## HW05: Web APIs

SEIS 752: Advanced Web Application Engineering – Spring 2014

Instructor: Lloyd Cledwyn

Student: Dave Olson

**Summary**

Gain experience implementing a publicly published Web API.

**Purpose**

To research available web services, discover the functionalities, and implement a solution of a combination of them.

**Results - Twilio**

This assignment requires the implementation of two web APIs. The first web API is Twilio and the second web API is a choice between Oauth 2.0 or another public web service API.

The implementation of Twilio’s API focused on its SMS capabilities. This implementation was fairly seamless due to better-than-average documentation. I was able to download and install the Twilio PHP library, set up a phone number, and create the required files using PHP that enabled my web app to send and receive SMS messages from any phone to display on my web site. The example is located at <http://bromad.com/hw05-twilio.php>.

The PHP files enabled my web app to receive and process the SMS messages enabled by the API and dump the default Twilio parameters and their values into my MySQL table with a simple insert statement:

INSERT INTO twilliotest (dump) VALUES ('$ans')

By using a query to parse and return only the information that I wanted to view I was able to see the relevant information, including a timestamp field:

|  |  |
| --- | --- |
| Time: | 2014-05-04 12:36:16 |
| From: | 6128125823 |
| To: | 6122551559 |
| Body: | Testing... 123. |

The Twilio API provides powerful capability for both text-based and call-based communications. My project team has integrated the SMS capability into our web site with the intent of providing text-based journaling functionality so that parts of the site are updateable through any device with a messaging application. That implementation can be viewed at <http://www.scrapattack.net/sms-twilio-view.php> and can be tested by texting 612-255-1559. The full integration into the journaling page of the site is a future enhancement.

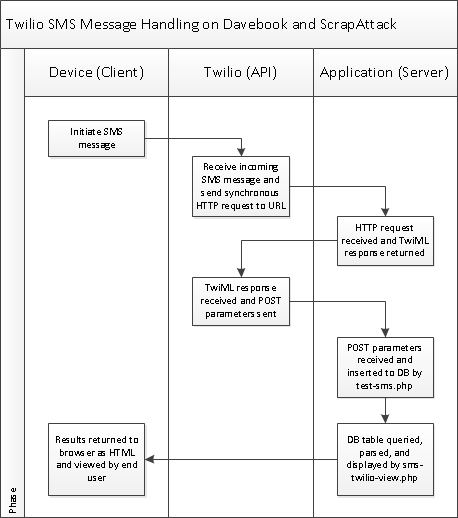


Figure 1: Twilio SMS process diagram of data flow.

**Results – Oauth 2.0**

The implementation of Oauth 2.0 provided considerably more challenges. I attempted to use the Google+ Oauth by setting up my Google Developer environment and attempting to implement the PHP Quickstart from the Google Developer subdomain. I was able to install the gplus-quickstart-php library and configure json using composer and integrate into my site. From there I was able to request the login/authorization page from Google+, generate the authentication code by authorizing the Google+ login, and generate the access token that returned data for the users Google+ profile. Unfortunately this implementation of Google+ Oauth did not return the users Gmail address or their Google+ ‘ID’. This resulted in not having one or both of the two data items I needed to integrate into my database and therefore the users profile on my site. My implementation of this Google+ Oauth can be viewed at <http://bromad.com//gplus-quickstart-php/signin.php>.

My next attempt was through enlisting Lloyd’s help to figure out what I had done wrong. He walked me through using a different library to access the Google+ Oauth API. The result is at <http://bromad.com//test1.php>. I am currently receiving the ‘Error: redirect\_uri\_mismatch’ and have had to put this aside for now while I work on my remaining homework. I received this error with my initial implementation and was able to work past it but I can’t afford to devote more time to Oauth at this stage.

I found Oauth 2.0 to be more challenging then I had initially expected. I believe it to be a valuable tool to facilitate a more streamlined authorization/authentication model but would like to see it more clearly defined in regard to its Google+ implementation.

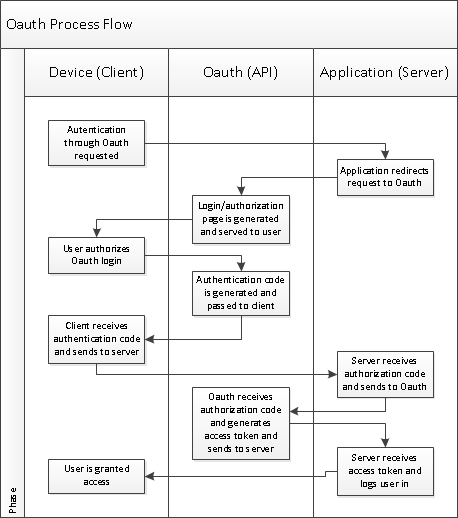


Figure 1: Oauth 2.0 process diagram of data flow.