Software Security Assignment Report

Table of Contents

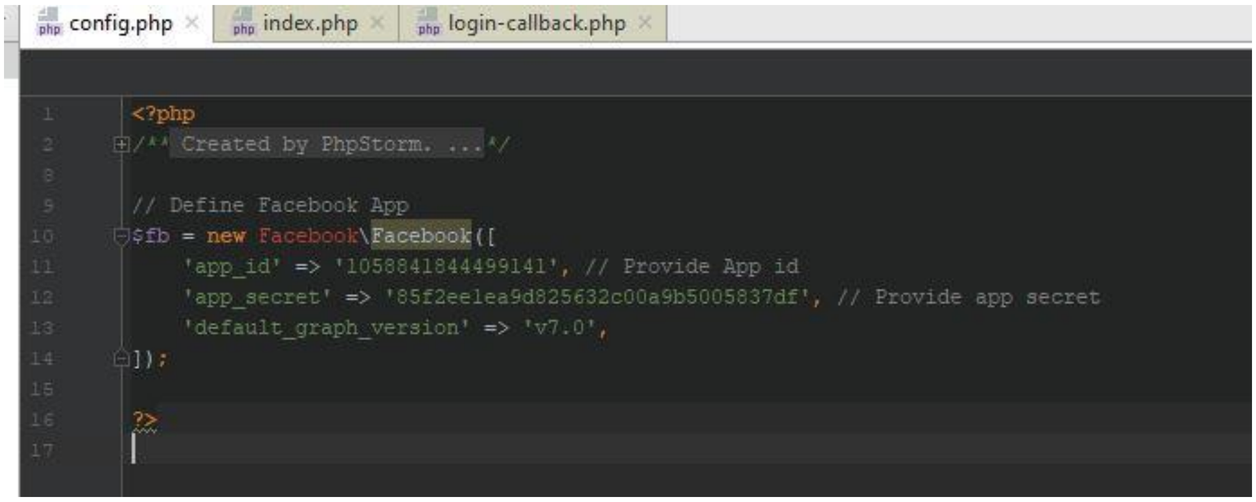
Table of Figures

1.0 Introduction

OAuth is simply put an authorization protocol or framework that allows safe access to their resources and assets without using initial logon credentials. This provides a way for unrelated services to provide authorized access without using login credentials.

2.0 Message Flow and Descriptions

2.1 Config.php



```php
<?php
/** Created by PhpStorm. ...*/

// Define Facebook App
$fb = new Facebook\Facebook([
    'app_id' => '1058841844499141', // Provide App id
    'app_secret' => '85f2ee1ea9d825632c00a9b5005837df', // Provide app secret
    'default_graph_version' => 'v7.0',
]);

?>
```

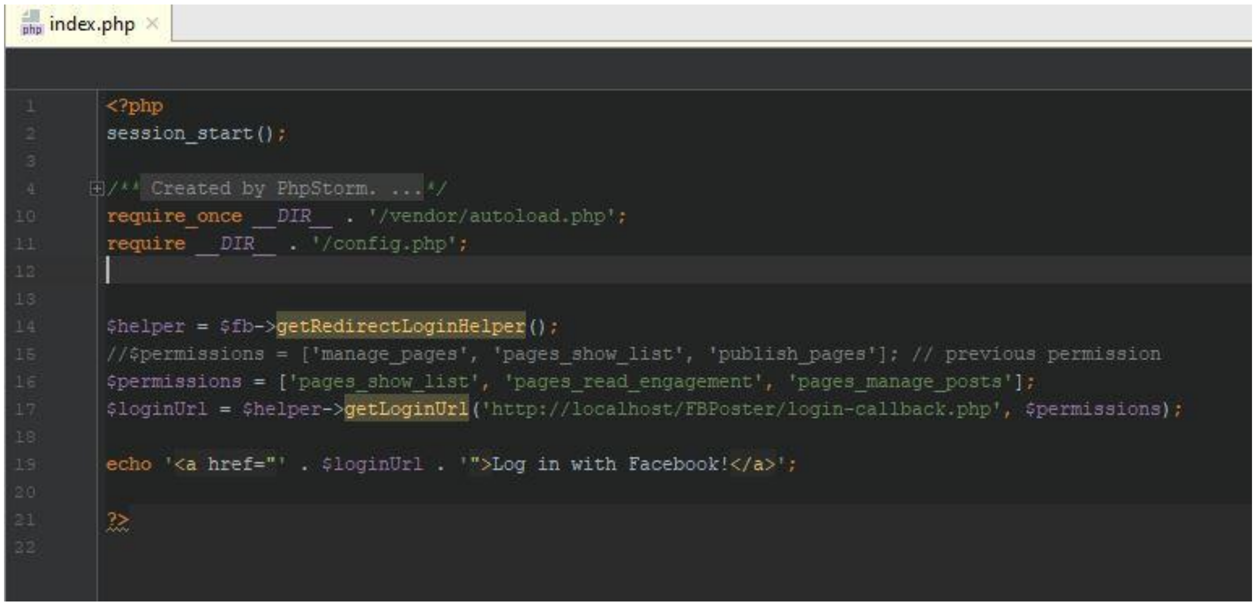
Figure 1- config.php

The config.php is used to define the Facebook application. The app_id and app_secret is used by the Facebook Graph API to uniquely identify each application. The latest graph version can be found here.

The object of type Facebook is created and assigned to the variable $fb using the above-mentioned app_id and app_secret. It is defined as the “The main service object that helps tie all the SDK components together” in the Core API. API Reference can be found here.

Note: The above provided app_secret and app_id has been reset after taking the screenshot.

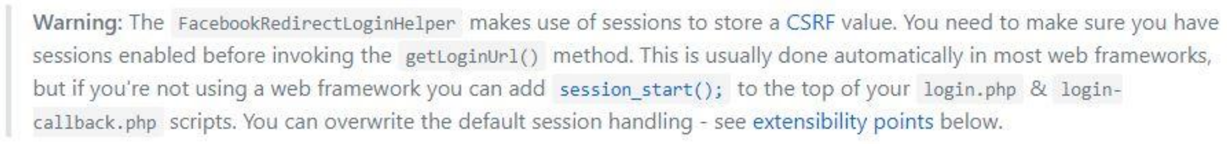
2.2 index.php



```
index.php

<?php
session_start();

/** Created by PhpStorm. ...*/
require_once __DIR__ . '/vendor/autoload.php';
require __DIR__ . '/config.php';


$helper = $fb->getRedirectLoginHelper();
//$permissions = ['manage_pages', 'pages_show_list', 'publish_pages']; // previous permission
$permissions = ['pages_show_list', 'pages_read_engagement', 'pages_manage_posts'];
$loginUrl = $helper->getLoginUrl('http://localhost/FBPoster/login-callback.php', $permissions);

echo '<a href="' . $loginUrl . '">Log in with Facebook!</a>';

?>
```

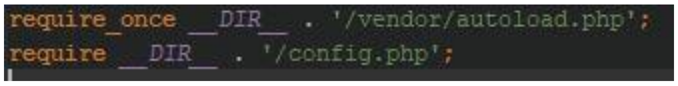
Figure 2 - index.php

A new session must be initiated since one of the classes RedirectLoginHelper uses sessions to store a CSRF (Cross-site request forgery) value as stated below.



Warning: The `FacebookRedirectLoginHelper` makes use of sessions to store a CSRF value. You need to make sure you have sessions enabled before invoking the `getLoginUrl()` method. This is usually done automatically in most web frameworks, but if you're not using a web framework you can add `session_start();` to the top of your `login.php` & `login-callback.php` scripts. You can overwrite the default session handling - see extensibility points below.

RedirectLoginHelper API reference can be found here.



```
require_once __DIR__ . '/vendor/autoload.php';
require __DIR__ . '/config.php';
```

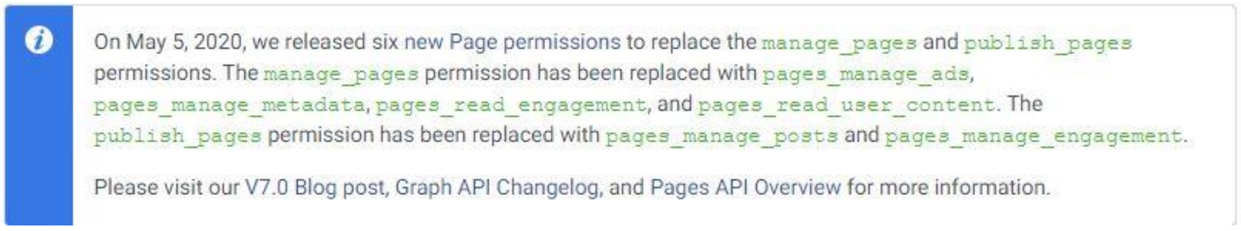
Figure 3 - Imports

The Facebook Graph SDK is imported using the ‘autoload.php’ while the app config in ‘config.php’ is imported afterwards.

The imported Facebook object is then used to obtain an object of type RedirectLoginHelper which will be used to obtain OAuth permissions for the app. The functions definition can be found here.

The permissions that user need to authorize using OAuth is defined in the $permissions variable. Previous permissions prior to the V7.0 has also been provided in comments in the source. List of permissions can be found here.

Note: The permission list has not been updated for Graph SDK V7.0 as of 08/05/2020) Refer https://developers.facebook.com/docs/pages/overview-1#permissions.



On May 5, 2020, we released six new Page permissions to replace the `manage_pages` and `publish_pages` permissions. The `manage_pages` permission has been replaced with `pages_manage_ads`, `pages_manage_metadata`, `pages_read_engagement`, and `pages_read_user_content`. The `publish_pages` permission has been replaced with `pages_manage_posts` and `pages_manage_engagement`.

Please visit our V7.0 Blog post, Graph API Changelog, and Pages API Overview for more information.

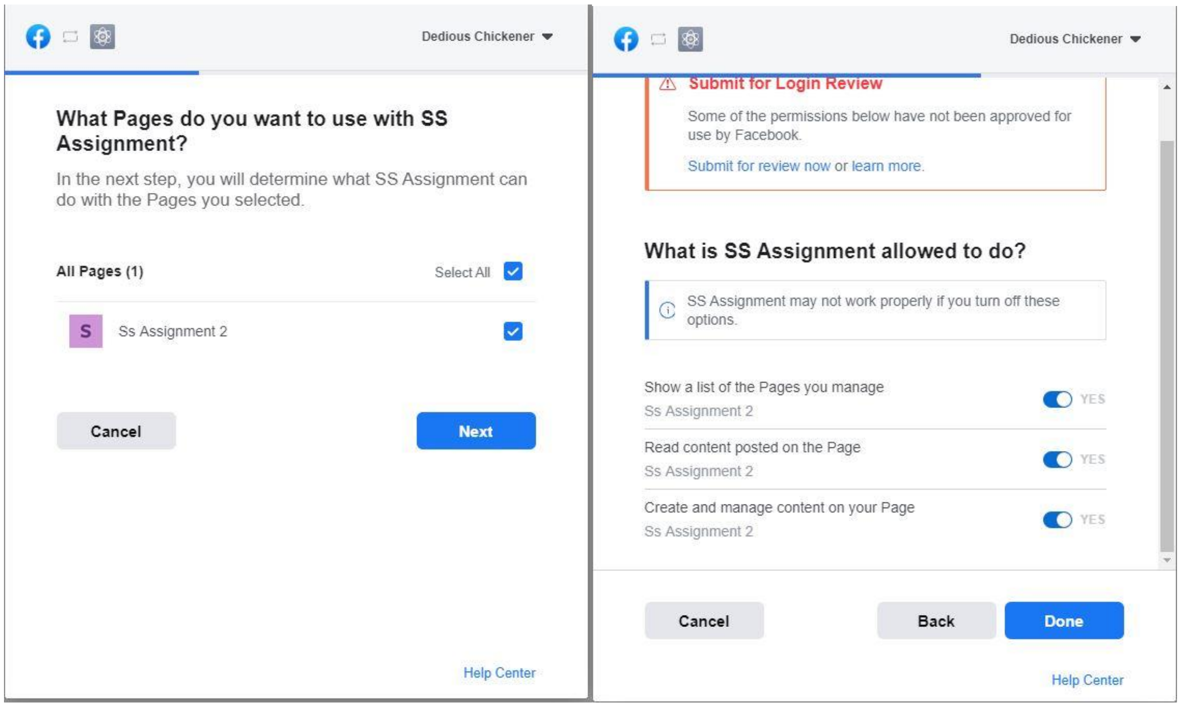
Figure 4 - Permission update

The selected permissions can be explained as follows:

- pages_show_list – Grants permission to view the pages handled by the user.
- pages_read_engagement – Get metadata about the page (i.e. Page ID)
- pages_manage_posts – Ability to publish posts, videos, photos to a page.

Afterwards a redirect URL is generated using the helper class. The callback URL provided handles the application logic after OAuth permission has been granted / denied by the user.

Note: If the app is hosted on a public domain replace 'localhost' by the relevant IP address or the domain name of your server.



Dedious Chickener

What Pages do you want to use with SS Assignment?

In the next step, you will determine what SS Assignment can do with the Pages you selected.

All Pages (1) — Select All ✓

Ss Assignment 2 ✓

Cancel Next

Help Center

Submit for Login Review

Some of the permissions below have not been approved for use by Facebook.

Submit for review now or learn more.

What is SS Assignment allowed to do?

SS Assignment may not work properly if you turn off these options.

Show a list of the Pages you manage — Ss Assignment 2 — YES

Read content posted on the Page — Ss Assignment 2 — YES

Create and manage content on your Page — Ss Assignment 2 — YES

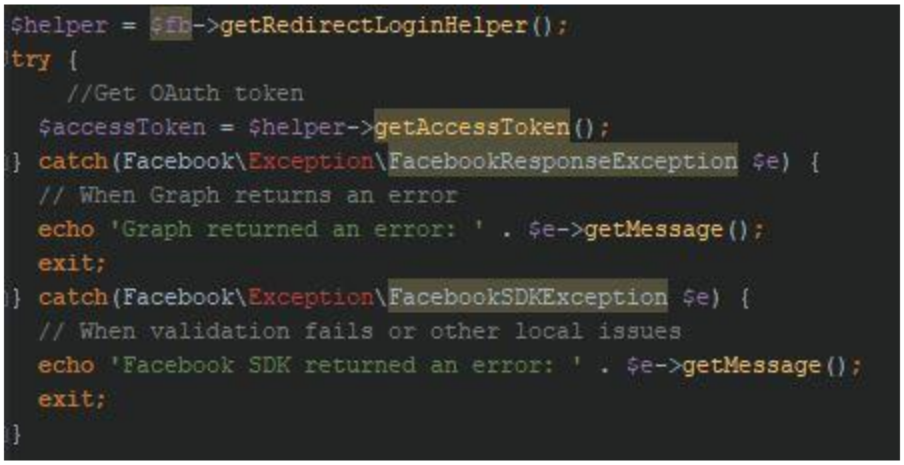
Cancel Back Done

Help Center

Figure 5 - OAuth

Above windows prompt the user to accept/deny the OAuth request by the application. Then the control flow is transferred back to the application through the redirect URL provided in 'index.php'. In this case, to the 'login-callback.php' script.

2.3 login-callback.php

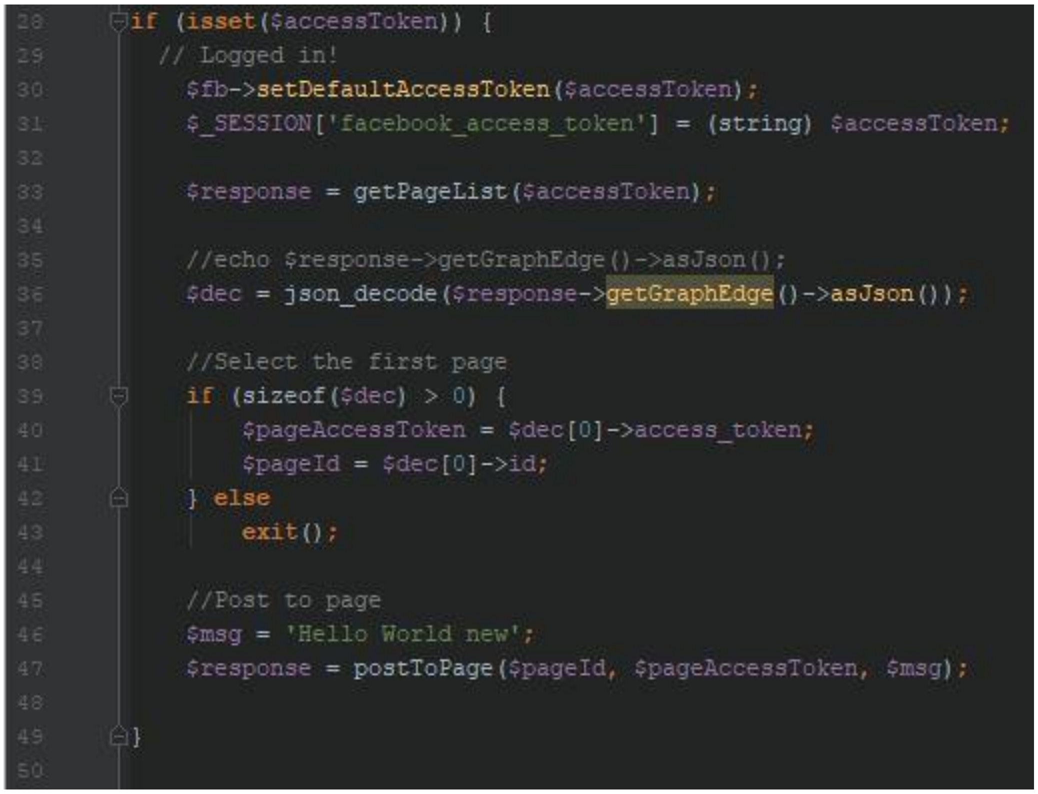


```
$helper = $fb->getRedirectLoginHelper();
try {
    //Get OAuth token
  $accessToken = $helper->getAccessToken();
} catch(Facebook\Exception\FacebookResponseException $e) {
  // When Graph returns an error
  echo 'Graph returned an error: ' . $e->getMessage();
  exit;
} catch(Facebook\Exception\FacebookSDKException $e) {
  // When validation fails or other local issues
  echo 'Facebook SDK returned an error: ' . $e->getMessage();
  exit;
}
```

Figure 6 - Access Token

The above code snippet tries to obtain an access token for the app to operate on. The getAccessToken function is provided by the SDK in the RedirectLoginHelper class. Function definition for this can be found here. The exceptions thrown will be used to recognize graph errors and CSRF validation fails (FacebookSDKException).

When the script is able to obtain the access token with the requested permissions, the application logic to invoke the Graph API to post a message onto a page will be executed. The Following snippet displays that logic.



```php
if (isset($accessToken)) {
  // Logged in!
    $fb->setDefaultAccessToken($accessToken);
    $_SESSION['facebook_access_token'] = (string) $accessToken;

    $response = getPageList($accessToken);

    //echo $response->getGraphEdge()->asJson();
    $dec = json_decode($response->getGraphEdge()->asJson());

    //Select the first page
    if (sizeof($dec) > 0) {
        $pageAccessToken = $dec[0]->access_token;
        $pageId = $dec[0]->id;
    } else
        exit();

    //Post to page
    $msg = 'Hello World new';
    $response = postToPage($pageId, $pageAccessToken, $msg);

}
```

Figure 7 – Application logic

First the isset() function is used to determine if the accessToken variable is not null since according to the getAccessToken() reference,

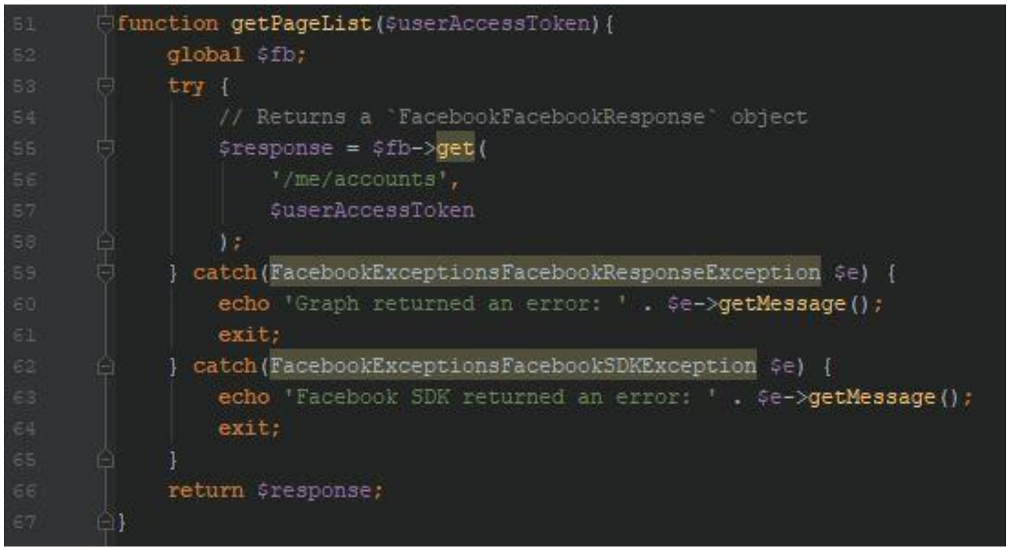
"*If no authorization code could be found from the code param in the URL, this method will return null.*"

After that it is possible to use the access token in two ways

1. Using setDefaultAccessToken function in Facebook class to set a fallback access token or
2. Setting a session variable with the access token.

Both methods are initiated here and either can be used to invoke resources from the Graph API.

Two resources from the API have been invoked in this script as to obtain the page list managed by the user and to post a message to the first page. Two separate functions have been created for them.

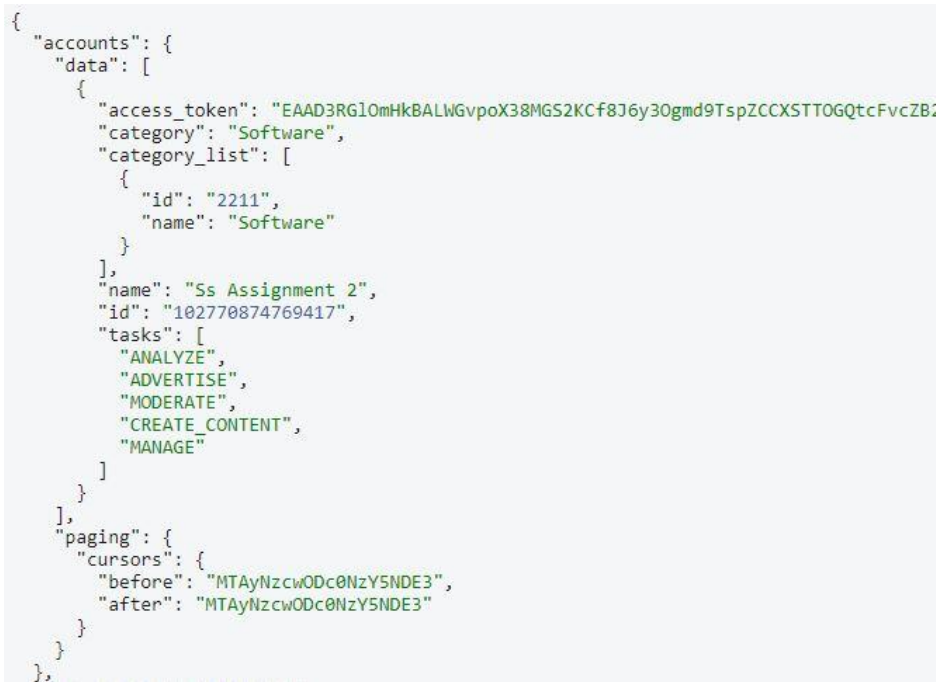


```php
function getPageList($userAccessToken){
    global $fb;
    try {
        // Returns a `FacebookFacebookResponse` object
        $response = $fb->get(
            '/me/accounts',
            $userAccessToken
        );
    } catch(FacebookExceptionsFacebookResponseException $e) {
        echo 'Graph returned an error: ' . $e->getMessage();
        exit;
    } catch(FacebookExceptionsFacebookSDKException $e) {
        echo 'Facebook SDK returned an error: ' . $e->getMessage();
        exit;
    }
    return $response;
}
```

Figure 8 - getPageList Function

The function [get()]() in the Facebook class can be used to obtain information from the Graph API. A GET request to *https://graph.facebook.com/v7.0/me?fields=accounts&access_token=''* with the user access token must be made to obtain the page list. It is done by the get() function as shown above.

Executing the above query against the API would provide a JSON encoded string as shown below.



```
{
  "accounts": {
    "data": [
      {
        "access_token": "EAAD3RGlOmHkBALWGvpoX38MGS2KCf8J6y3Ogmd9TspZCCXSTTOGQtcFvcZB
        "category": "Software",
        "category_list": [
          {
            "id": "2211",
            "name": "Software"
          }
        ],
        "name": "Ss Assignment 2",
        "id": "102770874769417",
        "tasks": [
          "ANALYZE",
          "ADVERTISE",
          "MODERATE",
          "CREATE_CONTENT",
          "MANAGE"
        ]
      }
    ],
    "paging": {
      "cursors": {
        "before": "MTAyNzcwODc0NzY5NDE3",
        "after": "MTAyNzcwODc0NzY5NDE3"
      }
    }
  },
```

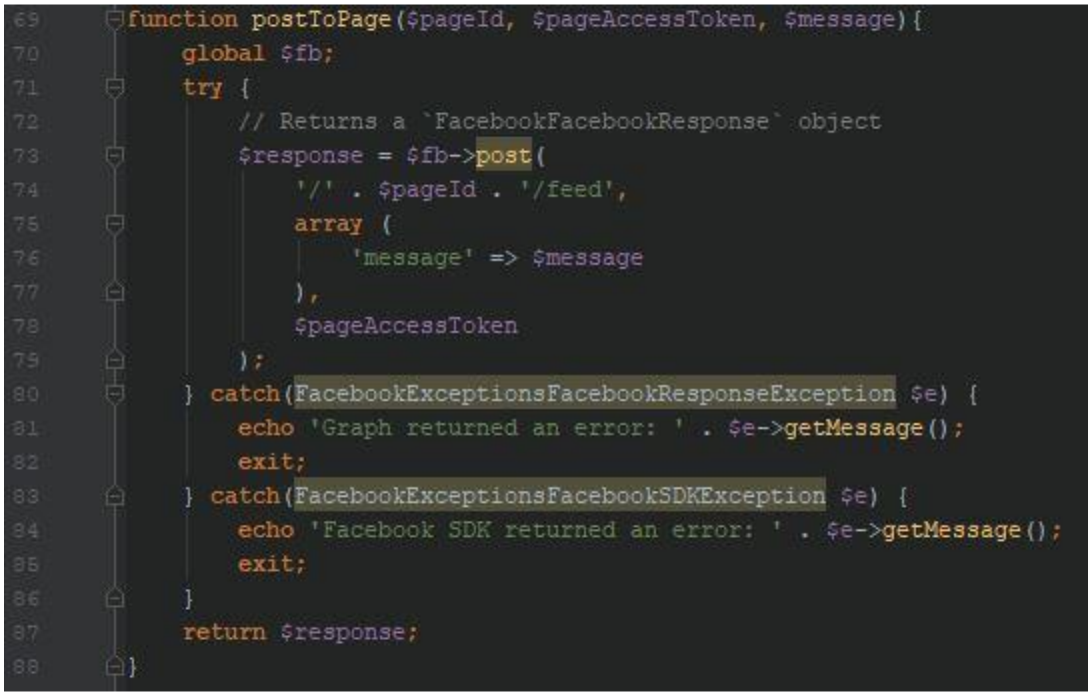
Figure 9 - Page List Result

To publish a message, video or photo to a page it requires two fields from the above output namely ‘id (The id field under data and not category_list)’ and ‘access_token’.

The graph API uses several types of Access tokens to provide access to different resources. These are generated according to the initial permissions requested by the application. To publish to a specific page the API requires an access token specific to that page (page access token).

The ‘id’ field represents the page id and the access_token field represents the page access token.

The returned json encoded string is then decoded using json_decode and assigned to the variable dec which in turn is used to obtain the page access token and the page id to post a message.



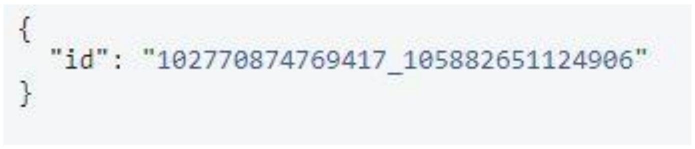
```
function postToPage($pageId, $pageAccessToken, $message){
    global $fb;
    try {
        // Returns a `FacebookFacebookResponse` object
        $response = $fb->post(
            '/' . $pageId . '/feed',
            array (
                'message' => $message
            ),
            $pageAccessToken
        );
    } catch(FacebookExceptionsFacebookResponseException $e) {
        echo 'Graph returned an error: ' . $e->getMessage();
        exit;
    } catch(FacebookExceptionsFacebookSDKException $e) {
        echo 'Facebook SDK returned an error: ' . $e->getMessage();
        exit;
    }
    return $response;
}
```

Figure 10 - postToPage function

Above obtained page id and page access token is used in the postToPage function to post a message to that particular page. The post() function in the Facebook class can be used for this.

A POST request to the following URL must be made to publish to a page.
https://graph.facebook.com/v7.0/{page_id}/feed?message=hello&access_token={pageAccessToken}

It is done using the post() function as shown above. If successful the response would contain a JSON encoded string with the message ID of the newly posted message as shown below.



```
{
  "id": "102770874769417_105882651124906"
}
```

Figure 11 - Message ID

All of the above functionality uses OAuth to access a user's profile and the pages they manage using the Graph API.

Appendix A – Source Code

config.php

```
<?php
/**
 * Created by PhpStorm.
 * User: user
 * Date: 5/3/2020
 * Time: 9:01 PM
 */
// Define Facebook App
$fb = new Facebook\Facebook([
    'app_id' => '1058841844499141', // Provide App id
    'app_secret' => '85f2ee1ea9d825632c00a9b5005837df', // Provide app secret
    'default_graph_version' => 'v7.0',
]);

?>
```

index.php

```
<?php
session_start();
/**
 * Created by PhpStorm.
 * User: user
 * Date: 4/29/2020
 * Time: 5:25 AM
 */
require_once __DIR__ . '/vendor/autoload.php';
require __DIR__ . '/config.php';


$helper = $fb->getRedirectLoginHelper();
//$permissions = ['manage_pages', 'pages_show_list', 'publish_pages']; // previous permission
$permissions = ['pages_show_list', 'pages_read_engagement', 'pages_manage_posts'];
$loginUrl = $helper->getLoginUrl('http://localhost/FBPoster/login-callback.php', $permissions);

echo '<a href="' . $loginUrl . '">Log in with Facebook!</a>';

?>
```

login-callback.php

```
<?php
session_start();
/**
 * Created by PhpStorm.
 * User: user
 * Date: 4/29/2020
 * Time: 5:25 AM
 */
require_once __DIR__ . '/vendor/autoload.php';
require __DIR__ . '/config.php';



$helper = $fb->getRedirectLoginHelper();
try {
    //Get OAuth token
  $accessToken = $helper->getAccessToken();
} catch(Facebook\Exception\FacebookResponseException $e) {
  // When Graph returns an error
  echo 'Graph returned an error: ' . $e->getMessage();
  exit;
} catch(Facebook\Exception\FacebookSDKException $e) {
  // When validation fails or other local issues
  echo 'Facebook SDK returned an error: ' . $e->getMessage();
  exit;
}

if (isset($accessToken)) {
  // Logged in!
    $fb->setDefaultAccessToken($accessToken);
    $_SESSION['facebook_access_token'] = (string) $accessToken;

    $response = getPageList($accessToken);

    //echo $response->getGraphEdge()->asJson();
    $dec = json_decode($response->getGraphEdge()->asJson());

    //Select the first page
    if (sizeof($dec) > 0) {
        $pageAccessToken = $dec[0]->access_token;
        $pageId = $dec[0]->id;
    } else
        exit();

    //Post to page
    $msg = 'Hello World new';
    $response = postToPage($pageId, $pageAccessToken, $msg);

}

function getPageList($userAccessToken){
    global $fb;
    try {
        // Returns a `FacebookFacebookResponse` object
        $response = $fb->get(
            '/me/accounts',
            $userAccessToken
        );
    } catch(FacebookExceptionsFacebookResponseException $e) {
        echo 'Graph returned an error: ' . $e->getMessage();
        exit;
    } catch(FacebookExceptionsFacebookSDKException $e) {
        echo 'Facebook SDK returned an error: ' . $e->getMessage();
        exit;
    }
    return $response;
}

function postToPage($pageId, $pageAccessToken, $message){
    global $fb;
    try {
        // Returns a `FacebookFacebookResponse` object
        $response = $fb->post(
            '/' . $pageId . '/feed',
            array (
                'message' => $message
            ),
            $pageAccessToken
        );
    } catch(FacebookExceptionsFacebookResponseException $e) {
        echo 'Graph returned an error: ' . $e->getMessage();
        exit;
    } catch(FacebookExceptionsFacebookSDKException $e) {
        echo 'Facebook SDK returned an error: ' . $e->getMessage();
        exit;
    }
    return $response;
}

?>
```