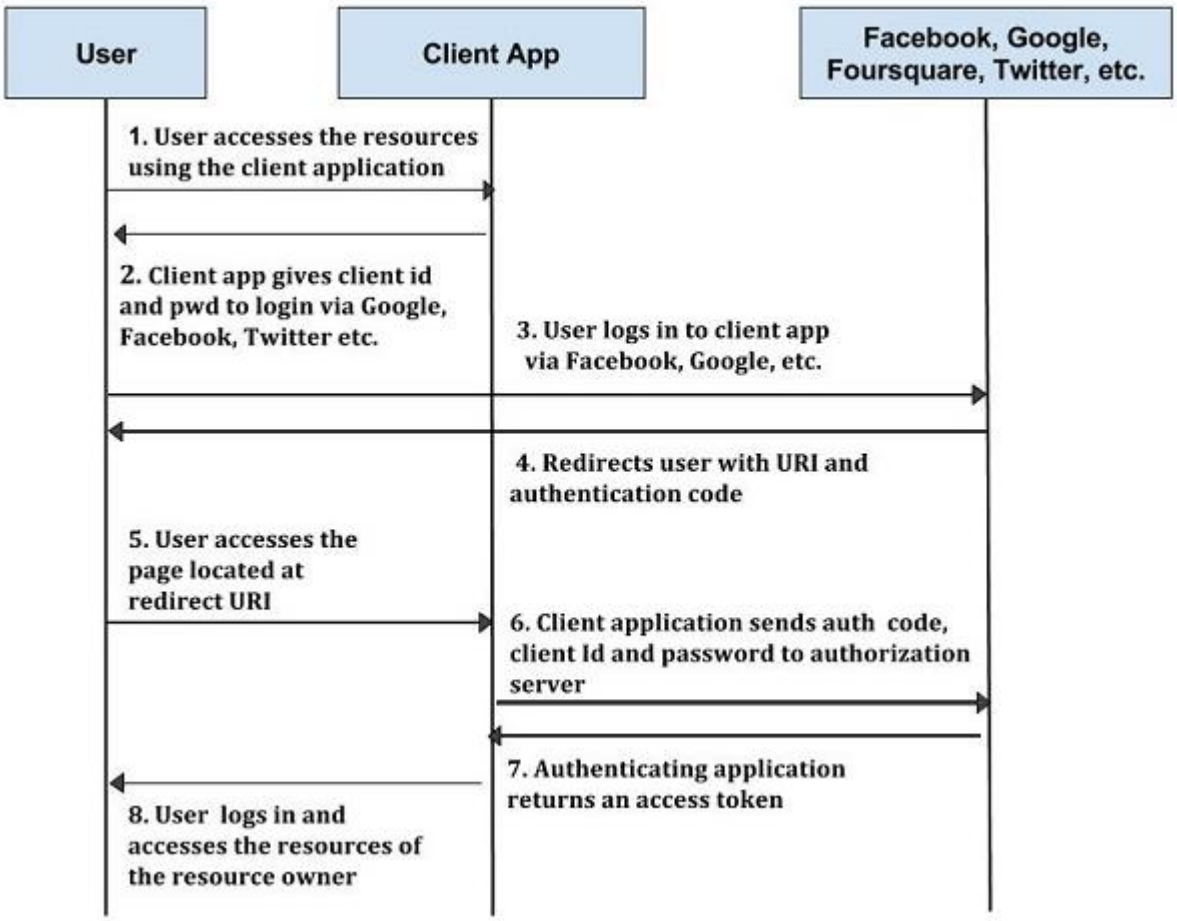
1. oAuth

   인터넷 사용자들이 비밀번호를 제공하지 않고 다른 웹 사이트 상의 자신들의 정보에 대한 웹사이트나 어플리케이션의 접근 권한을 부여할 수 있는 공통적인 수단으로서 사용되는 접근 위임을 위한 개방형 표준

   - concept



2. oAuth2 - PHP Source

- Grant Types : 인증 유형

  - Authorization Code
    - 가장 일반적인 유형
    - 3-Legged oAuth를 구현하고 사용자가 클라이언트에게 authorization_code를 부여한다.
    - grant_type : authorization_code
  - User Credentials
    - 리소스 소유자의 사용자 이름과 암호가 요청의 일부로 제출되고 인증 성공시 토큰이 발급된다.
    - grant_type : password
  - Client Credentials
    - 클라이언트는 자신의 자격 증명을 사용하여 액세스 토큰을 직접 사용할 수 있다.
    - grant_type : client_credentials
  - Refresh Token
    - 클라이언트는 refresh_token을 제출하여 access_token이 만료된 경우 새로운 access_token을 발급받을 수 있다.
    - grant_type : refresh_token
  - Implicit
    - Authorization Code와 유사하지만 인증 요청에서 authorization_code가 반환되는 것이 아니라 토큰이 클라이언트에게 반환된다.
    - 클라이언트 자격 증명을 안전하게 저장할 수 없는 클라이언트 장치에서 일반적으로 사용된다.
    - response_type : token
    - 성공 : redirect URI에 포함되어 반환
  - JWT Bearer
    - 클라이언트는 endpoint에 대한 요청에서 JWT를 제출할 수 있다.
    - 이후 access_token이 직접 반환된다.

- Controller
  - Authorize : Authorization Code, Implicit Grant_type을 위한 Controller
    - handleAuthorizeRequest
    - validateAuthorizeRequest
  - Resource : 자원 요청은 oAuth2 인증이 필요
    - verifyResourceRequest
    - getAccessTokenData
  - Token : Token endpoint로 access_token을 반환
    - grantAccessToken
    - handleTokenRequest
- Storage
  - 여러 DB 스토리지를 지원하는 중에 Mysql, Redis 사용
    - mysql : 사용자 정보
    - Redis : 휘발성 발급 토큰

3. Source

- 구성
  - php 8, Laravel 8, oAuth2 Library, nginx
- 라우팅

```
// client, redirect url 확인
Route::get('/oauth/valid','App\Http\Controllers\oAuth\Auth@valid');
// authorization code 발급
Route::post('/oauth/auth','App\Http\Controllers\oAuth\Auth@main');
// token endpoint :: access_token 발급
Route::post('/oauth/token','App\Http\Controllers\oAuth\Token@main');
// 인증 확인
Route::post('/oauth/token/verify','App\Http\Controllers\oAuth\Token@verify');
```

- app/Library/oAuth2/Server.php

```
namespace App\Library\oAuth2;

use OAuth2\HttpFoundationBridge\Response as BridgeResponse;
use OAuth2\Server as OAuth2Server;
use OAuth2\Storage\Pdo;
use OAuth2\Storage\Memory;
use OAuth2\Storage\Redis;
use OAuth2\OpenID\GrantType\AuthorizationCode;
use OAuth2\GrantType\UserCredentials;
use OAuth2\GrantType\RefreshToken;
use Predis\Client;

class Server{
    public function __construct(){
        $this->setup();
    }
    public function setup(){
        // Storage 생성 -
- 이후 setup parameter를 받아 DB 선택 ( ks_admin, ks_member, eyeon )
        $aDbConn = Array(
            'dsn' => 'mysql:host=1.234.15.178;dbname=ksadmin',
            'username' => 'root',
            'password' => 'apitest!!'
        );
        $oPredis = new Client();

        // Mysql table name :
            custom 가능
        $aPdoConfig = Array(
            'client_table' => 'oauth_clients',
            'access_token_table' => 'oauth_access_tokens',
            'refresh_token_table' => 'oauth_refresh_tokens',
```

```
            'code_table' => 'oauth_authorization_codes',
            'user_table' => 'oauth_users',
            'jwt_table'  => 'oauth_jwt',
            'jti_table'  => 'oauth_jti',
            'scope_table'  => 'oauth_scopes',
            'public_key_table'  => 'oauth_public_keys',
        );
        $oStorageMaria = new Pdo($aDbConn, $aPdoConfig);

        // redis key prefix : custom 가능
        $aRedisConfig = Array(
            'client_key' => 'oauth_clients:',
            'access_token_key' => 'oauth_access_tokens:',
            'refresh_token_key' => 'oauth_refresh_tokens:',
            'code_key' => 'oauth_authorization_codes:',
            'user_key' => 'oauth_users:',
            'jwt_key' => 'oauth_jwt:',
            'scope_key' => 'oauth_scopes:',
        );
        $oStorageRedis = new Redis($oPredis, $aRedisConfig);

        // grant type 배열 생성
        $aGrantTypes = Array(
            'user_credentials'   => new UserCredentials($oStorageMaria),
            'authorization_code' => new AuthorizationCode($oStorageRedis),
            'refresh_token'      => new RefreshToken($oStorageRedis, Array(
                'always_issue_new_refresh_token' => true
            )),
        );

        // instantiate the oauth server
        $this->oServer = new OAuth2Server(
            Array(
                'access_token'       => $oStorageRedis,
                'authorization_code' => $oStorageRedis,
                'refresh_token'      => $oStorageRedis,
                'jwt_bearer'         => $oStorageRedis,
                'scope'              => $oStorageMaria,
                'user_credentials'   => $oStorageMaria,
                'user_claims'        => $oStorageMaria,
                'client_credentials' => $oStorageMaria,
                'client'             => $oStorageMaria,
                'public_key'         => $oStorageMaria,
            ),
            Array(
                'access_lifetime' => 1800,  // access_token 30 분
                'refresh_token_lifetime' => 172800, // refresh_token 2 일
```

```
                'use_jwt_access_tokens' => true,
                'jwt_extra_payload_callable' => function ($sClient_id, $sUser_id, $sScope =
null, $aCustom = Array()){
                    return Array('custom' => 'make your own jwt payload');
                },

                'enforce_state' => true,
                'allow_implicit' => true,
                'use_openid_connect' => true,
                'issuer' => $_SERVER['HTTP_HOST'],
            ),
            $aGrantTypes
        );

        $this->oServer->addStorage($this->getKeyStorage(), 'public_key');

        // 브릿지 응답
        $this->oResponse = new BridgeResponse();
    }

    private function getKeyStorage()
    {
        $sPublicKey  = file_get_contents($this->getProjectRoot().'/data/pubkey.pem');
        $sPrivateKey = file_get_contents($this->getProjectRoot().'/data/privkey.pem');

        // create storage
        $oKeyStorage = new Memory(Array('keys' => Array(
            'public_key'  => $sPublicKey,
            'private_key' => $sPrivateKey,
        )));

        return $oKeyStorage;
    }

    private function getProjectRoot()
    {
        return dirname(dirname(dirname(__DIR__)));
    }
}
```

- Controller
  - app/Http/Controller/oAuth/Auth.php

```php
namespace App\Http\Controllers\oAuth;

use Illuminate\Http\Request;
use App\Http\Controllers\Controller;

use App\Library\oAuth2\Server;
use App\Models\User;

class Auth extends Controller
{
    public function __construct(){
        $this->oServer = new Server();
    }

    /**
    * @brief authorization code 생성 ( 비밀번호가 노출되기 때문에 POST 변경 )
    * @details
    * @author sjlee4628
    * @param
    * @return
    */
    public function main(Request $oRequest){
        $oServer = $this->oServer->oServer;
        $oResponse = $this->oServer->oResponse;

        $aParams = $oRequest->all();

        // 받아온 id, pass 값으로 authorization_code 를 보내줄 지 말지를 결정한다! -- 추가 필요
        $bAuthorized = false;
        $sId = $aParams['id'] ? $aParams['id'] : '';
        $sPw = $aParams['passwd'] ? $aParams['passwd'] : '';
        $aWhere = Array('ka_id' => $sId, 'ka_use_flag' => 'Y');
        $aUser = User::where($aWhere)->first();
        if ($aUser && $aUser['ka_passwd'] == hash("sha256", $sPw)){
            // 로그인 페이지 리다이렉트 필요
            $bAuthorized = true;
        }

        $oAuthorizationCode = $oServer-
>handleAuthorizeRequest($oRequest, $oResponse, $bAuthorized, $sId);

        print_r($oAuthorizationCode);

        //return $oAuthorizationCode;
```

```
    }

    /**
    * @brief 최초 client_id 검증
    * @details
    * @author sjlee4628
    * @param
    * @return
    */
    public function valid(Request $oRequest){
        $oServer = $this->oServer->oServer;
        $oResponse = $this->oServer->oResponse;

        // 검증
        if (!$oServer->validateAuthorizeRequest($oRequest, $oResponse)) {
            return $oServer->getResponse();
        }

        $aParams = $oRequest->all();

        $aData = Array(
            'response_type' => $aParams['response_type'],
            'redirect_uri' => $aParams['redirect_uri'],
            'state' => $aParams['state'],
            'client_id' => $aParams['client_id'],
            'scope' => $aParams['scope'],
        );

        // front 페이지로 redirect
        return redirect()->to('/oauth/login?'.http_build_query($aData));
    }
}
```

- app.Http/Controller/oAuth/Token.php

```
namespace App\Http\Controllers\oAuth;

use Illuminate\Http\Request;
use Symfony\Component\HttpFoundation\Response;
use App\Http\Controllers\Controller;

use App\Library\oAuth2\Server;

class Token extends Controller
{
    public function __construct(){
        $this->oServer = new Server();
    }
```

```php
    /**
    * @brief access_token, refresh_token 발급
    * @details
    * @author sjlee4628
    * @param
    * @return
    */
    public function main(Request $oRequest){
        $oServer = $this->oServer->oServer;
        $oResponse = $this->oServer->oResponse;

        $oAccessToken = $oServer->handleTokenRequest($oRequest, $oResponse);

        return $oAccessToken;
    }

    /**
    * @brief 토큰 확인
    * @details
    * @author sjlee4628
    * @param
    * @return
    */
    public function verify(Request $oRequest){
        $oServer = $this->oServer->oServer;
        $oResponse = $this->oServer->oResponse;

        // access_token(JWT) 를 통한 유효성 검사 필요

        if (!$oServer->verifyResourceRequest($oRequest, $oResponse)) {
            $aResponse = Array('Active' => 'F');
            $iCode = 401;
        } else {
            $aResponse = Array('Active' => 'T');
            $iCode = 200;
        }
        $oResult = new Response(json_encode($aResponse), $iCode);
        return $oResult;
    }
}
```

- Models
  - app/Models/User.php

```php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class User extends Model
{
    use HasFactory;
    protected $connection = 'ksadmin';
    protected $table ='ks_admin';
}
```

- Vendor 수정
  - vendor/Laravel/framework/src/Illuminate/Http/Request.php

```php
// 추가
use OAuth2\RequestInterface;

// implements 추가
class Request extends SymfonyRequest implements Arrayable, ArrayAccess, RequestInterface
{
    // ... 기존 내용
    /***************for oAuth request implement**************/
    public function query($name, $default = null)
    {
        return $this->query->get($name, $default);
    }

    public function request($name, $default = null)
    {
        return $this->request->get($name, $default);
    }

    public function server($name, $default = null)
    {
        return $this->server->get($name, $default);
    }

    public function headers($name, $default = null)
    {
        return $this->headers->get($name, $default);
    }

    public function getAllQueryParameters()
    {
```

```
        return $this->query->all();
    }
    /**************for oAuth request implement*************/
}
```

- vendor/bshaffer/oauth2-server-httpfoundation/src/OAuth2/HttpFoundationBridge/Response.php

```
// default 누락 추가
// public function setRedirect($statusCode = 302, $url, $state = null, $error = null, $erro
rDescription = null, $errorUri = null)
public function setRedirect($statusCode = 302, $url = '', $state = null, $error = null, $er
rorDescription = null, $errorUri = null)
```

- vendor/bshaffer/oauth2-server-php/src/OAuth2/Storage/Pdo.php

```
// Grant_type user_credentials 의 경우 암호화 sha256 변경
protected function hashPassword($password)
{
    return hash('sha256', $password);
}
```

4. 사용 예시

- Authorization Code

```
$ curl -XPOST http://oauth.server.com/oauth/auth -d 'response_type=code&redirect_uri=xyz&state=123&client_id=ksadmin&id=sjlee4628&passwd=testpass'
```

```
response

    ...

    302

    ...

    location xyz?code=296b88d372d1ad1f259ac0b6ec9eea9bed0e8646&state=123

    ...

$ curl -XPOST http://oauth.server.com/oauth/token -

d 'redirect_uri=xyz&state=123&client_id=ksadmin&client_secret=ksadmin&grant_type=authorization_code&code=296b88d372d1ad1f259ac0b6ec9eea9bed0e8646'

response

    {

        "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJjdXN0b20iOiJtYWtlIHlvdXIgb3duIGp3dCBwYXlsb2FkIiwiaWQiOiJjYjM1ODYyMGJlMzVmYWIwNmE0M2

E3Nzk2OTI3NzEzZGEwMTE5NmUzIiwianRpIjoiY2IzNTg2MjBiZTM1ZmFiMDZhNDNhNzc5NjkyNzcxM2RhMDExOTZlMyIsImlzcyI6ImxvY2FsaG9zdDo4MDgyIiwiYXVkIjoia3NhZG1pbiIsIn

N1YiI6InNqbGVlNDYyOCIsImV4cCI6MTYyNTU2MzAyNywiaWF0IjoxNjI1NTYxMjI3LCJ0b2tlbl90eXBlIjoiYmVhcmVyIiwic2NvcGUiOm51bGx9.AX0Qnm4sEBJ9YX95G9_l1hSCp3yLRBb5Y

ugxhiAfEwBEOuboF_fAB5C0d8c6v0qQ3-FGrd7PU6BJXsWmOiZWNjN_jUJSII0uQvaAgxmXyryv9QDp3c9O_el-B7bXf97GqFnXVZWwFYq4vaMeiesxv9-X2g9_om-

BiVYIJvZG00_0YIKDuhKsgt-

fiRTYNKFukhYggVAKR1m3rHTmnQnUX63ni5nTLrgQT1akJr_K8Q9VrfLaLPAYuh_uWxmKhrVKznmuXzLbyb4KiPdEfzaXLfKmXIuRaWtOrA6leeUkNexY_kZwRiZjYfkhTJvg_CT0uv-kF-

ro2onZoedIU7SVCA",

        "expires_in": 1800,

        "token_type": "bearer",

        "scope": null,

        "refresh_token": "680a9936950be3ef95d7076711868efaf28f7800"

    }
```

- User Credentials

```
$ curl -XPOST http://oauth.server.com/oauth/token -d 'client_id=ksadmin&client_secret=ksadmin&grant_type=password&username=sjlee4628&password=testpass'

response

    {

        "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJjdXN0b20iOiJtYWtlIHlvdXIgb3duIGp3dCBwYXlsb2FkIiwiaWQiOiIzMzRkOTlkMTYwNjJjN2EwMTRmY2JmMjF

kNzdjYzUxMTU1MTdiZWQ4IiwianRpIjoiMzM0ZDk5ZDE2MDYyYzdhMDE0ZmNiZjIxZDc3Y2M1MTE1NTE3YmVkOCIsImlzcyI6ImxvY2FsaG9zdDo4MDgyIiwiYXVkIjoia3NhZG1pbiIsInN1YiI6InNq

bGVlNDYyOCIsImV4cCI6MTYyNTU2MzIzNSwiaWF0IjoxNjI1NTYxNDM1LCJ0b2tlbl90eXBlIjoiYmVhcmVyIiwic2NvcGUiOm51bGx9.PRaCNY7H9mvKJRv_ppf9bcZmp61r3RKEqurZXAdObWAiG6ES

ygjyWVbg49dpGJV5l9WEbGaOjVPWltgLQjGvjzBrY-fjET9Sz2ZDhAYXhezc-v695nQHq341rUyYjHe7ECgtjcY4-o33-

EUxdD9UwhdfQU6PxKxWeNYnDep5TlBHiM6tkAxuinXEKB2aSmPg39mdgkP6FGfvjPVpe5joAjknRiU_KsZCwXrN5cNrAbHv6mhbap1qVL2-

2oqdqRINnX3dQHG6pJ_wfP6zKHoifrS55TZQshtxpZHUEe0lgXB1kBO8eLNpZxbJD6Jf04M53Sh2AmwfVR2dhAJCknDfgQ",

        "expires_in": 1800,

        "token_type": "bearer",

        "scope": null,

        "refresh_token": "2952de565b0cca631fb720a25895ba57afcd7b3b"

    }
```

- refresh_token

```
$ curl -XPOST http://oauth.server.com/oauth/token \

            -d 'grant_type=refresh_token&refresh_token=2952de565b0cca631fb720a25895ba57afcd7b3b&client_id=ksadmin&client_secret=ksadmin'

response

    {

        "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJjdXN0b20iOiJtYWtlIHlvdXIgb3duIGp3dCBwYXlsb2FkIiwiaWQiOiI1YmY0OTZlNzgzMGVmMjZhZWYwYmU3N2R

jMzIyNDFiZGM1YzllZmZlIiwianRpIjoiNWJmNDk2ZTc4MzBlZjI2YWVmMGJlNzdkYzMyMjQxYmRjNWM5ZWZmZSIsImlzcyI6ImxvY2FsaG9zdDo4MDgyIiwiYXVkIjoia3NhZG1pbiIsInN1YiI6InNq

bGVlNDYyOCIsImV4cCI6MTYyNTU2MzY0OSwiaWF0IjoxNjI1NTYxODQ5LCJ0b2tlbl90eXBlIjoiYmVhcmVyIiwic2NvcGUiOm51bGx9.RRLnst2wWUBseqOeh0JQy2UJ1nPjz89E9ckc_SK7V6uHn5QX

TfHwH-xn55xlCv__cRjRaZw2rSVcp9WeoGRx7_1d-950vY2C6hpkKBmOp06GdslVt468YTQp2snM0wPzQBKz1dwM7BzSPLe3Aq2pvsw47fKHJly5c_gOw287YW9VnL4lOhhdPSc6Uc-

D0fqI2TfNg60aVb6w4hDDURH0rPyRIoFMQRlsN0FUloDdVKggaeD2n3DR7AM4mXrzwACIzrOw4uTTgJXA88OarZfdruk6grC8FdVAdRS7OXEr0henzUf8ljALz3CeMTJIo6_jO1SSkHwcINmpXnEn2_7U

NQ",

        "expires_in": 1800,

        "token_type": "bearer",

        "scope": null,

        "refresh_token": "6cc2402208233f454bb3408611d41e345b8b3281"
```