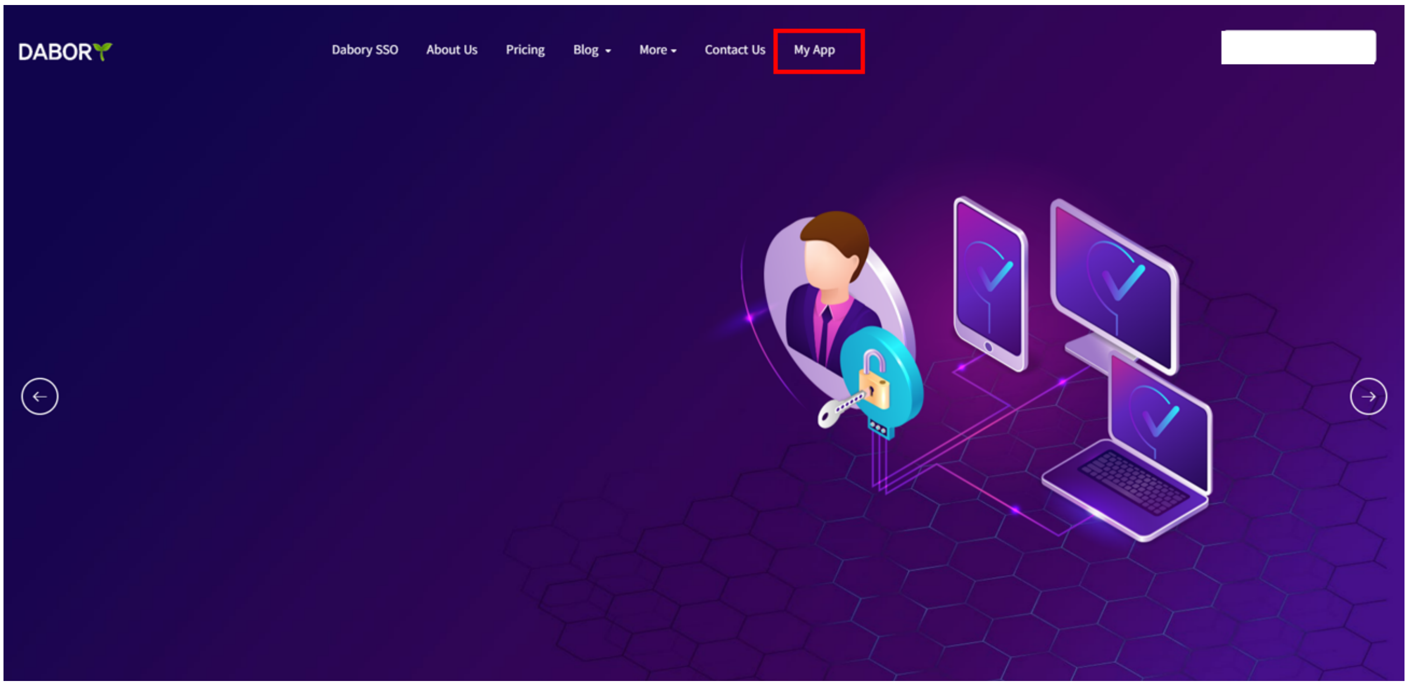
공통 API 가이드

- 사전준비 사항
- APP 등록하기
- PAP-API

[illegible]



APP 등록하기

> 다보리 **SSO**

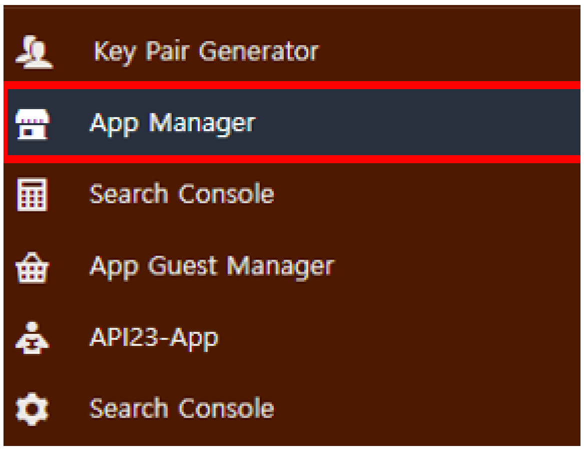
다보리**SSO**는 OAuth 2.0 계정 인증 서비스입니다.

> **APP** 등록하기

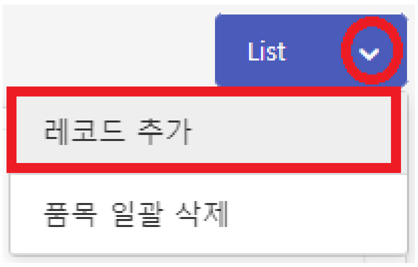
다보리 **API**를 사용하하기 위해서는 다보리 **APP**이 필요합니다. **APP**은 다보리 **API**에 등록한 서비스 정보입니다. 각 **APP**은 다보리 **API** 요청 허가(**Authorizatio**
갖습다보리**SSO**에서 **APP**을 등록하는 방법은 다음과 같습니다.

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

2. 좌측 메뉴에서 App Manager를 클릭합니다.

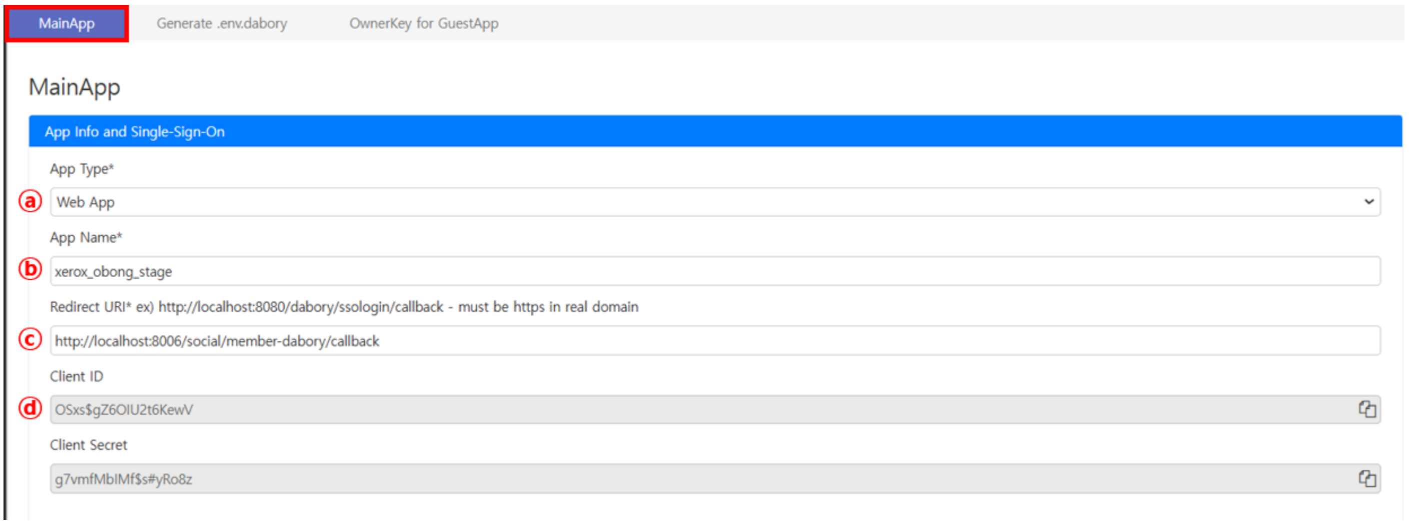


3. □□ □□□ List □ □□□ □□□ □□ -> □□□ □□ □□□ □□□□□.



4. □□ □□□ □□□□ a~h□□ □□□□ □□□□□.

App info and Single-Sign-On



MainApp

App Info and Single-Sign-On

App Type*

ⓐ Web App

App Name*

ⓑ xerox_obong_stage

Redirect URI* ex) http://localhost:8080/dabory/ssologin/callback - must be https in real domain

ⓒ http://localhost:8006/social/member-dabory/callback

Client ID

ⓓ OSxs$gZ6OIU2t6KewV

Client Secret

g7vmfMbIMf$s#yRo8z

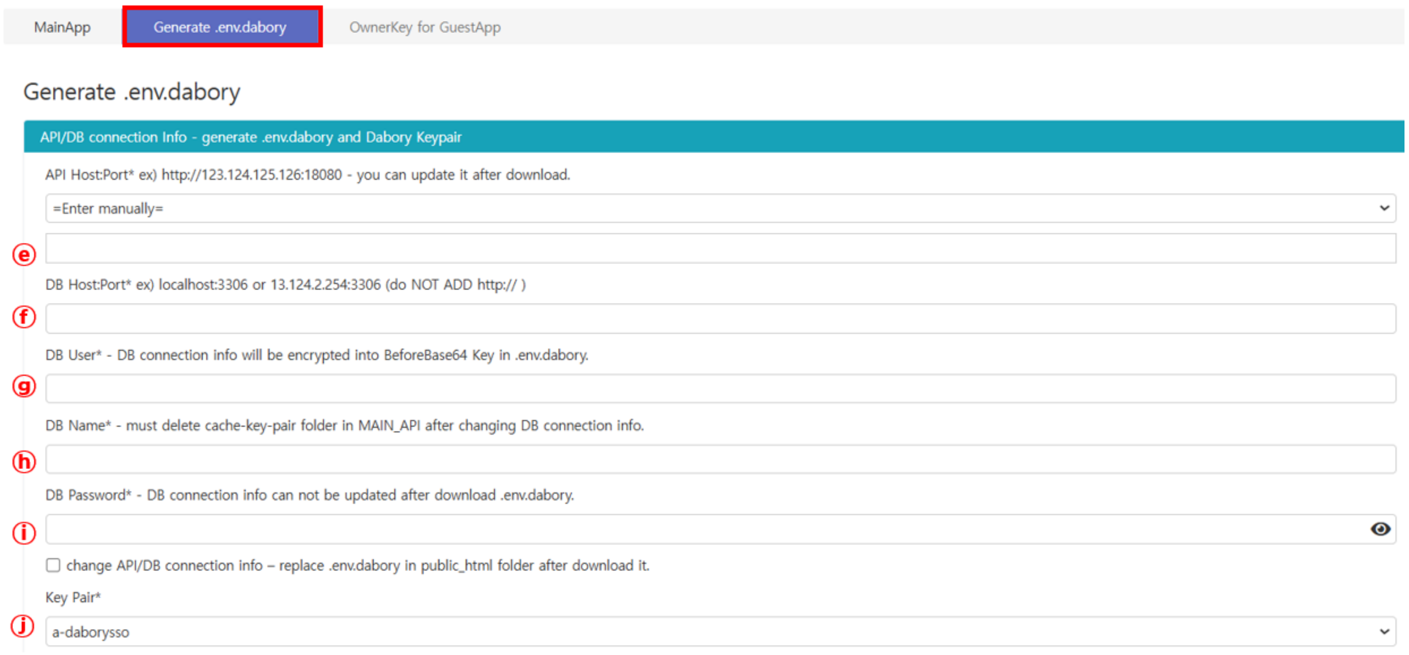
□App Type : app□ type□ □□□□□.

□App Name : □□□ app□ name□ □□□□□.

□Redirect URI : □□□ app□ Redirect_URI (account server□□ □□□ □□ callback)

□Client ID, Client Secret : app□□ □□□□ □□ key □□□□.

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



MainApp | Generate .env.dabory | OwnerKey for GuestApp

Generate .env.dabory

API/DB connection Info - generate .env.dabory and Dabory Keypair

API Host:Port* ex) http://123.124.125.126:18080 - you can update it after download.

=Enter manually=

ⓔ

DB Host:Port* ex) localhost:3306 or 13.124.2.254:3306 (do NOT ADD http://)

ⓕ

DB User* - DB connection info will be encrypted into BeforeBase64 Key in .env.dabory.

ⓖ

DB Name* - must delete cache-key-pair folder in MAIN_API after changing DB connection info.

ⓗ

DB Password* - DB connection info can not be updated after download .env.dabory.

ⓘ

☐ change API/DB connection info – replace .env.dabory in public_html folder after download it.

Key Pair*

ⓙ a-daborysso

□API Host : □□□ main api □□□ host □□

□DB Host : □□□ db □□□ host

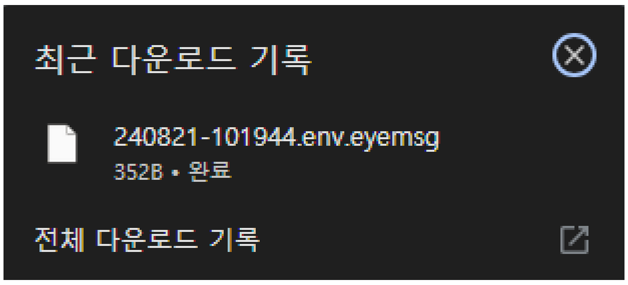
□DB User : □□□ db □□□ username□

□DB Name : □□□ db □□□ dbname

□DB Password : □□□ db server□ password

□Key Pair : key pair□ □□□ □□□□ □□□□□.

6. save □□□ □□□□ □□□ □□ □□□□□ □□□□□.


최근 다운로드 기록

240821-101944.env.eyemsg
352B • 완료

전체 다운로드 기록

7. □□□□ □□ □□□ □□□ □□ □□□ □□□□□. □□□ □□□□ □□□ □□□□.

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

8. FRONTEND□□ config □□□ □ □□□□ □□ □□□□□.

□ □□□ □□

* □□□ □□ □□□ □□□□□ account url□ □□□□ url □□ (
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=tUhPBnu5RFOpCPm8lRBMayCSe5FKBRuG) □□ □□

* app □□□ app_manager□ □□□□ Redirect_URI□□ callback □□ □□

* □ □□□□ □□ MAIN_API_URL□ □□□ dabory main_api server□□ token□ □□(token □□□□ □□ □□)

* □□□□ token□ □□□ □□□ □□ request□ header□ □□□□□.

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 [illegible]

[illegible] SELECT [illegible] [illegible] [illegible] 2[illegible] [illegible] [illegible] [illegible].

Query, Asc, Desc, Limit, Offset [illegible] [illegible] [illegible] request[illegible] data[illegible] [illegible] [illegible].

php

```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

```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)
```