS TOKEN 발급의 응답 STATUS → 200

```
ACCESS TOKEN 쑧땛 肄뷇뱶 : 200
移댁뭅삀 쑧땛 body 쑱 궡슜 : {"access_token":"TwzAwv0G_a2vPfyPtJZtAtSjBtqq7L6sAAAAAQo9cxcAAAGRlt8td4E8pQXSEbh1
 "token_type":"bearer","refresh_token":"Mk6GYwon-23nSDXBhuyxxNU2XYJDXe6SAAAAAgo9cxcAAAGRlt8tdIE8pQXSEbh1",
"expires_in":21599,"scope":"profile_image profile_nickname","refresh_token_expires_in":5183999}
```

```
access_token : TwzAwv0G_a2vPfyPtJZtAtSjBtqq7L6sAAAAAQo9cxcAAAGRlt8td4E8pQXSEbh1
refresh_token : Mk6GYwon-23nSDXBhuyxxNU2XYJDXe6SAAAAAgo9cxcAAAGRlt8tdIE8pQXSEbh1
```

발급 된 토큰을 확인!

Access Token 으로 유저 정보 받기

```java
public JsonObject getUserInfo(String access_Token) {  1 usage
    String reqURL = "https://kapi.kakao.com/v2/user/me";
    JsonObject userInfo = new JsonObject();
```

Access 토큰으로 유저 정보를 요청하는
API 는 여기 입니다!

API 의 요구에 맞게 주소를 만들어서 요청을 보냅니다!

```
try {
    URL url = new URL(reqURL);
    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
    conn.setRequestMethod("POST");
    conn.setRequestProperty("Authorization", "Bearer " + access_Token);
```

토큰의 타입이 Bearer 타입이었으므로 해당 정보 기입

그리고 우리가 받은 Access Token 을 같이 전달!

카카오 API 서버가 보낸 STATUS 확인

```java
int responseCode = conn.getResponseCode();
System.out.println("유저 정보 요청 응답 코드 : " + responseCode);

BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream()));
String line = "";
StringBuilder result = new StringBuilder();
while ((line = br.readLine()) != null) {
    result.append(line);
}
System.out.println("유저 정보 요청 응답 Body 의 내용 : " + result);
```

카카오가 보낸 응답을 객체로 만들어서 출력

유저 정보 요청에 대한 응답 코드 : 200

```
responseCode : 200
response body : {"id":3680546220,"connected_at":"2024-08-26T19:48:16Z","properties":{"nickname":"
 ","profile_image":"http://k.kakaocdn.net/dn/05WbA/btrLgq1ywgh/pgafGUadTpCBNkstWEk2V0/img_640x640.jpg",
 "thumbnail_image":"http://k.kakaocdn.net/dn/05WbA/btrLgq1ywgh/pgafGUadTpCBNkstWEk2V0/img_110x110.jpg"},
 "kakao_account":{"profile_nickname_needs_agreement":false,"profile_image_needs_agreement":false,
 "profile":{"nickname":"","thumbnail_image_url":"http://k.kakaocdn
 .net/dn/05WbA/btrLgq1ywgh/pgafGUadTpCBNkstWEk2V0/img_110x110.jpg","profile_image_url":"http://k.kakaocdn
 .net/dn/05WbA/btrLgq1ywgh/pgafGUadTpCBNkstWEk2V0/img_640x640.jpg","is_default_image":false,
 "is_default_nickname":false}}}
```

전달 받은 응답의 결과!

동의 항목에 추가 한 내용인
닉네임과 프로필 사진을 받아왔습니다!

받은 유저 정보로

유저 정보 추출

```
JsonParser parser = new JsonParser();
JsonElement element = parser.parse(result.toString());
JsonObject response = element.getAsJsonObject();

// ID 추출
if (response.has( memberName: "id")) {
    long id = response.get("id").getAsLong();
    System.out.println("카카오 ID : " + id);
    userInfo.addProperty( property: "id", id);
}

// 닉네임 추출
JsonObject properties = response.getAsJsonObject( memberName: "properties");
if (properties != null && properties.has( memberName: "nickname")) {
    String nickname = properties.get("nickname").getAsString();
    System.out.println("카카오 닉네임 : " + nickname);
    userInfo.addProperty( property: "nickname", nickname);
}

br.close(); // 자원 닫기
```

JSON 을 객체로 변환하는 라이브러리 사용

카카오 ID(숫자로 된 ID) 를 추출

카카오 닉네임 추출

```
    return userInfo;  // 오류 발생 시에도 수집된 정보 반환
}
```

컨트롤러에 전달할 유저 정보를 리턴해 줍니다!

카카오 컨트롤러 작성하기!

https://github.com/xenosign/spring-code-repo/blob/main/kakao/KakaoController.java

요 친구는 우리가 직접 요청을 보내는게 아니라
카카오가 redirect 로 보내주는 값을 받는 형태입니다!

방금 구현한 KakaoService 와
회원 정보 저장을 위해 UserService 를
주입 받아서 사용

```java
@Controller  kdtTetz
@RequiredArgsConstructor
@RequestMapping("/kakao")
public class KakaoController {
    private final KakaoOauthService kakaoLoginService;
    private final UserService userService;

    @GetMapping("/login")  kdtTetz
    public String kakaoLogin(@RequestParam("code") String code) {
        String accessToken = kakaoLoginService.getAccessToken(code);
        JsonObject userInfo = kakaoLoginService.getUserInfo(accessToken);
```

```java
@Controller  kdtTetz
@RequiredArgsConstructor
@RequestMapping("/kakao")
public class KakaoController {
    private final KakaoOauthService kakaoLoginService;
    private final UserService userService;

    @GetMapping("/login")  kdtTetz
    public String kakaoLogin(@RequestParam("code") String code) {
        String accessToken = kakaoLoginService.getAccessToken(code);
        JsonObject userInfo = kakaoLoginService.getUserInfo(accessToken);
```

카카오 에서 전달한 인증 코드를
사용해서 AccessToken 을 받기!

받은 AccessToken 을 다시 전달해서
사용자 정보를 받기

```java
if (userInfo != null) {
    String nickname = userInfo.get("nickname").getAsString();

    if (userService.findByUsername(nickname) == null) {
        User kakaoUser = new User();
        kakaoUser.setUsername(nickname);
        kakaoUser.setPassword("kakao");
        kakaoUser.setRoles("ROLE_KAKAO");
        userService.save(kakaoUser);
    }

    // Authentication 객체 생성
    Authentication authentication = new UsernamePasswordAuthenticationToken(
            nickname, credentials: null, Collections.singletonList(new SimpleGrantedAuthority( role: "ROLE_KAKAO")));

    // SecurityContext에 Authentication 객체를 저장
    SecurityContextHolder.getContext().setAuthentication(authentication);

    return "redirect:/security/member";
```

KakaoService 로 부터 받은
유저 정보에서 nickname 추출

해당 닉네임으로 DB 의 회원을
찾아서 없을 경우 가입 진행

회원이 있을 경우 바로 nickname 을
시큐리티에 전달하여 로그인 진행!

```
    return "redirect:/security/member";
} else {
    // 에러 처리
    return "redirect:/security/login-failed";
}
```

상황에 맞춰서 로그인 성공 페이지 혹은
실패 페이지로 리다이렉트!

헤더

수정 하기

https://github.com/xenosign/spring-code-repo/blob/main/jsp/security/header.jsp

요기에 본인의 REST API 키를
넣어 주시면 됩니다!

```jsp
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%
    String REST_API_KEY = "fe2ce67ae1aa8d5ab53a015eb2a03bea";
    String REDIRECT_URI = "http://localhost:8080/kakao/login";
    String KAKAO_URI = "https://kauth.kakao.com/oauth/authorize?client_id=" + REST
%>
<header>
```

```jsp
<a href=<%=KAKAO_URI%>>카카오 로그인</a>
```

시큐리티

설정 변경!

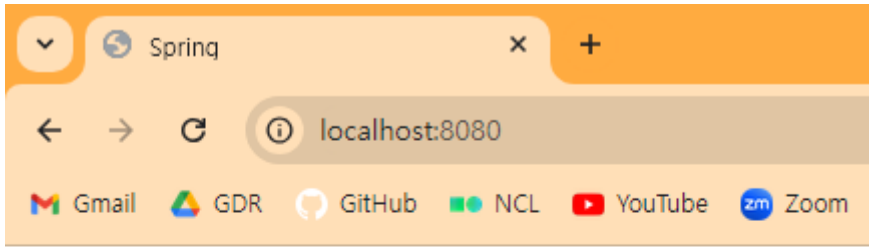
카카오에서 보내주는 요청도 받아야 하기 때문에
해당 주소도 모두 접근 가능하도록 변경

```
@Override  Tetz *
protected void configure(HttpSecurity http) throws Exception {
    http.authorizeRequests()
            .antMatchers("/").permitAll()
            .antMatchers("/user/**").permitAll()
            .antMatchers("/security/admin").access( attribute: "hasRole('ROLE_ADMIN')")
            .antMatchers("/security/**").permitAll()
            .antMatchers("/kakao/**").permitAll()
            .antMatchers("/**").access( attribute: "hasAnyRole('ROLE_MEMBER', 'ROLE_KAKAO')");
```

카카오 권한을 가진 사용자도
다른 페이지 접속이 가능해야 하므로
권한을 추가하여 적용

카카오 로그인

확인!



V1 MyBatis

HOME 게시글 목록 404 error

V1 REST

HOME 게시글 목록

V1 JPA

HOME 게시글 목록

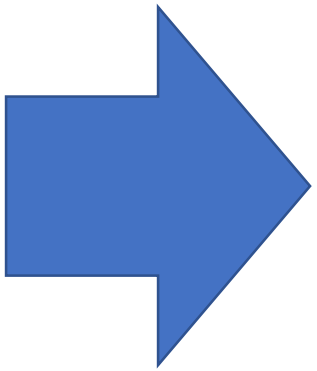
회원 기능

회원가입 로그인 로그아웃

시큐리티 회원 기능

admin member 로그인 로그아웃 카카오 로그인

Hello, Spring World!



SECURITY 로그인 성공

사용자 정보

사용자명: 이효석

권한: ROLE_KAKAO