

Name: Adeniran Adeniyi

Class: CS518

Graduate Project 2

1. Introduction

For this project, I worked on using a Github account or a Google account or a twitter account to sign into the website in order to skip the registration process of using the chat app. Once signed in, the user can enjoy all perks of using the chat application.

2. Login Process

I first did GitHub then moved on to the google which I then finalized with twitter. I noted that all three had similar processes.

2.1 Creation

First Process was to have an account on the desired platform of the app. For Github, once signed in, it was going to settings then to developer settings and click on New OAuth App. For Twitter it is as simple as going to the <https://apps.twitter.com>, then click on create an app, while as for Google it was as simple as going to the <https://console.developers.google.com>, go to the library and search for Google+ API and enable the service. Once enabled, go back to the panel screen and click credentials, proceed by clicking Create Credentials proceed by choosing OAuth client ID and select Web application.

During the creation processing for each three, you will be required to fill some important details about your application. The application name is required, the website homepage URL to be used, and callback URL will be the major things to be noted. Other information required for the application creation may include application description, terms of service URL and privacy policy URL. Once you successfully create the application for each platform, there would be a Client ID and Client Secret automatically generated specifically for your application. For Twitter, it is called API KEY

and API Secret KEY Respectively. Those two keys are what would be used for the development phase.

2.2 Development

Development phase takes those two keys which will be used to send a user to the platform area requesting the user's permission to allow the application we create to access the user details. To create a user, I needed four requirements. The Full name of the user, the email address of the user, user handle and user profile picture link. The application would let the user know what informational details my application would be able to access if permission is granted to the application. Once permission is granted, an access token is then sent back to me to the callback URL I kept in place. I process this token and extract the major details I needed from the user account.

2.2 Registration and Sign In

Once this information is gotten I register the user to my local database and log the user into the system.

Below I specified ways I went about the code development for all three processes.

For GitHub, I had a function that processed, taken the user to sign with their GitHub account.

```
“function gitLogin()
{
  ///was using the two generated tokens from github
  define('OAUTH2_CLIENT_ID', '541b6e510ce03d138842');
  define('OAUTH2_CLIENT_SECRET',
'728df75aa33cd44226a78ee687be6df49f69a722');
  $authorizeURL = 'https://github.com/login/oauth/authorize';
  $tokenURL    = 'https://github.com/login/oauth/access_token';
  $apiURLBase  = 'https://api.github.com/';
  // Start the login process by sending the user to Github's authorization
page
```

```

if(get('action') == 'login')
{
    // Generate a random hash and store in the session for security
    $_SESSION['state'] = hash('sha256', microtime(TRUE) . rand() .
$_SERVER['REMOTE_ADDR']);
    unset($_SESSION['access_token']);
    $params = array(
        'client_id' => OAUTH2_CLIENT_ID,
        'redirect_uri' => 'http://' . $_SERVER['SERVER_NAME'] .
$_SERVER['PHP_SELF'],
        'scope' => 'user:email',
        'state' => $_SESSION['state']
    );
    // Redirect the user to Github's authorization page
    header('Location: ' . $authorizeURL . '?' . http_build_query($params));
    die();
}

```

A condition that checks the callback. The callback after a successful login returns a code and state for github. The callback basically gets the access token and extracts the users details. Once the details are gotten the condition the sends the details to the signup area. Once signup is complete the user is set to the login area and now the user in the welcome page.

3. Problems and Solution

3.1 Google Login solution:

As for Google, it was the easiest out of the three because Google made the integration of signing in with a google account without chat application simple. From the link <https://developers.google.com/identity/sign-in/web/sign-in>, you can see a pre-made code for developers on how to make this integration possible. All I had to do was get the information I receive to sign up and then to the login page in other to get the user logged in.

3.2 Twitter Login problems, solutions, and resources:

Out of all the three twitter gave me a lot of problems.

I followed a good tutorial on youtube that leads me in getting the information but the user email address was missing. Youtube link https://www.youtube.com/watch?v=Jp-Yb-lwYkY&list=PLfdtiltiRHWHRMnQjPn9G2c1_mbdST9Ao. I had to search about the tool, codebird, being used in other to get all the required information of the user. I went to codebird GitHub page <https://github.com/jublo/codebird-php>, where I made an issue to the developers about the problem <https://github.com/jublo/codebird-php/issues/245>, I also posted the same problem on stack overflow <https://stackoverflow.com/questions/53797163/how-to-retrieve-twitter-user-email-using-codebird-php/53807163#53807163>.

I found a solution to the problem on StackOverflow, but with a few research I solved the problem and was able to get all the necessary information that I needed - you can view my responses on the StackOverflow link. After getting all the necessary information I sent the information to the signup page and logged the user in using my local machine.

3.3 Docker problems, solution, and resources:

When I was testing the Twitter functionality on docker I encountered "a file not found" problem. I was initially using the composer file to load the code properties of codebird.php like this "require_once __DIR__ .'/vendor/autoload.php';" but when I used it directly like this "require_once 'codebird.php';" was able to use codebirds code.

Another error arose which stated "error setting certificate verify locations: CAfile:" after searching online I found a solution in this link <https://support.routexl.com/t/error-setting-certificate-verify-locations-cafile-cacert-pem-capath-none/254>, I downloaded the cacert.pem file in the "app/" directory. When I tested the twitter functionality again it worked.

I also created a logout button which is located at the sidebar of the Welcome.php. The logout button manages the logging out from the application once signed in with any the mentioned platforms above.

4. DESIGN

For a different email address used we would have a new user. For instance, when a user signs up with a Gmail account(someOne@gmail.com) it will be logged as just one account on any platform you will log in from. Once logged in, the profile picture will assume the profile picture from the platform you used to log in with. If the user handle already exists, the application will reject the sign in. The user would have to edit their user their user handle on that platform to sign in.

5. Conclusion

In conclusion, making the user have the opportunity in using any 3 of the platforms - Google, Github, Twitter - to sign up gives great simplicity in using the application. It has also help exposed my knowledge and offered hands-on exercises in using such API's.