

Docker에서 Server 이미지 다운 받고 실행하기

```
docker pull ithinkso/testserver
```

```
docker run -p 8080:8080 ithinkso/testserver
```

build 아키텍처가 linux/amd64이므로

애플 arm cpu를 사용하시는 분은 실행이 느릴 수 있습니다.

실행 log에 아래와 같이 뜨면 서버 실행 완료입니다.

```
2022-07-08 03:53:19.928 INFO 1 --- [          main] o.s.m.s.b.SimpleBrokerMessageHandler : Started.  
2022-07-08 03:53:20.013 INFO 1 --  
- [          main] com.example.demo.DemoApplication : Started DemoApplication in 26.835 seconds (JVM running for 30.136)  
2022-07-08 03:53:20.094 INFO 1 --- [          main] com.example.demo.InitDB : initialize database
```

# DB 확인하기

http://localhost:8080/h2-console

Login

Saved Settings: Generic H2 (Embedded) ▼

Setting Name: Generic H2 (Embedded) Save Remove

---

Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:test

User Name: sa

Password:

Connect Test Connection

초기 데이터가 들어가 있습니다.

jdbc:h2:mem:test

- OAuth2
- USERS
- INFORMATION\_SCHEMA
- Sequences
- Users
- H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM OAuth2;  
SELECT * FROM users;
```

SELECT \* FROM OAuth2;

O_AUTH2_ID	GOOGLE	KAKAO	NAVER	MEMBER_ID
2	null	null	null	1

(1 row, 18 ms)

SELECT \* FROM users;

MEMBER_ID	EMAIL	NAME	PASSWORD	ROLE
1	asdf@asdf.com	홍길동	\$2a\$10\$E79QKK9NtHfhMFYiaNsWsuGMrMpcXo/.pbf89bwvDXEITkNy/penS	ROLE_USER

(1 row, 2 ms)

Test Api – postman을 이용하시면 편리하게 Api를 호출할 수 있습니다.

## 1. 회원가입

Url : <http://localhost:8080/signup>

Method : Post

Body : name(string), email(string), password(string), role(string)

USER 또는 ADMIN



# Test Api

## 1. 회원가입 예시

The screenshot displays a REST client interface with the following details:

- Method:** POST
- URL:** http://localhost:8080/signup
- Body:** A JSON object containing registration details:

```
{  "name": "홍길동2",  "email": "asdf2@asdf.com",  "password": "asdf@",  "role": "ADMIN"}
```
- Response:** Status 200 OK, Time 107 ms, Size 574 B. The response body is shown in a pretty-printed format:

```
1 회원가입 완료.
2 name=홍길동2, email=asdf2@asdf.com, password(암호화됨)=$2a$10$1SvxwbP4ZXQqSHq8iiB5puPzhmEfUXUV2PWLkWwp8sF2L9at/u4nK,
  role=ROLE_ADMIN
```

# Test Api

## 2. 로그인

Url : <http://localhost:8080/login>

Method : Post

Body : email(string), password(string)

Response Header에 Authorization : Bearer Token(JWT)이 전송됩니다.

이후 로그인이 필요한 Api를 호출하실 때에는 반드시 Request Header에 해당 JWT가 포함되어 있어야 합니다.

# Test Api

## 2. 로그인 예시

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/login`. The request body is a JSON object with email and password fields. The response status is 200 OK, and the response headers include Vary, Access-Control-Request-Method, Access-Control-Request-Headers, and an Authorization header with a Bearer token. The Authorization header is highlighted with a red box.

**Request:**

- Method: POST
- URL: `http://localhost:8080/login`
- Body (JSON):

```
{  "email": "asdf@asdf.com",  "password": "asdf@"}
```

**Response:**

- Status: 200 OK
- Time: 245 ms
- Size: 625 B
- Headers (13):

KEY	VALUE
Vary	Origin
Vary	Access-Control-Request-Method
Vary	Access-Control-Request-Headers
Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJzdWIiOiLthqDtgbDs...

## Test Api

### 3. User와 Admin이 접근 가능한 API

Url : <http://localhost:8080/user>

Method : Get

Header : Authorization에 JWT가 포함되어 있어야 합니다.

JWT 인증 실패 또는 접근 권한이 없으면 403 에러가 뜹니다.

## Test Api

### 4. Admin만 접근 가능한 API

Url : <http://localhost:8080/admin>

Method : Get

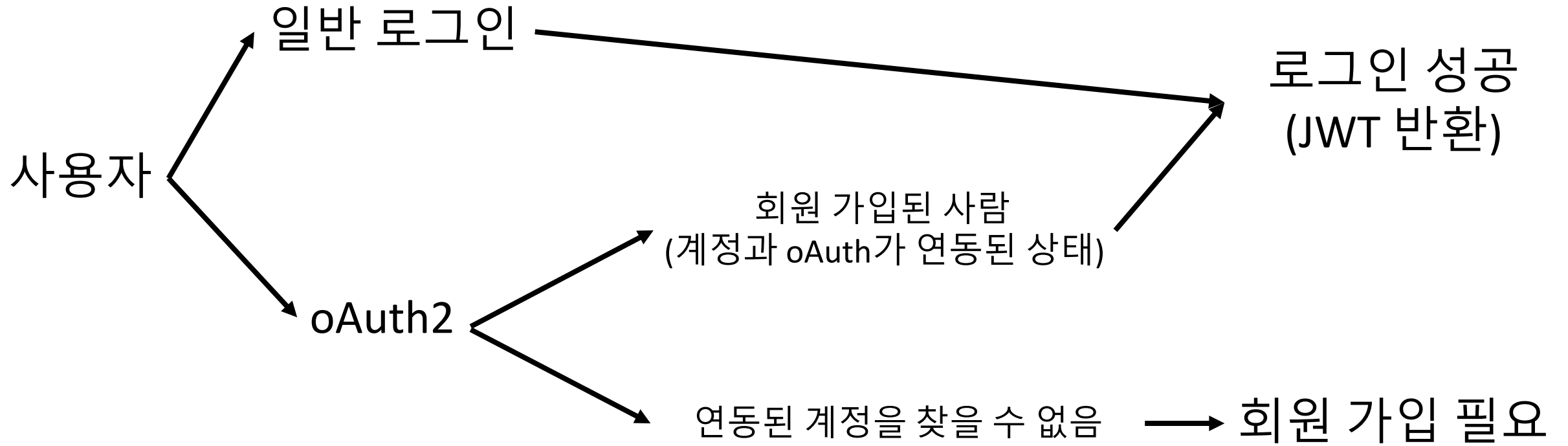
Header : Authorization에 JWT가 포함되어 있어야 합니다.

JWT 인증 실패 또는 접근 권한이 없으면 403 에러가 뜹니다.



OAuth2

# 로그인



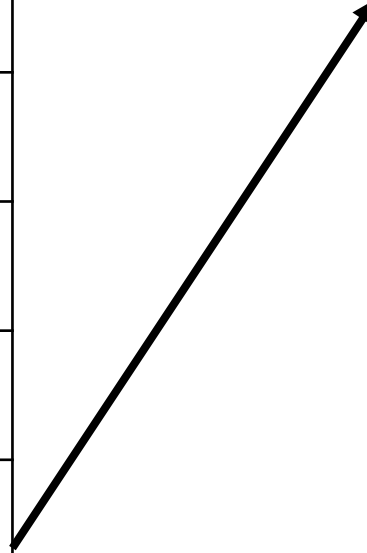
# DB Table 예시

oAuth Table

<b>o_auth2_id (PK)</b>
kakao
google
naver
<b>member_id (FK)</b>

user Table

<b>member_id (PK)</b>
email
password
name
role



# 1. 회원가입시

oAuth table과 user table이 1:1 관계로 생성

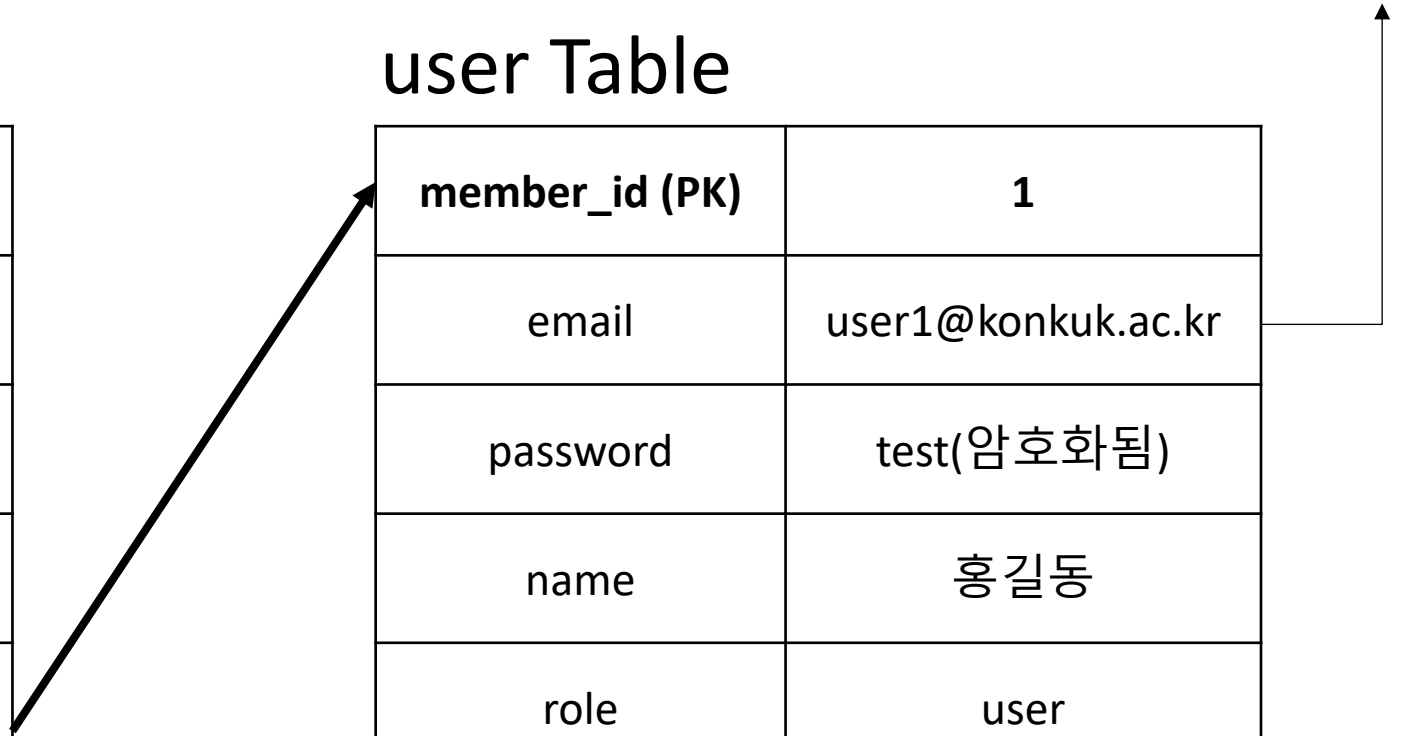
실제 서비스에서는  
user1만 저장됩니다.

oAuth Table

o_auth2_id (PK)	2
kakao	null
google	null
naver	null
member_id (FK)	1

user Table

member_id (PK)	1
email	user1@konkuk.ac.kr
password	test(암호화됨)
name	홍길동
role	user



## 2. OAuth와 연동(kakao)

kakao에서 제공하는 유저id입니다.

oAuth Table

o_auth2_id (PK)	2
kakao	3142534
google	null
naver	null
member_id (FK)	1

user Table

member_id (PK)	1
email	user1@konkuk.ac.kr
password	test(암호화됨)
name	홍길동
role	user

# oAuth2 Api

Url :카카오에서 제공하는 웹 페이지이므로 브라우저에서 접속해주세요.

[https://kauth.kakao.com/oauth/authorize?client\\_id=437bc2fb95b24ca5a80d5763e4619f54&redirect\\_uri=http://localhost:8080/auth/kakao/callback&response\\_type=code](https://kauth.kakao.com/oauth/authorize?client_id=437bc2fb95b24ca5a80d5763e4619f54&redirect_uri=http://localhost:8080/auth/kakao/callback&response_type=code)

Method : Get

2가지 기능

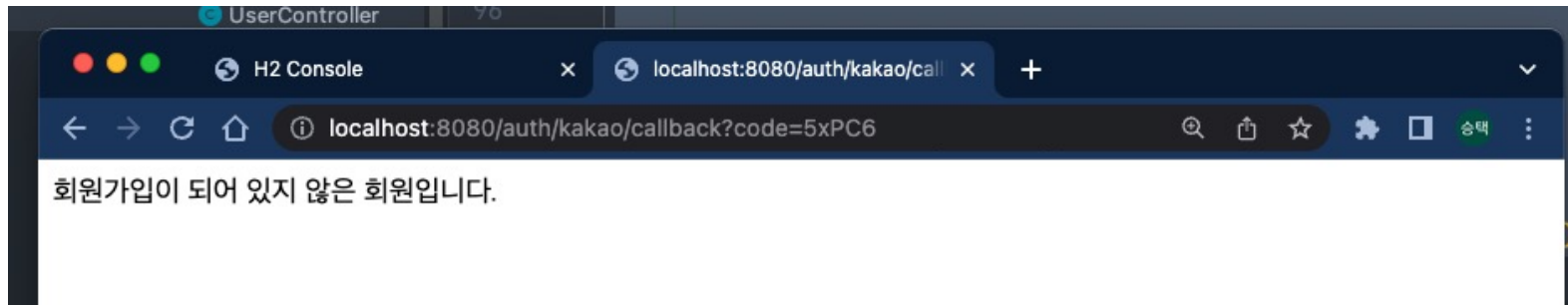
1. 로그인 서비스 : oAuth Table 조회 후 존재하면 연동된 계정으로 로그인 성공.  
이때 계정과 oAuth가 연동된 상태이어야 합니다.
2. 계정과 연동 서비스 : Request header의 Authorization에 JWT가 포함되어 있어야 합니다.

크롬 확장프로그램 ModHeader 사용하시면 request header 조작을 편하게 하실 수 있습니다.

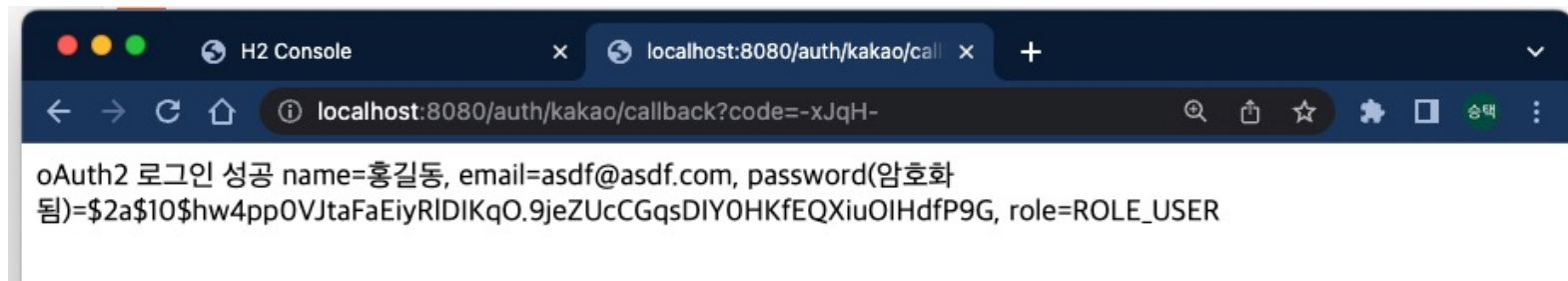
# oAuth2 Api 예시

## 1. 로그인 서비스

### 1-1 계정 없음==해당 oAuth와 연동된 계정 없음

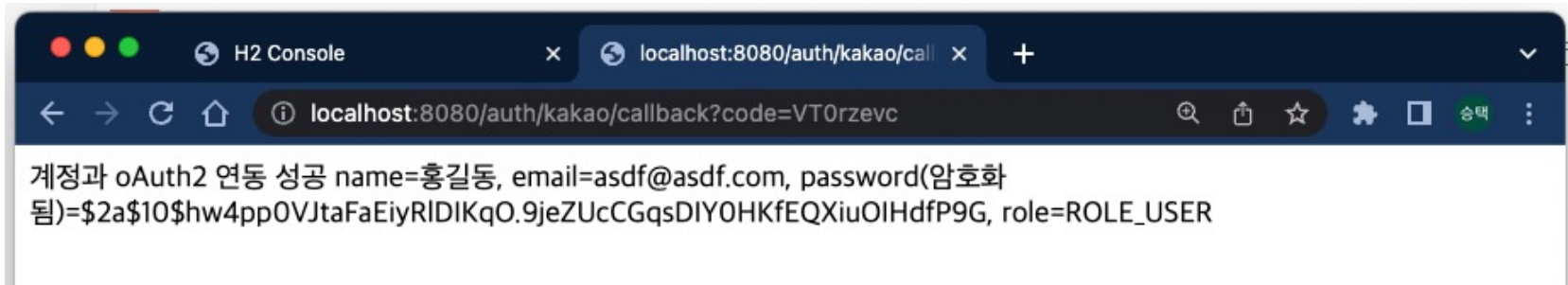


### 1-2 계정 존재==계정과 oAuth 연동된 상태. Response header에 JWT 확인



# oAuth2 Api 예시

2. 계정과 연동 서비스 (header에 JWT가 포함되어 있어야 합니다.)





# API 테스트 순서(종합)

1. OAuth 로그인 시도 -> 로그인 실패
2. 회원가입 (<http://localhost:8080/signup>)
3. 일반 로그인 (<http://localhost:8080/login>)
4. 접근권한이 필요한 서비스 이용 (<http://localhost:8080/user> 또는 [/admin](http://localhost:8080/admin))
5. OAuth를 계정과 연동
6. OAuth 로그인 시도 -> 로그인 성공