Lab 7 - Security and OAuth

OAuth Facebook

OAuth เป็นมาตรฐานที่ใช้ในการพิสูจน์ตัวตน ที่ปัจจุบันมีการนิยมใช้ในหมู่นักพัฒนาเพิ่มขึ้นเรื่อย ๆ เนื่องจากระบบ Login ที่ใช้ OAuth จะทำให้แอพพลิเคชันมีความปลอดภัยในการเข้าใช้งานของผู้ใช้ เนื่องจาก ในนั้น OAuth หลังจากที่ Login กับ Authorization server เรียบร้อยแล้ว OAuth จะใช้ Token ในการแลกข้อมูลกับเซิร์ฟเวอร์ ทำให้ User Name กับ Password ของผู้ใช้ จะไม่ถูกส่งไปยัง ผู้ให้บริการอื่น ๆ นอกจาก Token เท่านั้น นอกจากนี้ ยังสามารถใช้งานกับเว็บและโมบายแอพพลิเคชันได้อีกด้วย

ตัวอย่าง setting ของ facebook application

Pre-Requirements:

- **Facebook Application**
  https://developers.facebook.com/apps/
- **Graph explorer**
  https://developers.facebook.com/tools/explorer
- **Facebook API Reference**
  https://developers.facebook.com/docs/facebook-login/web

Client login Example

public/index.html

```
<!DOCTYPE html>
<html>
<head>
   <title>Facebook Login JavaScript Example</title>
   <meta charset="UTF-8">
```

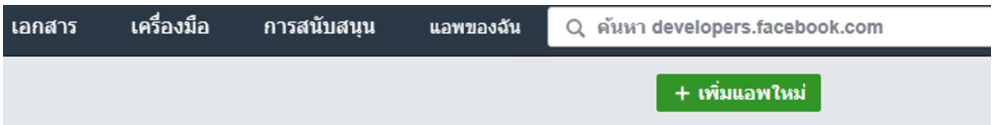
```
</head>
<body>
<script>
   // This is called with the results from from FB.getLoginStatus().
   function statusChangeCallback(response) {
       console.log('statusChangeCallback');
       console.log(response);
       // The response object is returned with a status field that lets the
       // app know the current login status of the person.
       // Full docs on the response object can be found in the documentation
       // for FB.getLoginStatus().
       if (response.status === 'connected') {
           // Logged into your app and Facebook.
           testAPI();
       } else {
           // The person is not logged into your app or we are unable to tell.
           document.getElementById('status').innerHTML = 'Please log ' +
               'into this app.';
       }
   }

   // This function is called when someone finishes with the Login
   // Button.  See the onlogin handler attached to it in the sample
   // code below.
   function checkLoginState() {
       FB.getLoginStatus(function(response) {
           statusChangeCallback(response);
       });
   }

   window.fbAsyncInit = function() {
       FB.init({
           appId      : '137842830268395',
           cookie     : true,  // enable cookies to allow the server to access
                               // the session
           xfbml      : true,  // parse social plugins on this page
           version    : 'v3.1' // The Graph API version to use for the call
       });

       // Now that we've initialized the JavaScript SDK, we call
       // FB.getLoginStatus().  This function gets the state of the
       // person visiting this page and can return one of three states to
       // the callback you provide.  They can be:
       //
       // 1. Logged into your app ('connected')
       // 2. Logged into Facebook, but not your app ('not_authorized')
       // 3. Not logged into Facebook and can't tell if they are logged into
       //    your app or not.
       //
       // These three cases are handled in the callback function.

       FB.getLoginStatus(function(response) {
           statusChangeCallback(response);
       });

   };

   // Load the SDK asynchronously
   (function(d, s, id) {
       var js, fjs = d.getElementsByTagName(s)[0];
```

```
        if (d.getElementById(id)) return;
        js = d.createElement(s); js.id = id;
        js.src = "https://connect.facebook.net/en_US/sdk.js";
        fjs.parentNode.insertBefore(js, fjs);
    }(document, 'script', 'facebook-jssdk'));

    // Here we run a very simple test of the Graph API after login is
    // successful.  See statusChangeCallback() for when this call is made.
    function testAPI() {
        console.log('Welcome!  Fetching your information.... ');
        FB.api('/me', function(response) {
            console.log('Successful login for: ' + response.name);
            document.getElementById('status').innerHTML =
                'Thanks for logging in, ' + response.name + '!';
        });
    }
</script>

<!--
 Below we include the Login Button social plugin. This button uses
 the JavaScript SDK to present a graphical Login button that triggers
 the FB.login() function when clicked.
-->

<fb:login-button scope="public_profile,email" onlogin="checkLoginState();">
</fb:login-button>

<div id="status">
</div>

</body>
</html>
```

ในโปรแกรม login.html ให้ แก้ไขค่า appId

```
        appId      : '137842830268395',
```

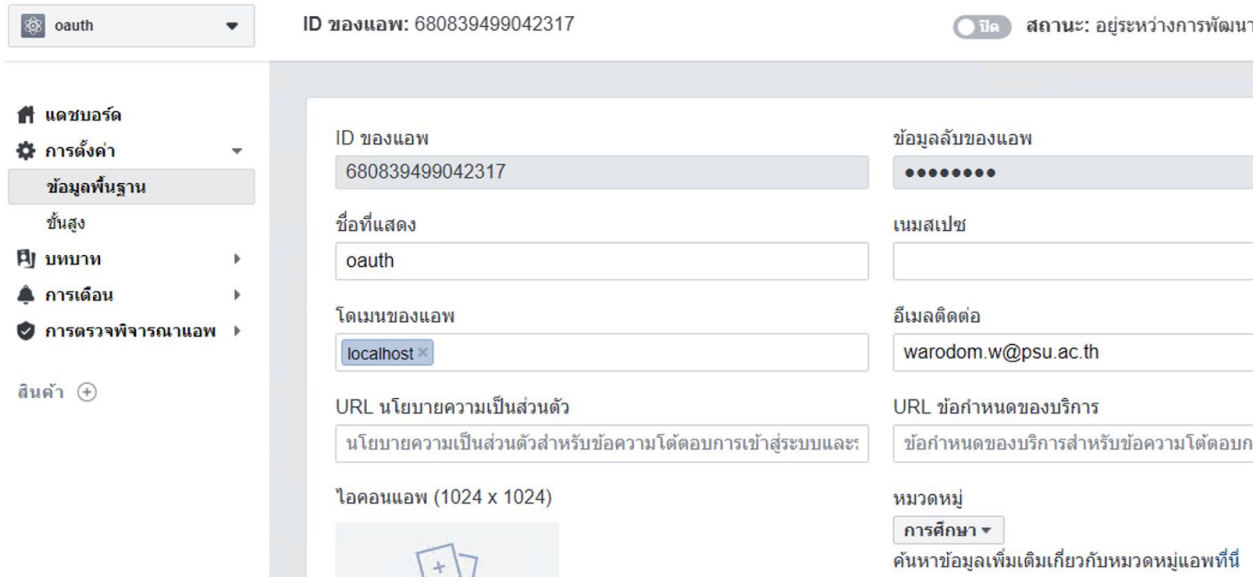
เป็นค่า ID ของตัวเอง โดยเข้าไปสร้าง facebook application ได้จาก https://developers.facebook.com/apps/ เลือกที่ +เพิ่มแอพใหม่



จากนั้นป้อนข้อมูลลงในแบบฟอร์มให้เรียบร้อย ชื่อที่แสดงจะใช้เป็นชื่ออะไรก็ได้ จากนั้นกดปุ่ม สร้าง ID ของแอพ (อาจจะมีหน้าให้กรอกรหัสความปลอดภัย)

เมื่อสร้างเสร็จเรียบร้อย จะเปลี่ยนเข้ามาที่หน้า facebook app ที่สร้างดังนี้

เลือกที่การตั้งค่า ⇒ ข้อมูลพื้นฐาน กำหนด โดเมนของแอพเป็น localhost และ หมวดหมู่ กำหนดเป็นการศึกษา จากนั้น กำหนดบันทึกการเปลี่ยนแปลง และนำค่า ID ของแอพ ไปใส่ใน login.html



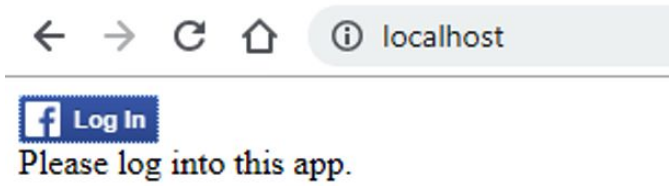
จากนั้นเรียกใช้ facebook application โดย การเรียกแม้ว่าจะเป็นไฟล์ javascript ที่สามารถ run ที่ client side ผ่าน browser แต่การใช้ จะต้องเรียกผ่าน web server เนื่องจาก facebook มีการตรวจสอบโดเมนของแอพ ที่ได้กำหนดเป็น localhost ไว้ก่อนหน้านี้

ดังนั้นจำเป็นต้องสร้าง npm project และ สร้างไฟล์ index.js ดังนี้

index.js

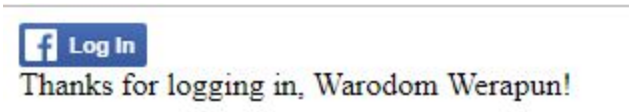
```
var express = require('express');
var app = express();
app.use(express.static(__dirname + '/public'));
app.listen(80);
```

และสั่งให้ทำงาน โดยใช้ node index.js และเปิดหน้า browser ไปที่ http://localhost จะ แสดงข้อมูลให้ Login ผ่าน facebook โดยอ่านจาก ./public/index.html ดังรูป



เมื่อกดปุ่ม Login แล้ว จะเห็นหน้าต่างที่ Facebook ถามสิทธิ์ในการเข้าถึงเจ้าของข้อมูล (Resource owner) ให้ ตอบดำเนินการต่อไป

จากนั้นจะเห็น สถานะว่า Login เรียบร้อยแล้ว พร้อมกับ แสดงชื่อของผู้ที่ Login (บัญชี Facebook)



ทดลองเรียกใช้ Facebook API เพิ่มเติม

ในส่วนของฟังก์ชันต้องเพิ่มใน tag <script> ส่วน html เพิ่มใน tag <body>

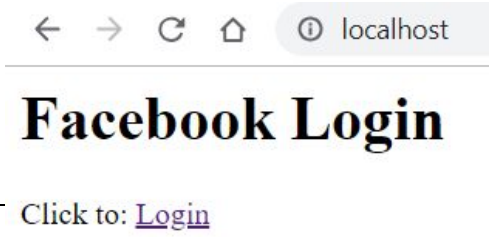
```
function getMe() {
   FB.api('/me', function (response) {
       console.log(JSON.stringify(response));
       document.getElementById('status').innerHTML = JSON.stringify(response);
   });
}

function logout() {
   FB.logout(function(response) {
      console.log('Logout')
       document.getElementById('status').innerHTML =
           'Thanks for logging out!';
   });
}


...

<button onClick="getMe()" >Get Me </button>
<button onClick="logout()" >Logout</button>
```

ในระหว่างที่ ทดสอบโปรแกรม จะเห็น warning เตือนเป็นข้อความที่ console ของ browser ดังนี้

> The Login Button plugin will soon stop working on http pages. Please update your site to use https for Facebook Login. https://developers.facebook.com/blog/post/2018/06/08/enforce-https-facebook-login/

หมายความว่า การเรียกใช้ จะต้องเรียกผ่าน https:// ไม่ใช่ http:// แต่การกำหนด https จำเป็นจะต้องมี Domain Name ของตัวเองที่เข้าถึงได้จริง จึงไม่สามารถทดลองใช้ https:// ได้

Source: Facebook.html



Facebook Server login

Using code from:
https://github.com/wwarodom/facebook_api

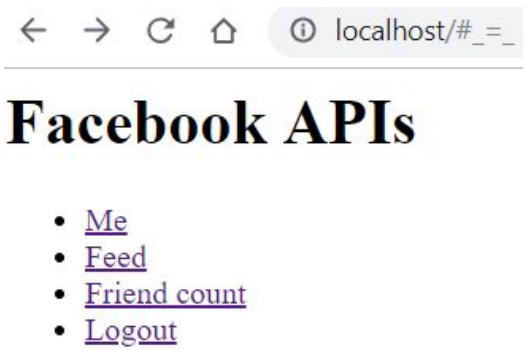
$ git clone https://github.com/wwarodom/facebook_api.git

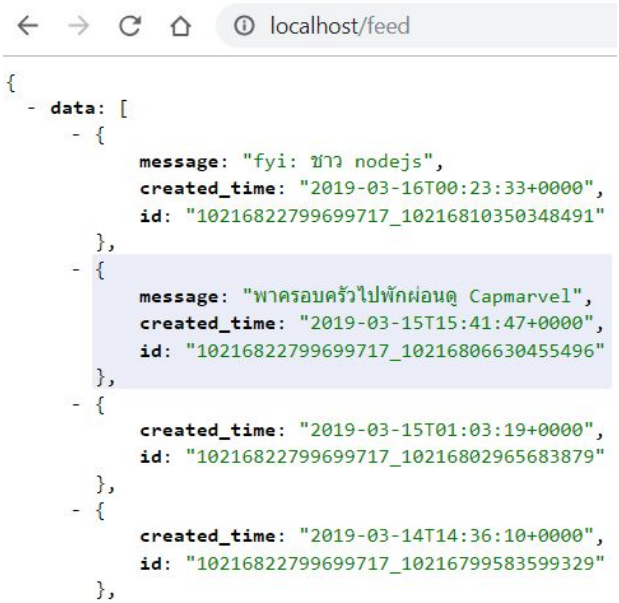
จากนั้นแก้ไขไฟล์ .env-example และเปลี่ยนชื่อไฟล์เป็น .env

- จัดการเปลี่ยนค่า APP_ID และ APP_SECRET ตามที่กำหนดใน Facebook Application (OAuth)

$ npm i เมื่อติดตั้งเรียบร้อย สั่ง $ node server

ผลการทำงาน





PSU Passport Web service (Login)

สามารถใช้ soap library เพื่อขอติดต่อ web service ของ psu passport เพื่อขอข้อมูลของผู้ใช้ได้
สร้าง npm project => npm init --y จากนั้น npm i --save soap body-parser และ node psusoap.js

psusoap.js

```
const express = require('express');
const soap = require('soap');
const bodyParser = require('body-parser')
const url = 'https://passport.psu.ac.th/authentication/authentication.asmx?wsdl';
const app = express()
const router = express.Router()
app.use(bodyParser.urlencoded({extended: false}), router)
app.use(bodyParser.json, router)

const out = `
<html>
<body>
  <h2>PSU Passport Authentication (SOAP) </h2>
 <form action="/" method="post">
 Username: <input type="text" name="username" /> <br>
 Password: <input type="password" name="password" /> <br>
 <input type="submit" value="Submit">
</form>
</body>
</html>
`

router.route('/')
  .get((req, res) => {
     res.send(out)
  })
  .post((req, res) => {
     soap.createClient(url, (err, client) => {
        if (err) console.error(err);
        else {
           let user = {}
           user.username = req.body.username
           user.password = req.body.password

           client.GetStaffDetails(user, function (err, response) {
              // client.GetStudentDetails(args, function(err, response) {
              if (err) console.error(err);
              else {
                 console.log(response);
                 res.send(response);
              }
           });
        }
     });
  })

app.listen(80, () => console.log('Server is ready!'))
```

Source: soap_psu.js