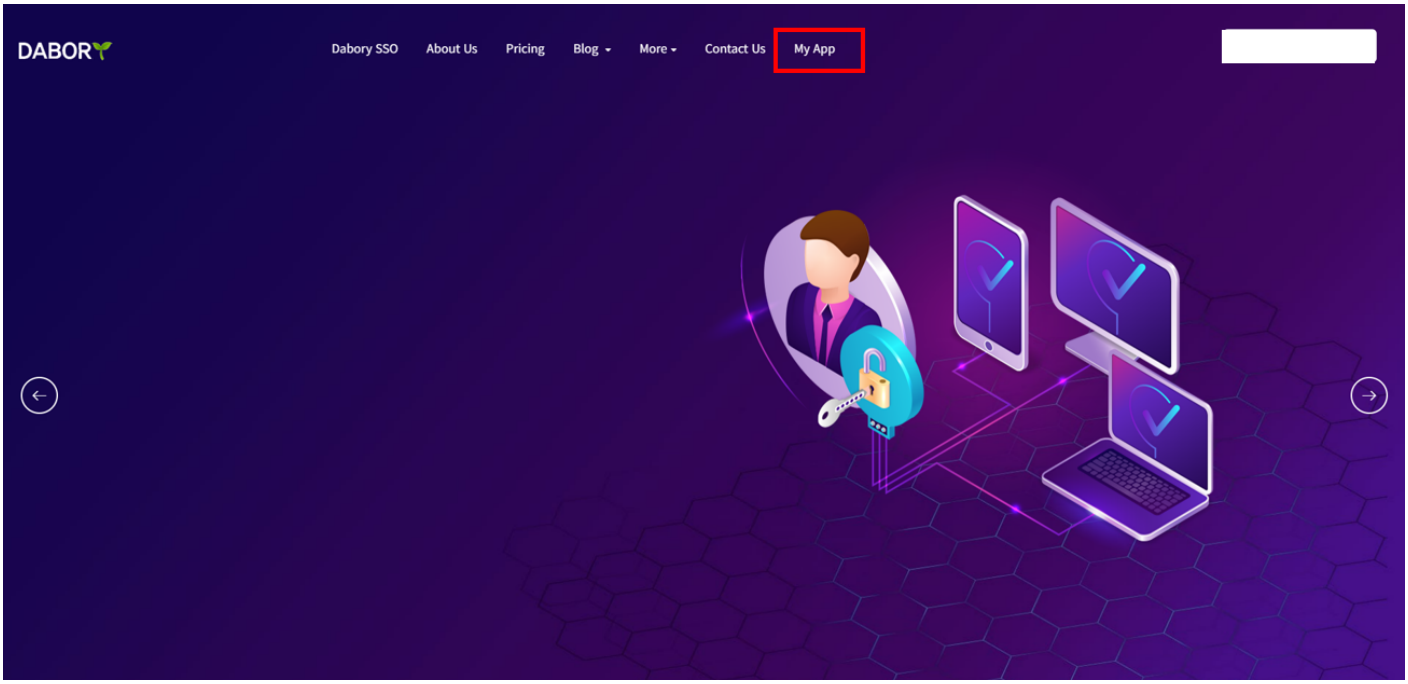
# 공통 API 가이드


- 사전준비 사항
- APP 등록하기
- PAP-API



# APP 등록하기

> 다보리 **SSO**

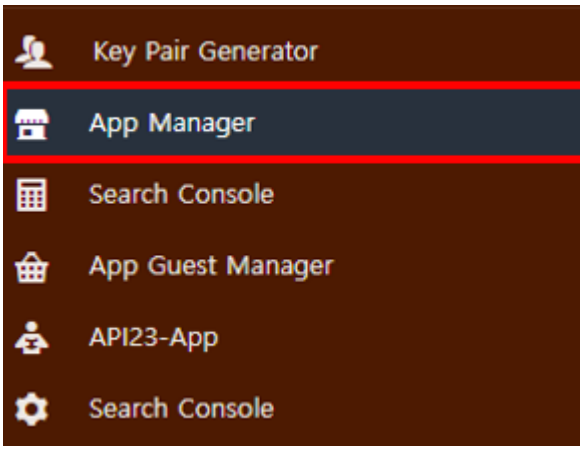
다보리**SSO**는 OAuth 2.0 계정 인증 서비스입니다.

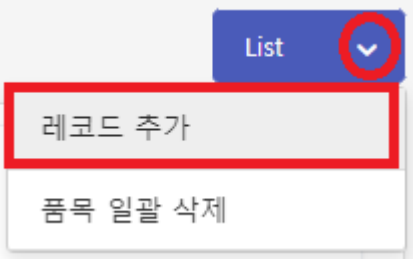
> **APP** 등록하기

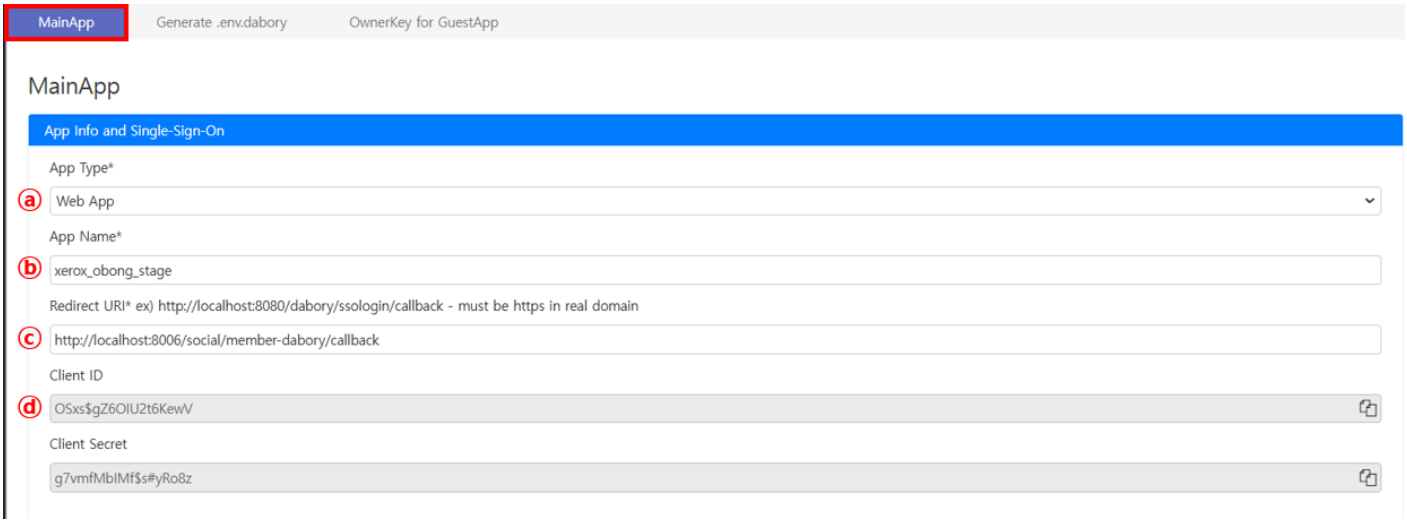
다보리 **API**를 사용하하기 위해서는 다보리 **APP**이 필요합니다. **APP**은 다보리 **API**에 등록한 서비스 정보입니다. 각 **APP**은 다보리 **API** 요청 허가(**Authorizatio**
갖습다보리**SSO**에서 **APP**을 등록하는 방법은 다음과 같습니다.

**1.** 다보리**SSO** 메뉴에서 **My APP** 메뉴를 클릭합니다.

2. 좌측 메뉴에서 App Manager를 클릭합니다.







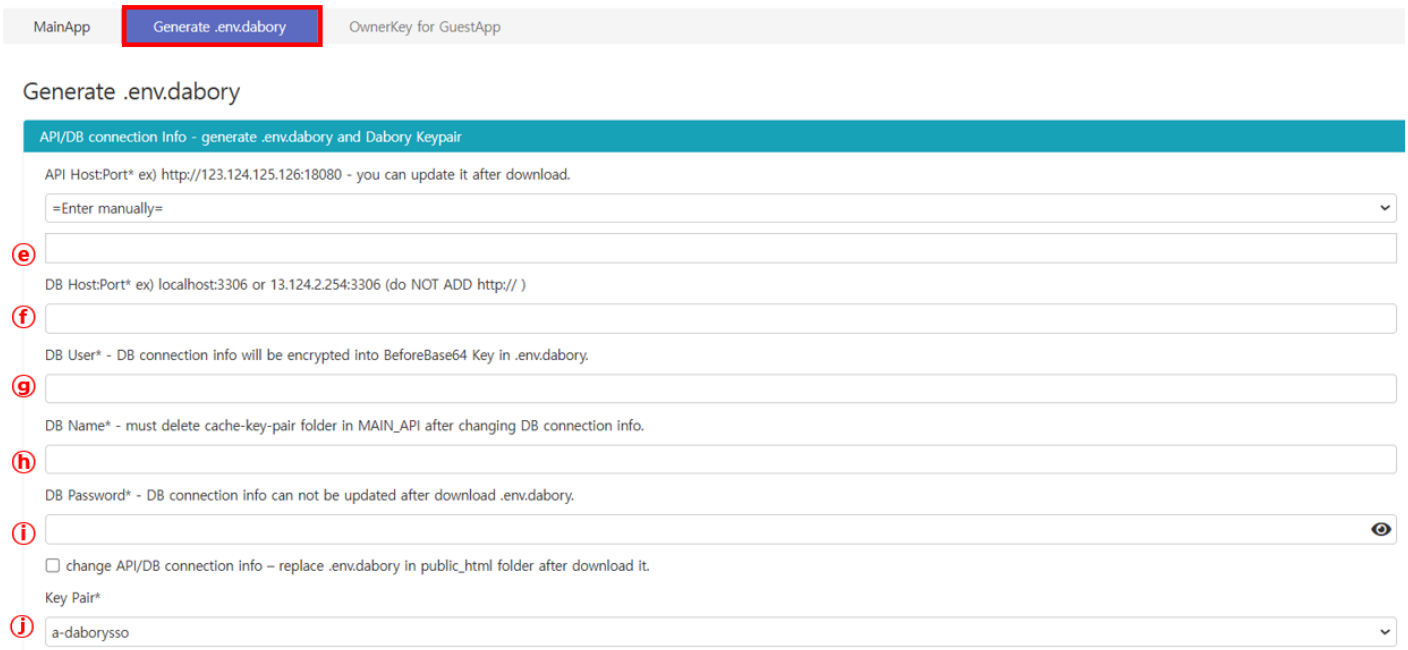
3. 우측 상단의 List 옆 화살표 버튼을 클릭 -> 레코드 추가 버튼을 클릭합니다.

4. 아래 설명을 참고하여 a~h까지 차례대로 입력합니다.

## App info and Single-Sign-On

a App Type : app의 type을 선택합니다.

b App Name : 등록할 app의 name을 작성합니다.



Redirect URI : 등록할 app의 Redirect_URI (account server에서 사용자 인[
callback)

Client ID, Client Secret : app별로 생성되는 고유 key 값입니다.

> **API/DB connection info - generate .env.dabory and Dabory Keypair**

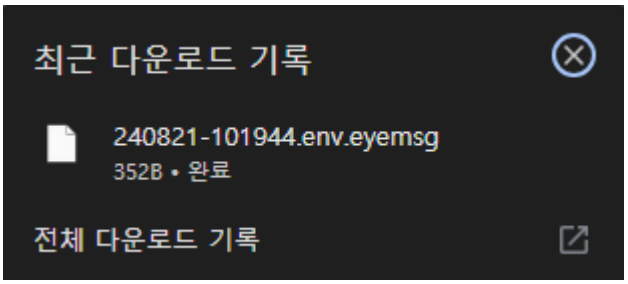
API Host : 연동할 main api 서버의 host 선택

DB Host : 연동할 db 서버의 host

DB User : 연동할 db 서버의 username과

DB Name : 연동할 db 서버의 dbname

DB Password : 연동할 db server의 password



Key Pair : key pair는 어떤걸 선택해도 무방합니다.

6. save 버튼을 클릭하면 아래와 같이 설정파일이 생성됩니다.

7. 편집기를 통해 생성된 파일이 열어 내용을 확인합니다. 생성된 변수들은 다음과 같습니다.

```
MAIN_API_URL=''
MAIN_API_CLIENT_ID=''
MAIN_API_CLIENT_SECRET=''
MAIN_API_BEFORE_BASE64=''
MAIN_API_OWNER_KEY=''
```

8. FRONTEND에서 config 파일에 위 변수들과 값을 추가합니다.

> 그 이후의 과정

* 로그인 버튼 클릭시 백엔드에서 account url의 파라미터 url 생성 (
ex : https://accounts.dabory.com/o/oauth2/authorize?client_id=ysTHfKT4noL-xaJkbc&redirect_uri=https://visionnote.eyerecord.co.kr/wp-login.php&response_type=code&scope=all&state=tUhPBnu5RFOpCPm8IRBMayCSe5FKBRuG) 하여 요청

* app 등록시 app_manager에 등록했던 Redirect_URI에서 callback 함수 구현

* 위 변수들을 통해 MAIN_API_URL에 등록된 dabory main_api server에서 token을 발급(token 발급받는 함수 구현)

* 발급받은 token은 인증이 필요한 모든 request의 header에 포함됩니다.

# PAP-API

> 시작하기 전에

이 문서는 다보리 앱에서 개별 Table 단위의 데이터 처리를 위한 API 구조를 정의하는 방법을 설명하고 있습니다. PAP-API의 구조는 기본적으로 아래와 같습니다.

- **P**ick(단일 데이터 추출)
- **A**ct(데이터의 추가, 수정, 삭제)
- **P**age(다수의 row 추출)

## **API URL** 네이밍 규칙

PAP-API 호출시 데이터를 사용하려는 해당 테이블명에 -pick, -act, -page를 붙여 공통적으로 api url을 네이밍하고 있습니다.

예를들어

- item 테이블에서 특정 상품 하나의 정보를 불러오고 싶다면 url은 item-pick
- member 테이블에서 특정 회원의 정보를 불러오고 싶다면 url은 member-pick
- item 테이블에 특정 상품을 추가, 삭제, 수정하고 싶다면 url은 item-act
- member 테이블에서 특정한 status(휴면회원)인 회원 page를 불러오고 싶다면 member-page
- member 테이블에 특정회원을 추가하고 싶다면 member-act (Id : 양수)
- member 테이블에 특정회원을 삭제하고 싶다면 member-act (Id : 음수)

## **pick-api** 사용예제

쿼리의 WHERE 절에 유니크한 값인 Id로 조회하여 단일데이터 및 다수의 레코드를 추출할 수 있는 api입니다.

php

```
$itemPick = $this->callApiService->callApi([
  'url' => 'item-pick',
  'data' => [
    'Page' => [
            ['UpdatedMd5' => $updatedMd,]
```

```
    ],
  ],
  'headers' => [
  'GateToken' => $this->gateToken['main']
  ]
]);
```

javascript

```
const response = await axios.post('/ajax/get-data', {
  url: 'item-pick',
  data: {
    Page: [
          { Id:  parseInt(window.User['SgroupId']) }
    ]
  }
});
```

## act-api 사용예제

Insert, Update, Delete와 같은 데이터 작업을 수행합니다. Id의 값에 따라 Insert(0), Update(양수), Delete(음수)로 구분합니다.

만약 Id값이 0보다 큰 양수라면 해당 Id값에 해당하는 데이터를 Update합니다.

Delete는 다른 필드 없이 ID 값만으로 처리할 수 있습니다.

php

```
// insert
$itemAct = $this->callApiService->callApi([
          'url' => 'item-act',
          'data' => [
             'Page' => [
                [
                   'Id' => 0, // 0 : insert, 양수: update, 음수 : delete
                   'IgroupId' => 526,
```

```
                'ItemCode' => Str::limit($linkproMd5, 21, ''),
                'ItemSlug' => $linkproMd5,
                'ItemName' => $scrap['ItemName'],
                'SalesPrc' => (string)$scrap['SalesPrice'],
            ]
        ],
    ],
    'headers' => [
        'GateToken' => $this->gateToken['main']
    ]

// update
$itemAct = $this->callApiService->callApi([
    'url' => 'item-act',
    'data' => [
        'Page' => [
            [
                'Id' => 4, // 0 : insert, □□: update, □□ : delete
                'IgroupId' => 526,
                'ItemCode' => Str::limit($linkproMd5, 21, ''),
                'ItemSlug' => $linkproMd5,
                'ItemName' => $scrap['ItemName'],
                'SalesPrc' => (string)$scrap['SalesPrice'],
            ]
        ],
    ],
    'headers' => [
        'GateToken' => $this->gateToken['main']
    ]


// delete
$itemAct = $this->callApiService->callApi([
    'url' => 'item-act',
    'data' => [
        'Page' => [
            [ 'Id' => -4 ] // 0 : insert, □□: update, □□ : delete
        ],
    ],
    'headers' => [
```

```
            'GateToken' => $this->gateToken['main']
        ]
```

javascript

```
const response = await axios.post('/ajax/get-data', {
        url: 'item-act',
        data: {
          Id : 0,
          ItemCode: $(item_form).find('#item-code-txt').val(),
          IgroupId: Number($(item_form).find('#igroup-id-txt').data('id')),
          ItemName: $(item_form).find('#item-name-txt').val(),
          SubName: $(item_form).find('#sub-name-txt').val(),
          ItemSlug: $(item_form).find('#item-slug-txt').val(),
        }
});

const response = await axios.post('/ajax/get-data', {
        url: 'item-act',
        data: {
          Id : 3,
          ItemCode: $(item_form).find('#item-code-txt').val(),
          IgroupId: Number($(item_form).find('#igroup-id-txt').data('id')),
          ItemName: $(item_form).find('#item-name-txt').val(),
          SubName: $(item_form).find('#sub-name-txt').val(),
          ItemSlug: $(item_form).find('#item-slug-txt').val(),
        }
});

const response = await axios.post('/ajax/get-data', {
        url: 'item-act',
        data: {Id : -3}
});
```

## **page-api** 사용예제

하나의 SELECT 쿼리로 테이블의 레코드를 2차원 리스트 형태로 가져옵니다.

Query, Asc, Desc, Limit, Offset 등의 파라미터를 통해 request할 data의 필터링을 설정합니다.

php

```
$this->callApiService->callApi([
  'url' => 'app-guest-page',
  'data' => [
    'PageVars' => [
      'Query' => "app_name = '$appName' and is_on_use = 1",
      'Limit' => 1,
    ]
  ],
'headers' => [
  'GateToken' => $this->gateToken['main']
]
```

javascript

```
let response = await get_api_data('setting-search-page', {
  QueryVars: {
    QueryName: 'igroup',
    FilterName: 'dbr_igroup.id',
 },
  PageVars: {
    Limit: 9999,
    Offset: 0,
  }
})

// get_api_data(url, data)
```