

□□ API □□□

- [□□□□ □□](#)
- [APP □□□](#)
- [PAP-API](#)



APP 管理

“

SSO

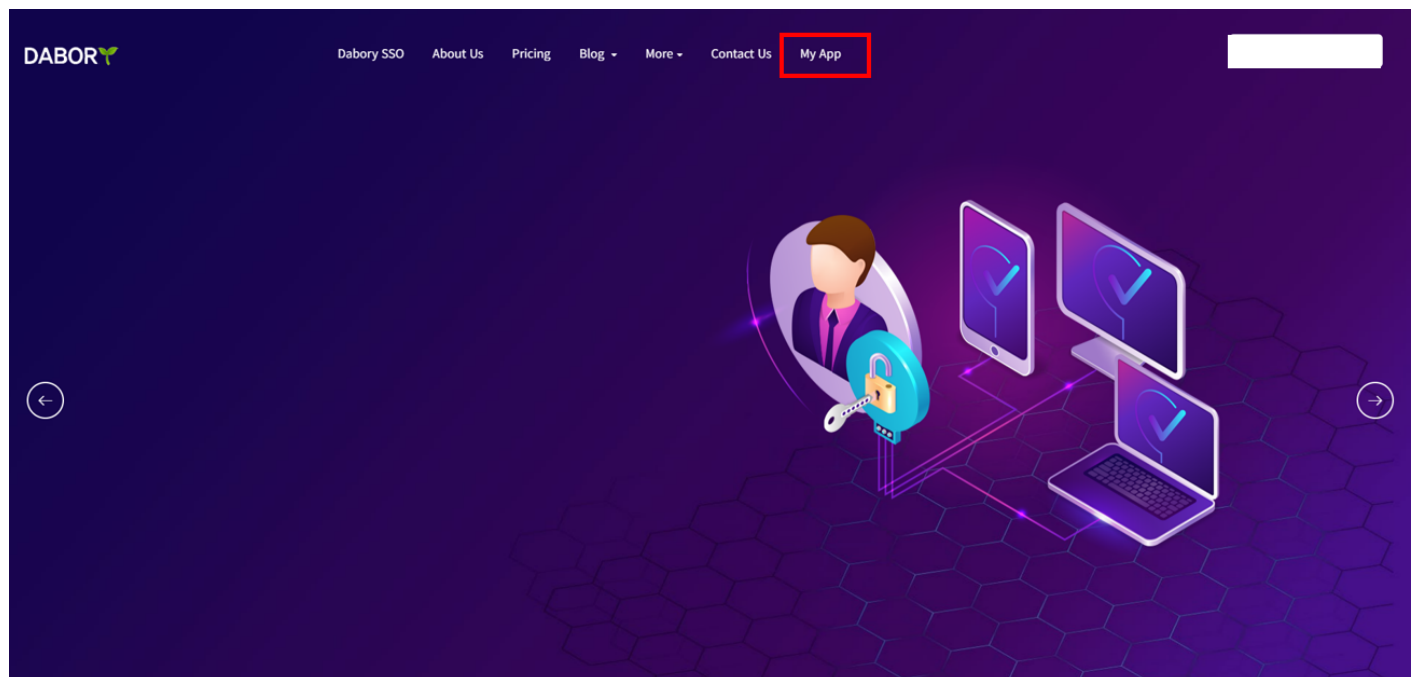
SSO 与 OAuth 2.0 的区别

“

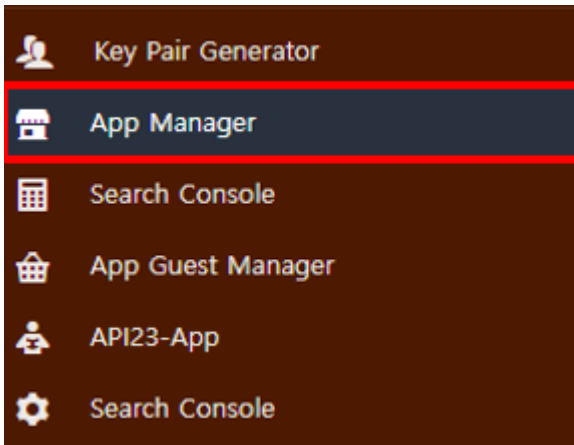
APP 管理

API 管理 与 APP 管理。APP 管理 API 管理 授权 (Authorization) SSO 与 APP 管理 区别

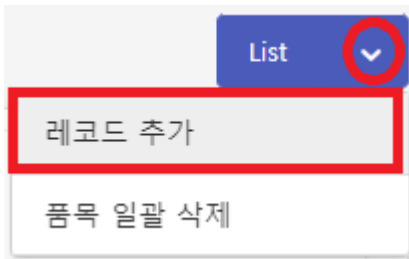
1. SSO 与 My APP 管理



2. App Manager 管理



3. List 버튼을 클릭하여 List 버튼을 클릭합니다.



4. 레코드 추가 버튼을 클릭하여 a~h 버튼을 클릭합니다.

App info and Single-Sign-On

MainApp
Generate .env.dabory
OwnerKey for GuestApp

MainApp
App Info and Single-Sign-On

App Type*
a Web App

App Name*
b xerox_obong_stage

Redirect URI* ex) http://localhost:8080/dabory/ssologin/callback - must be https in real domain
c http://localhost:8006/social/member-dabory/callback

Client ID
d OSxs\$gZ6OIU2t6KewV

Client Secret
g7vmfMbIMf\$yRo8z

App Type : app type을 선택합니다.

App Name : app name을 입력합니다.

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

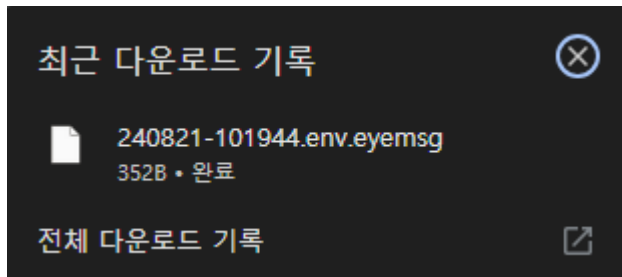
DB User : db username

DB Name : db dbname

DB Password : db server password

Key Pair : key pair을 생성합니다.

6. save 버튼을 클릭하여 저장합니다.



7. MAIN_API_URL, MAIN_API_CLIENT_ID, MAIN_API_CLIENT_SECRET, MAIN_API_BEFORE_BASE64, MAIN_API_OWNER_KEY를 입력합니다.

```
MAIN_API_URL=""
MAIN_API_CLIENT_ID=""
MAIN_API_CLIENT_SECRET=""
MAIN_API_BEFORE_BASE64=""
MAIN_API_OWNER_KEY=""
```

8. FRONTEND의 config 파일을 수정합니다.

config

* account url은 account url을 입력합니다 (

ex : [https://accounts.dabory.com/o/oauth2/authorize?client_id=ysTHfKT4noL-](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=tUhpBnu5RF0pCPm8IRBMayCSe5FKBRuG)

[xajkbc&redirect_uri=https://visionnote.eyerecord.co.kr/wp-](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=tUhpBnu5RF0pCPm8IRBMayCSe5FKBRuG)

[login.php&response_type=code&scope=all&state=tUhpBnu5RF0pCPm8IRBMayCSe5FKBRuG](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=tUhpBnu5RF0pCPm8IRBMayCSe5FKBRuG))

* app의 app_manager의 Redirect_URI는 callback URL을 입력합니다

* MAIN_API_URL은 dabory main_api server의 token URL(token URL을 입력)

* token은 request header에 입력합니다.

PAP-API

API 概要

このAPIは Table 形式でデータを API 経由で取得するための API です。PAP-API の詳細は以下の通りです。

- **Pick**(取得する ID)
- **Act**(操作する ID, 操作種別, 操作対象)
- **Page**(取得する row 数)

API URL 一覧

PAP-API の API URL は以下の通りです。-pick, -act, -page はオプションで指定します。

例

- item 取得 API URL: item-pick
- member 取得 API URL: member-pick
- item 操作 API URL: item-act
- member 操作 API URL: member-act (Id : ID)
- member 操作 API URL: member-act (Id : ID)

pick-api 概要

この API は WHERE 句で指定した ID を取得するための API です。

php

```
$itemPick = $this->callApiService->callApi([
    'url' => 'item-pick',
    'data' => [
        'Page' => [
            ['UpdatedMd5' => $updatedMd5]
```

```

    ],
    ],
    '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 □□□□

Insert, Update, Delete □ □ □□ □□ □□□□. Id □ □ □ Insert(0), Update(□□), Delete(□□) □ □□□□.

□□ Id □□ 0 □ □ □□□ □ □□□ □□□□ Id □□ □□□ □□□□ Update□□□.

Delete □ □ □ □ □ ID □□□□ □□ □ □□□□.

php

```

// insert
$itemAct = $this->callApiService->callApi([
    'url' => 'item-act',
    'data' => [
        'Page' => [
            [
                'Id' => 0, // 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']
]

```

// 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

SELECT FROM 2

Query, Asc, Desc, Limit, Offset request data

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)
```