ONE SOCIAL

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Fulfillment of the

Requirements for the

Degree Master of Computer Science

By

Sowmya Sampath

December 2010

ONE SOCIAL

By

Sowmya Sampath

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

ACKNOWLEDGEMENTS

I feel extremely grateful to Dr. Chris Pollett, for his encouragement, management and support. This project would not have been possible without his technical advice and guidance. I'm very thankful to the almighty who has given me the strength and courage to successfully complete my Master's program. I also want to thank my committee members Dr. Teng Moh and Dr. Mark Stamp, who took the time to discuss the project and offer me helpful suggestions and support.

I also would like to thank my husband, Prasanna Padmanabhan, and my parents, Sampath Kumar Narasimhan and Lalitha Sampath, for helping me with good suggestions and supporting me in countless ways. Last but not the least, I would like to thank my sisters, Ramya Sampath and Kavitha Padmanabhan, SJSU peer mentor, Manuel Silva, who proofread my report.

ABSTRACT

Facebook streams and Twitter tweets represent user content in two common social media systems. Users often have accounts on both these systems and to keep their content current for both of them they must update each separately. It is thus difficult to manage and maintain both of these social networks together. The goal of this project is to build a social website called One Social, where users can view their Facebook streams and Twitter tweets in one single view and thereby easily organize and maintain their updates. In One Social, users can enter a status message and can share it with both their Facebook and Twitter connections in just one click. Moreover, all the Facebook and Twitter comments for a status are threaded together to improve user experience. REST APIs are published for external developers to access and update the merged threaded streams and leverage the work done by OneSocial, when creating new social applications. These REST APIs also provide a common data format for both Facebook and Twitter streams, thereby making application development quicker and easier.

Table of Contents

1. Introduction

Social networks connect people with their friends and acquaintances. Users visit these sites to keep up with their friends, upload their photos and share links and videos. In the project developed for One Social, users can post and comment to messages, to both their Facebook and Twitter friends. Users can also view the messages posted by their friends in both these social networks as a single threaded view in OneSocial.

Various studies show that people connect with more people using social networks than they would connect with using emails and instant messages [10][11][12]. These social networks attract the younger generation by offering fast-paced, inexpensive online communication. The concept of social networks has been around for quite sometime now, but the newer generation of social networking sites such as Facebook, Twitter, MySpace, Friendster and Bebo have redefined the way users share and communicate information with their friends [13]. Facebook and Twitter together have over 700 million users as of today, and their growth rate is exponential [14][15].

In a recent study, it was found that many social networking users create their user profile in more than one social network [16]. Over a period of time, not only do these users get confused on what status they posted where, but also on what comments they received in which network as they might have different set of friends in different social networks. Eventually, users find it very hard to keep up with their connections in these different social networks.

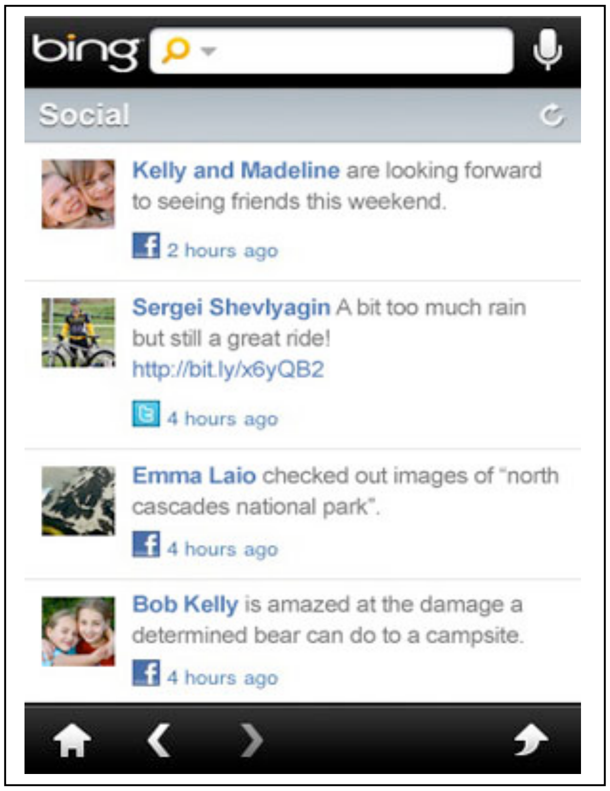
1.1 Competitors

Facebook and Twitter offer similar functionality. Many websites have combined the feeds from Facebook and Twitter because of their popularity. Bing's Social iPhone App and threadsy.com are two of the websites that pull in the feeds from Facebook and Twitter. It is useful to briefly look at the features of these products to understand the improved functionality that was achieved in our project, One Social.

Both these websites do an exceptionally good job of fetching all the feeds from Facebook and Twitter and displaying them in reverse chronological order. However, they still miss out on a few valuable features that would help users have better experience and engagement. Firstly, neither Bing nor Threadsy, thread Twitter statuses with their respective comments. Hence, for a given tweet, users do not have a clear way to identify comments to their tweets. Even, twitter.com does not thread tweets with their respective comments. Secondly, Bing and Threadsy do not thread the Facebook and Twitter messages together, even if the statuses are the same. Threading messages across social networks is even more challenging as the response data format would be different for different networks. Finally, neither Bing nor Threadsy update statuses to both Facebook and Twitter. Threadsy allows users to at-least post their status to either Facebook or Twitter. However, Bing's Social App doesn't perform any updates at all. Moreover, neither of these websites can render picture or video feeds from Facebook or Twitter.

One of the main goals of using a social network is to stay up-to-date with people's connections and friends. However, unless there is a better way to manage the communications between these social networks, users still find it hard to keep up with their connections. The following are the screenshots of these two websites that exemplify how their feeds are mixed.

Bing Social IPhone App



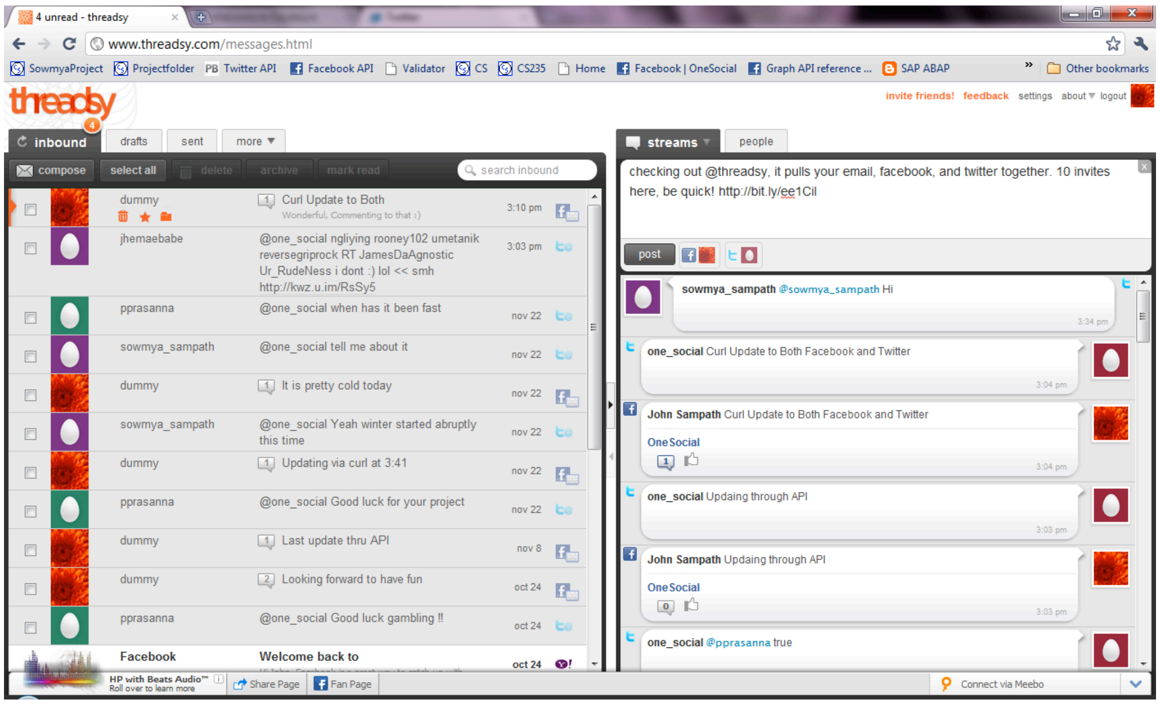
Kelly and Madeline are looking forward to seeing friends this weekend.
2 hours ago

Sergei Shevlyagin A bit too much rain but still a great ride! http://bit.ly/x6yQB2
4 hours ago

Emma Laio checked out images of "north cascades national park".
4 hours ago

Bob Kelly is amazed at the damage a determined bear can do to a campsite.
4 hours ago

Threadsy.com



dummy	Curl Update to Both Wonderful, Commenting to that :)	3:10 pm
jhemaebabe	@one_social ngliying rooney102 umetanik reversegriprock RT JamesDaAgnostic Ur_RudeNess i dont :) lol << smh http://kwz.u.im/RsSy5	3:03 pm
pprasanna	@one_social when has it been fast	nov 22
sowmya_sampath	@one_social tell me about it	nov 22
dummy	It is pretty cold today	nov 22
sowmya_sampath	@one_social Yeah winter started abruptly this time	nov 22
dummy	Updating via curl at 3:41	nov 22
pprasanna	@one_social Good luck for your project	nov 22
dummy	Last update thru API	nov 8
dummy	Looking forward to have fun	oct 24
pprasanna	@one_social Good luck gambling !!	oct 24
Facebook	**Welcome back to**	oct 24

checking out @threadsy, it pulls your email, facebook, and twitter together. 10 invites here, be quick! http://bit.ly/ee1Cil

sowmya_sampath @sowmya_sampath Hi 3:34 pm

one_social Curl Update to Both Facebook and Twitter 3:04 pm

John Sampath Curl Update to Both Facebook and Twitter
OneSocial 1 3:04 pm

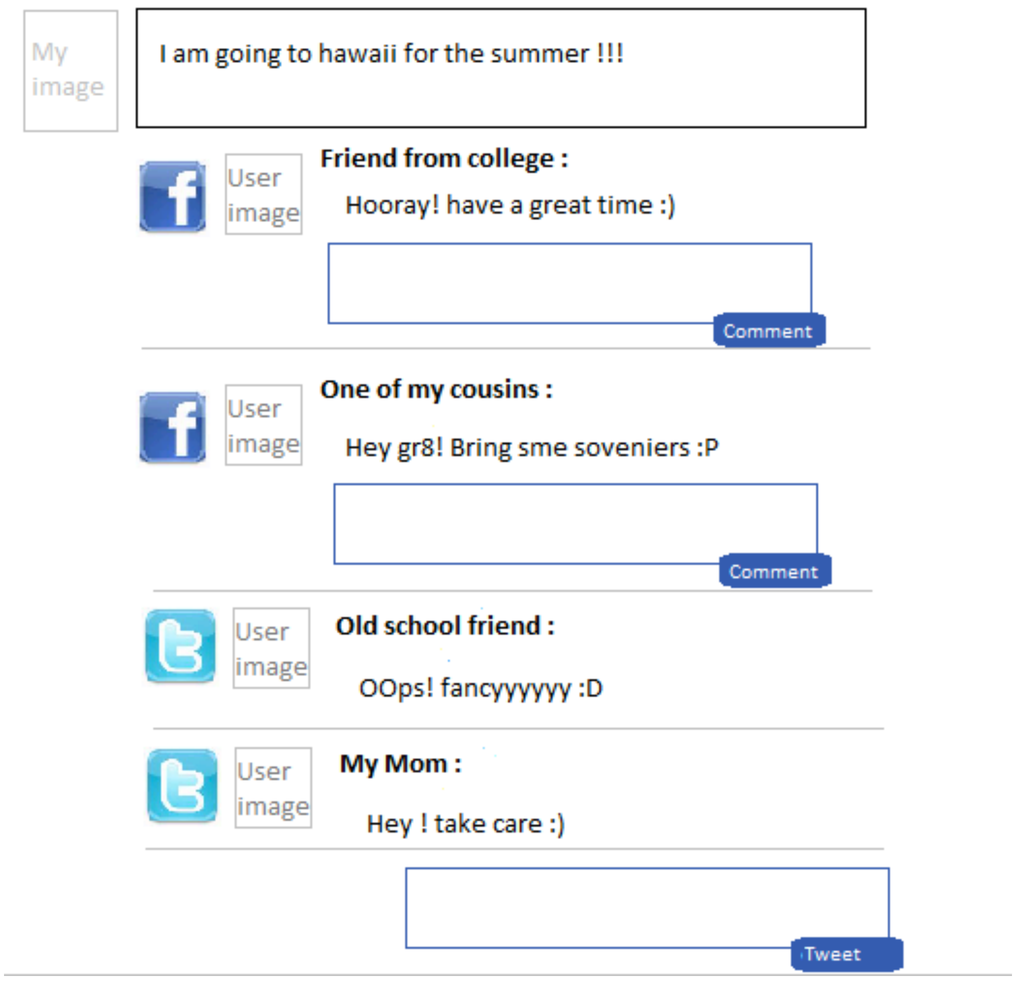
one_social Updaing through API 3:03 pm

John Sampath Updaing through API
OneSocial 0 3:03 pm

one_social @pprasanna true

1.2 OneSocial

The goal of this project was to build a social networking website that can leverage the features offered by both Facebook and Twitter and still be able to easily organize and maintain users and user's connection's status streams. Merging the feeds of both Facebook and Twitter, threading their messages and displaying them in a single view accomplished this goal. A single update to this website also in turn updates both Facebook and Twitter. A sample prototype of threading Facebook and Twitter feeds is shown below. A user will be able to see all the comments for a status posted in both Facebook and Twitter together in a single thread.



I am going to hawaii for the summer !!!

Friend from college :

Hooray! have a great time :)

One of my cousins :

Hey gr8! Bring sme soveniers :P

Old school friend :

OOps! fancyyyyyy :D

My Mom :

Hey ! take care :)

Finally, OneSocial should publish REST APIs of its implementation for external developers to take leverage on the work that has already been done by OneSocial to combine the two social networks.

1.3 Organization

The remainder of this paper is organized as follows. Section 2 discusses the design patterns and class diagrams used in OneSocial. Section 3 discusses the environment setup required for hosting OneSocial website and the steps needed to create a Facebook and Twitter application. The implementation details of OneSocial are covered in the subsequent sections. Section 4 discusses the authentication flow used in OneSocial and how these two networks are linked together. Section 5 discusses how status updates are posted from OneSocial. Section 6 discusses how these two social network streams are combined and threaded together. Section 7 discusses the web services API that OneSocial exposes to external developers. Section 8 discusses the front-end page rendering. Section 9 discusses various test cases developed for OneSocial. Section 10 concludes the paper and identifies future directions. Finally, user interface design screen shots for OneSocial are shown in the Appendix.

2. Design pattern

OneSocial was designed using MVC design pattern. Hence the modules for fetching and getting feeds from Facebook and Twitter are aligned with MVC design pattern. MVC design pattern is not only an apt framework for the current project; it also helps all the pieces to fall together. It is also the most commonly used and popular design pattern.

2.1 MVC Design Pattern

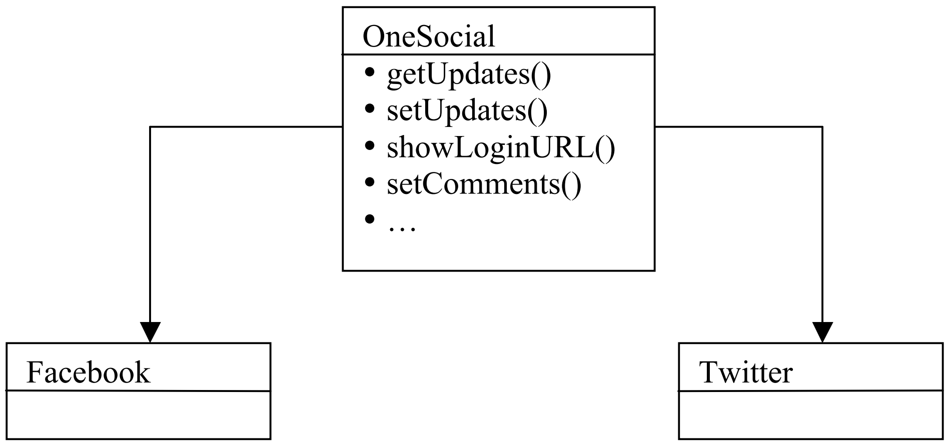
Model–View–Controller (MVC) is a software architecture, currently considered an architectural pattern used in software engineering. The pattern isolates "domain logic" (the application logic for the user) from the user interface (input and presentation), permitting independent development, testing and maintenance [1]. In OneSocial, Facebook.php and Twitter.php act as models that manage the behavior and data of these social networks and respond to requests for information about their states. OneSocial.php acts as the view to render the model into a webpage. Two different views, HTML and XML, exist for displaying the web page and the web service data respectively. Main.php acts as the controller that receives input and initiates a response by making calls on model objects.

2.2 Class Diagram

Facebook and Twitter have various common functionalities like

1. getUpdates
2. showLoginURL
3. setUpdates
4. setComments

OneSocial is designed in such a way that the common functionalities are implemented in a base or the parent class called OneSocial. This class contains the common methods for Facebook and Twitter. This class also contains the custom algorithm and the structure, which holds all the statuses, tweets and their comments.



The above class diagram helps us to get a gist of the overall code organization. The OneSocial is an abstract class and hence it has some functions defined and some functions abstract. The above-mentioned functions are all abstract as they have different implementations in their child classes.

3. Environmental Setup

3.1 Apache, PHP and MySQL Environment Setup

In this project, Apache is used as the web server. The release version of Apache 2.2 is first downloaded and installed. PHP 5.2.14 is then downloaded is used as the backend server side language. MySQL 5.1 that is used for storing user and session specific information. After installing Apache, PHP and MySQL following are some of the steps taken to setup a secure Apache web server that works together with PHP and MySQL. These changes are done in httpd-ssl.conf.

1) Listen: 443 is the default secure port that apache will be listening.
2) DocumentRoot: This will point to the directory, which will contain all the required PHP/CSS/Images files for OneSocial.
3) SSLEngine: This will be ON to enable SSL requests.
4) SSLCertificateFile: This will point to the PEM encoded server certificate. This certificate identifies that the server a user is connecting to be the correct OneSocial server.
5) SSLCertificateKeyFile: This will point to the SSL key file.
6) LoadModule php5_module: This loads the PHP Apache DLL that is required for running PHP web application in Apache.
7) RewriteEngine: This is turned ON for allowing rewrite rules. The most basic rewrite rule is to redirect all index.html, index.htm and index.php to Main.php, which is the entry point for OneSocial.

Curl and MySQL are not part of the default PHP settings and have to be configured manually in php.ini.

3.2 Facebook and Twitter Application Settings

The first step in creating OneSocial is to create a Facebook and Twitter application in their respective portals. http://www.facebook.com/developers/createapp.php and http://twitter.com/apps/new are where we can create new Facebook and Twitter applications respectively.

After creating a Facebook application, Application ID, API key and an Application secret will be issued and these need to be used in all further communications with Facebook. Similarly, after creating a Twitter application, a consumer key and a consumer secret will be issued and they need to be used in all further communications with Twitter. Following are the list of parameters that are required to successfully setup a Facebook/Twitter application:

1) Site URL: OneSocial's web application URL. For OneSocial, this URL is set to https://onesocial.dyndns.org/.
2) Application Name and Description: As the name implies, this is the name of the application that is created with its associating description for new users to know about the application.
3) Post-Authorize Redirect URL: Facebook and Twitter redirects users to this URL when they first authorize the application. For OneSocial, this redirect URL is set to https://onesocial.dyndns.org/callback.php?fb=1 and https://onesocial.dyndns.org/callback.php?tw=1 respectively.

4) Application Type: OneSocial is a web application, and hence, the application type should be web.
5) Contact Email: This is the email address where OneSocial users can reach out.

Every time we change any of these settings, Facebook and Twitter will update the secret. Alternatively, application developers could also reset this secret as and when needed for security reasons. Facebook and Twitter will return a return code 403 if application use an invalid secret.

3.3 Database Schema

There are primarily 2 MySQL tables used for OneSocial. One is the User table, which stores the meta-information of all OneSocial users. The other is the Sessions table, which stores the Facebook and Twitter request and access tokens of OneSocial users. There is a 1:N relationship between the Users and the Sessions table. Following is the schema and sample records in these tables:

Users table

UserID	Password	FirstName	LastName
Sowmya.sampath	**********	Sowmya	Sampath
john.doe	********	John	Doe

Sessions table

UserID	AppID	Session Token	Req Token
Sowmya.sampath	Facebook	343gkfgf0343mflfdd3mll43343439	Rern3knff9dfgddrefdhbf
Sowmya.sampath	Twitter	5knkfg9565fknkppcletitnfpd6ee67	Gfe973nsog5gds3tojl74f

4. Authentication and Authorization

Facebook and Twitter API specifications can be found at:

1. Facebook wiki - http://developers.facebook.com/docs/reference/api/
2. Twitter wiki - http://apiwiki.twitter.com/Twitter-API-Documentation

The first iteration of OneSocial was developed using Facebook's old REST APIs, which had a lot of issues including problems with post authorization callback and access token's lifetime. Eventually, they were marked as deprecated by Facebook itself. Now with the introduction of Facebook's new Graph APIs, OneSocial was rewritten using these new APIs. Facebook's graph APIs is similar to Twitter's REST APIs, and hence, we can follow a similar structure and the integration just falls into place.

Facebook or Twitter APIs have a common format, which is described as follows. http://<url-for-the-api>.com/<version-number>/<function-name>/<format>. An example would be - http://api.twitter.com/v1/statuses/update.json

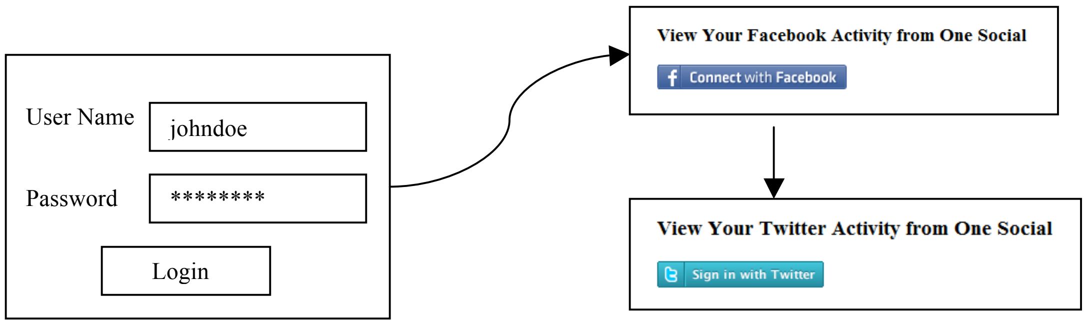
This phase is dedicated to Login, Authentication and Authorization. Facebook and Twitter handle the Login in different ways. OneSocial uses a single sign-on methodology where when the user logs-in, they can associate his OneSocial id, with their Facebook and Twitter accounts. OneSocial does not handle the authentication for Facebook and Twitter. Rather, the security is passed over to the respective social networks. It also helps in a way, if the user has signed into Facebook and Twitter in the same browser session then they need not renter their credentials. Hence, users do not feel unsecured to be in an alien site to provide their login information for Facebook and Twitter. HTTPS protocol is also used to ensure that the credentials are passed over a secured network. The code implementation starts based on the user's login information that is being provided. For first time users a link is provided when they login to create a new account with OneSocial. User types in the following information:

- First Name
- Last Name
- User id
- Password
- Confirm Password

These details are fetched from the page and stored into the database. While storing the password a hash is taken and then stored. This is to ensure further security. If his details are successful he is passed on to the next steps. On the case if the user id already exists for OneSocial, he is prompted with a message saying that the userid already exits and hence he creates a new user id.

For returning users, they just login with their username and password. Hence, as soon as the user enters his credentials for OneSocial, the authenticity of the user is compared with the database entries and validated. Based on the comparison the user is provided with a feedback. If the user finds out that he has an invalid user name or password by the feedback message, he can attempt to login again.

After the user clears his credential checks he is next provided with steps to associate his Facebook and Twitter Account. The above figure depicts the singe sign-on with One Social and how the redirection for the individual sign-ins to Facebook and Twitter happens.


User Name
johndoe
Password
****
Login
View Your Facebook Activity from One Social
Connect with Facebook
View Your Twitter Activity from One Social
Sign in with Twitter

4.1 Password Security

When the user details are stored in the database table, everything except the password is stored in clear. The hash of the password is stored. This is because hashing is a one-way process, and hence, any hack done on the database table will not reveal the passwords to the hacker. Since hashing is one way, to verify that the user after he has logged in, the password has to be hashed again before it is compared with the database entry.

Not only this, user credentials are passed via HTTPS protocol, also the website has its own SSL certificate leading to a secured login/authentication. Hence gains users confidence and trust. More on HTTPS protocol is discussed in Phase VI.

4.2 Facebook Authentication

Facebook uses the OAuth 2.0 protocol for user authentication and application authorization. The first step in authenticating a user is to obtain an access token. Following are the steps required to obtain an access token:

a) Redirect the user to https://graph.facebook.com/oauth/authorize and pass the application id and post authorize callback URL as parameters to this API.

b) User enters their credential in the above Facebook URL and after successfully authorizing the application, Facebook would redirect the user to the authorize callback url along with a verification string in the argument code, which can be exchanged for an OAuth access token.

c) Generate an access token using the above verification code by fetching https://graph.facebook.com/oauth/access_token

Once the access token is obtained, all further communications to Facebook will only require this token instead of the user's credentials. There are various authorization levels, which can be requested from an OneSocial Facebook user by passing arguments to https://graph.facebook.com/oauth/authorize. They are as follows:

1) publish_stream: If the user authorized this level, OneSocial will be able to post statuses, comments and likes to a user's and their friends' stream.
2) offline_access: If the user authorized this level, OneSocial will be able to perform authorized requests on behalf of the user at any time.
3) create_event: If the user authorized this level, OneSocial will be able to create and modify events on the user's behalf.
4) sms: If the user authorized this level, OneSocial will be able to send messages to the user and respond to messages from the user via text message.

Currently, for OneSocial, only publish_stream and offline_access permission levels are requested for OneSocial users.

Code Snippet

```
public function getSession($user, $args) {
              $sessionData = parent::getSessionFromDB($user,
Constants::FACEBOOK_APP_ID);
              if (!empty($sessionData['secret'])) {
                     return $sessionData;
              }

              if (!empty($args['code'])) {
                     $out = $this->getAccessToken($args['code']);
                     list ($key, $accessToken) = split("=", $out);
                     parent::insertSessionIntoDB($user,
Constants::FACEBOOK_APP_ID, null, $accessToken);
                  return array('token' => null, 'secret' => $accessToken);
              } else {
                     return $sessionData;
              }
}
```

4.3 Twitter authentication

Similar to Facebook, Twitter also supports OAuth for authentication. Following are the three steps needed to acquire an access token for Twitter, which would eventually be used in all further communications with twitter for user identification purposes.

1. The first step to authenticating a user in Twitter is to obtain an unauthorized request token from twitter using http://api.twitter.com/oauth/request_token. The consumer key we obtained during app creation is also passed to this API. Any request that we make to Twitter should include a signature for security purposes. HMAC-SHA1 is used on the API URL, its parameters and headers (if applicable) to

generate the request signature. Twitter returns a 403 if the signature does not match with the one that is generated by Twitter after receiving the request.

2. In the next step, the user is taken to https://api.twitter.com/oauth/authorize where the user is asked to enter their credentials and is authenticated by Twitter. This API also takes a callback URL where the user will be redirected upon successful authentication and user authorization to OneSocial.

3. The last step is to exchange the request token for an access token, which would be used in all subsequent requests to Twitter. This is done by calling https://api.twitter.com/oauth/access_token and by passing the request token, consumer key and the request signature.

The code snippet for this algorithm is as shown following:

Code Snippet

```
public function getSession($user, $args) {

    $sessionData = parent::getSessionFromDB($user, Constants::TWITTER_APP_ID);
    if (!empty($sessionData['secret']) && !empty($sessionData['token'])) {
       return $sessionData;
    }

    if (!empty($args['oauth_token'])) {
       $auth         = $this->getAccessToken($args['oauth_token']);
       $authToken     = $auth['oauth_token'];
       $authSecret    = $auth['oauth_token_secret'];
       $this->insertSessionIntoDB($user, Constants::TWITTER_APP_ID, $authToken,
$authSecret);
       return (array('token' => $authToken, 'secret' => $authSecret));
    } else {
       return $sessionData;
    }
}
```

Both the request and access tokens are stored in a MySQL database for the user's Facebook and Twitter accounts. These tokens are not restricted with time unless Facebook/Twitter changes their policy and make them expire after a particular period of time. When a user logs into OneSocial, access tokens from the database is retrieved and used directly, if they are available. All the above requests to obtain an access token are made only when they are not available in the MySQL database. Hence, a user will have to link their Facebook and Twitter accounts only once in OneSocial.

Twitter, in contrast to Facebook, does not offer various levels of user authorization. By default, once a twitter user authorizes an application, the application will be able to send requests on user's behalf unless the user explicitly removes the authorization. In the case of Facebook, this does not happen by default and happens only when the user explicitly grants offline_access.

4.4 Sessions and Cookies

In OneSocial, user sessions are maintained in a cookie. Cookies are used to determine the user who logged into OneSocial for the current browser session. The cookie domain is set to onesocial.dyndns.org. The name of the cookie is onesocial_user. The value of the cookie, which is the (one way) hash of the OneSocial user, is set as soon as a user logs into OneSocial. This cookie is cleared as soon as the user logs off or the browser session is closed. When this cookie is already set, the users credentials are not required to interact with OneSocial. PHP's setCookie() and global variable $_COOKIES are used to set and get the user's session cookie.

5. Publish Stream

Once the user is authenticated and the application authorized, the next obvious step is to read and write statuses or tweets from OneSocial. In this section, we will dive deep into the write or update use case. Facebook and Twitter APIs give us an easy way to set the statuses/tweets and also comments on existing statuses or tweets. PHP curl libraries are used to make all the HTTP web services requests. The functionality of setUpdates API is discussed prior moving to the functionality of setComments API.

5.1 Facebook Set Updates

When a user updates their status from OneSocial, the following API with the necessary parameters is called to update the status in Facebook.

API Usage:

a) URL: https://graph.facebook.com/me/feed

b) Request Parameters:

    a. Access Token – This is the user's secret that uniquely identifies a user

    b. Status – The status message to be posted to Facebook

c) Request Method: POST

d) Response data: 1 on success, 0 on failure or an error code.

Code Snippet:

```
public function setUpdates($authSecret, $authToken, $status) {
              $xPost['access_token'] = $authSecret;
              $xPost['message']      = $status;

              $ch = curl_init();
           curl_setopt($ch, CURLOPT_URL,
"https://graph.facebook.com/me/feed");
           curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
              curl_setopt($ch, CURLOPT_POSTFIELDS, $xPost);
           curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'POST');

           curl_exec($ch);
}
```

5.2 Twitter Set Updates

When a user updates their status from OneSocial, apart from calling the above Facebook API, the following API with the necessary parameters are also called to update the status in Twitter simultaneously.

API Usage:

a) URL: http://api.twitter.com/1/statuses/update.json
b) Request Parameters:
    a. Auth token – This is the user's secret that uniquely identifies a user
    b. Status – The status message to be posted
    c. Signature – This is the hash of the request URL and all the request parameters.
c) Request Method: POST
d) Response data: Returns HTTP status 403 when the status message is more than 140 characters or when the current status is the same text as the most recently posted

status. On success, all the meta information of the status message and the authenticated user is returned.

Code Snippet

```
public function setUpdates($authSecret, $authToken, $status) {

              $url     = "http://api.twitter.com/1/statuses/update.json";
           $params = array('oauth_consumer_key'     => self::API_KEY,
                           'oauth_token'                      => $authToken,
                           'oauth_signature_method' => 'HMAC-SHA1',
                           'oauth_timestamp'           => time(),
                           'oauth_nonce'                      =>
md5(microtime() . mt_rand()));
           $sigParams = $this->getSignedHeader($url, $params, array('status'
=> $status), 'POST', $authSecret);

           $ch = curl_init();
           curl_setopt($ch, CURLOPT_URL, $url);
           curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
           curl_setopt($ch, CURLOPT_HTTPHEADER, array($sigParams));
           curl_setopt($ch, CURLOPT_POST, 1);
           curl_setopt($ch, CURLOPT_POSTFIELDS, "status=$status");

           $out = curl_exec($ch);

           if(curl_errno($ch)) {
               print 'Curl error: ' . curl_error($ch); exit;
           }

           $tweets = json_decode($out, true);
       }
```

5.3 Set Comments

Setting Comments to statuses or feeds in Facebook is slightly different from posting statuses. This is because when posting statuses, we do not have to reference the status id that gets generated from the post. Whereas, while posting comments we need to refer the status id for which the comment or tweet needs to be posted.

API Usage:

a) URL: https://graph.facebook.com/STATUS_ID/comments. The Status ID, for which the comment has to be added, should be replaced in the above url.

b) Request Parameters:

    a. Access Token – This is the user's secret that uniquely identifies a user

    b. Message – The comment to be posted to Facebook.

c) Request Method: POST

Response data: 1 on success, 0 on failure or an error code.

6. Fetch, Group and Thread Feeds

The next obvious step is to provide a mechanism to fetch or read all the statuses or tweets from Facebook and Twitter and to thread these two feeds together. Facebook and Twitter APIs also give us an easy way to read a user's statuses and tweets.

Facebook.com, by default, combines all the comments of a given status together into a single thread. Similarly, their APIs also include the comments to the statuses within the same json structure. Twitter on the other hand, does not handle threading of the comments within their tweets. Hence an initial threading has to be implemented within Twitter tweets and its replies or comments.

Further, additional threading needs to be implemented between Facebook and Twitter feeds to show a single view of a user's Facebook and Twitter friends' comment on a common post. The getUpdates() method in Facebook and Twitter class handles fetching both the statuses and their comments and threading them together. To ease the threading between the feeds from Facebook and Twitter a common data structure is used in which all the statuses and the comments from Facebook, tweets from Twitter are collected.

6.1 Facebook Get Stream

API Usage:

a) URL: https://graph.facebook.com/me/home?access_token=SECRET. SECRET should be replaced by the actual user's access token.

b) Request Parameters: None

c) Request Method: GET

d) Response data: On success, Facebook returns the user's News feed in a json format. In the event of a failure, Facebook returns an appropriate HTTP error code and a message.

The response structure from Facebook for each status is of the following form:

```
[id] => 1437286239_172765289409397
[from] => (
       [name] => John Doe
       [id] => 1437286239
      )

[message] => It is so cold in SFO this winter
[type] => status
[created_time] => 2010-11-26T19:10:59+0000
 [updated_time] => 2010-11-26T22:26:20+0000
```

The response structure with sample data of comments for the above status is of the following form:

```
[comments] => (
       [data] => (
               [0] => Array (
                     [id] => 1437286239_172765289409397_2081304
                     [from] => Array (
                            [name] => Will Rafael
                            [id] => 3434451
                                     )
                     [message] => Welcome to my life in New York
                     [created_time] => 2010-11-26T21:11:12+0000
                          )
                         [1] => Array (
                            [id] => 1437286239_172765289409397_2081655
                            [from] => Array (
                                     [name] => Glen Mcgrath
                                     [id] => 634622076
                                     )
                     [message] => How cold is it now?
                     [created_time] => 2010-11-26T22:26:20+0000
                         )
             )
       [count] => 2
)
```

The entire Facebook news feed is a collection of statuses with each status containing a collection of comments.

Code Snippet

```
$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, https://graph.facebook.com/me/home?access_token=$secret");
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_TIMEOUT_MS, 10000);

$out  = curl_exec($ch);
$info = curl_getinfo($ch);

// Likely that there is something wrong with the session
if ($info['http_code'] != 200) {
       return false;
}

$data   = json_decode($out, true);
$stream = $data['data'];
```

6.2 Twitter Get Tweets

API Usage:

a) URL: http://api.twitter.com/1/statuses/home_timeline.json

b) Request Parameters:

   a. Auth token – This is the user's secret that uniquely identifies a user

   b. Signature – This is the hash of the request URL and all the request parameters.

c) Request Method: POST

d) Response data: On success, the top 20 most recent tweets, including re-tweets if they exist, posted by the authenticating user and the user's they follow. In the event of a failure, appropriate http code and an associated error message is returned.

Following is the sample response of the Twitter Get Stream API:

```
<statuses type='array'>
<status>
  <created_at>Fri Jul 16 16:58:46 +0000 2010</created_at>
  <id>18700887835</id>
  <text>got a lovely surprise from @craftybeans</text>
  <source>web</source>
  <in_reply_to_status_id></in_reply_to_status_id>
  <favorited>false</favorited>
  <user>
    <id>29733</id>
    <name>cindy li</name>
    <screen_name>cindyli</screen_name>
    .
    .
    .
  </user>
</status>
…

</statuses>
```

CodeSnippet

```
$params = array('oauth_consumer_key'     => self::API_KEY,
                'oauth_token'            => $authToken,
                'oauth_signature_method' => 'HMAC-SHA1',
                'oauth_timestamp'        => time(),
                'oauth_nonce'            => md5(microtime() . mt_rand()));

$sigParams = $this-getSignedHeader("http://api.twitter.com/1/statuses/home_timeline.json",
$params,  array(), 'GET', $authSecret);

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL,"http://api.twitter.com/1/statuses/home_timeline.json");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_HTTPHEADER, array($sigParams));

$out = curl_exec($ch);

if(curl_errno($ch)) {
     print 'Curl error: ' . curl_error($ch); exit;
}

$tweets  = array();
$updates = json_decode($out, true);
foreach ($updates as $update) {
       $id = '' . $update['id'];
       $tweets[$id] = $update;
}
```

6.3 Threading Facebook and Twitter feeds together

Following are the goals to be achieved by threading Facebook and Twitter feeds together:

1) Single view of what is happening in both Facebook and Twitter worlds and to get users socially up-to-date in an easier fashion.

2) Threading messages within a given topic would help users keep track of their context better.

3) Threading messages could also drive users to try both Facebook and Twitter and eventually drive user growth in both these social networks.

The algorithm to thread messages is as follows:

1) As we have mentioned earlier, the stream format is very different between Facebook and Twitter. Hence, the first objective would be to convert these streams formats into a standard format called as OneSocial stream format. Following is the sample OneSocial stream format:

```
<message>
       <id>1437286239_172765289409397</id>
       <image>https://graph.facebook.com/1437286239/picture</image>
       <user>Kalyan Sundaram</user>
       <text>Break the Rules for next 6 months.....</text>
       <app>Facebook</app>
       <time>12 hour(s) ago</time>
       <likes>1</likes>
       <comments>
         <image>https://graph.facebook.com/1437286239/picture</image>
         <user>Kalyan Sundaram</user>
         <text>
             @Harjot : Yes
             @Bala : yes,early  start to enjoy the Thanks Giving :-)
             @Maha : Cheers .. Party lam already started.   unforgettable
             start :-)
          </text>
          <app>Facebook</app>
          <time>51 minute(s) ago</time>
       </comments>
</message>
```

2) Twitter, as said before, does not group messages even in their website. Hence, the next step is to identify a way to group twitter messages. Every tweet in Twitter has an associated "id" for it. Fortunately, twitter exposes another meta data for each tweet called "in_reply_to_status_id", which could be used to easily group messages. Tweets that do not have "in_reply_to_status_id" are considered the Root tweets. Any tweet that has an "in_reply_to_status_id" set to the "id" of the Root tweet is considered as its Comment Tweet. Comment Tweets can still be Root tweets for sub-comment tweets and so on and so forth. By the end of Step 2, we would be able to successfully chain common tweets within Twitter.

3) Use array merge in PHP to merge the results between Twitter and Facebook with keys to each Facebook and twitter feed being its feed created time.

4) Perform a numeric sort on the resultant array keys to give us the merged feeds sorted by created time. Parse through the sorted feeds array and figure out if any of the messages have the same text. If so, those messages are grouped with their associated comments into a single feed.

```
    foreach ($oneSocialResponse as $key => $response) {
        foreach ($oneSocialResponse as $ikey => $iresponse) {
            if ($key != $ikey && $response['0']['text'] == $iresponse['0']['text'])
{
                $found = true;
                break;
            }
        }
        if ($found == true ) {
            if ($key < $ikey) {
                $finalResponse[$key] = $response;
                if (isset($iresponse['0']['message'])) {
                    foreach ($iresponse['0']['message'] as $msg) {
                        $finalResponse[$key]['0']['message'][] = $msg;
                    }
                }
            }
        } else {
            $finalResponse[$key] = $response;
        }
```

7. Feed Rendering

This section is where the display or the rendering for the pages is covered. The HTML code sections are present in the base class to be used common to all the sub-classes. Even though Facebook and Twitter classes are the derived classes from OneSocial, they are not the ones that are called upon for rendering. The code runs based on a class called Main.php. This class creates objects for all the other classes and calls the appropriate functions. Since the feeds from Facebook and Twitter are time dependent and the relevancy to the user depends on when they are posted, a custom algorithm was devised to calculate the time since a status was posted. Some examples are like “15 seconds ago”, “1 hour ago”, and “1 month ago”. This aids in identifying the most recent messages the user needs to concentrate, and also to locate a particular piece of message, which was posted a specific time ago.

The algorithm to determine this time gap is as follows:

1) The time difference (in Unix timestamp) between the system timestamp and the message timestamp is obtained.
2) If the time difference is less than 60 – The message must be posted seconds ago.
3) If the time difference is less than 60 * 60 – The message must be posted minutes ago.
4) If the time difference is less than 60 * 60 * 24 – The message must be posted hours ago.
5) If the time difference is less than 60 * 60 * 24 * 7 – The message must be posted days ago.

7.1 Rendering HTML pages

This section is dedicated to the front-end rendering. The CSS or the Cascading Styling sheet was used extensively for designing the front-end of the website. Although designing the website was not the main theme of this project, good efforts were put on coming up with a good background, font and color for the website. The front page or the login page is well suited with a custom designed image. The inner page which displays the Facebook and the Twitter feeds also makes use of the common CSS. While displaying the Facebook and the Twitter feeds special attention is given to differentiate the statuses or the tweets from the comments obtained. All the images that have been used in the CSS are custom designed. The screen shots for the front-end design are attached in the Appendix section.

8. Publishing a REST API

This section describes the functionalities in OneSocial that are offered as REST APIs. The code path for the REST API is exactly the same as the regular request flow. The only difference between the REST API and the regular request is that the output format for the REST API is XML, whereas the output format for the regular request is HTML.

As mentioned in earlier sections, the same response structure is maintained for both Facebook and Twitter in OneSocial. An external developer who would like to tie another blooming social networking site to Facebook and Twitter will have to simply convert the network specific feed format into OneSocial format. The three REST APIs that are published are as follows:

1) Set Updates: This API is used to write a status message into both Facebook and Twitter. Following are the API attributes:

a) URL: https://onesocial.dyndns.org/api/v1/setupdates
b) Request Parameters:
    a. User ID – OneSocial user ID
    b. Password – The password of the OneSocial user
    c. Status – The status message to be updated in both Facebook and Twitter
c) Request Method: POST

d) Response data: On success, the ID of the status is returned. In the event of a failure, appropriate http code and an associated error message is returned.

Following is a sample curl command used to call SetUpdates REST API:

```
curl -k "https://onesocial.dyndns.org/api/v1/setupdates" -d
        "userid=master&pwd=master&status=Example Status Update"
```

The “k” option in curl surpasses the ssl certificate authentication. As OneSocial, does not yet offer a rusted SSL certificate, users are confronted with a question to trust the certificate. The option “k” explicitly allows curl to perform insecure SSL connections and transfers. All SSL connections are attempted to be done secure by using the CA certificate bundle installed by default. Here the values for the userid, password and the status are passed via POST. The setUpdates API allows the developer to set a specific update. In the above example “Example Status Update” is the status that is set to OneSocial, which in-turn updates both Facebook and Twitter. After the curl is executed, the status id of the posted status is returned. This helps the developer to locate the statuses, or in the case of setting comments to these statuses.

2) Set Comments: This API allows an external developer to set comments for a particular status. Following are the API attributes:

a) URL: https://onesocial.dyndns.org/api/v1/setupdates

b) Request Parameters:

    a. User ID – OneSocial user ID

b. Password – The password of the OneSocial user

c. Status ID – The ID of the status message for which the comment has to be applied

d. Comment – The comment to be updated in both Facebook and Twitter

c) Request Method: POST

d) Response data: On success, the ID of the status is returned. In the event of a failure, appropriate http code and an associated error message is returned.

For the SetComments functionality the sample curl command would look like:

```
curl -k "https://onesocial.dyndns.org/api/v1/setupdates" -d
           "userid=master&pwd=master&statusID=100001530595145_15804212
           7570475&comment=Response thru API"
```

The above set of parameters used look similar to SetUpdates API except for the additional information such as the status-id and the comment for that particular status. This comment sent to OneSocial, identifies the application type for the status and updates the comment in the appropriate social network.

3) Get Updates: This API allows an external developer to read both the Facebook and Twitter stream in a single feed. The feed will also be threaded appropriately in the correct reverse order of time. Following are the API attributes:

a) URL: https://onesocial.dyndns.org/api/v1/getupdates

b) Request Parameters:

a. User ID – OneSocial user ID

b. Password – The password of the OneSocial user

c) Request Method: POST

d) Response data: On success, the combined stream of Facebook and Twitter is returned. In the event of a failure, appropriate http code and an associated error message is returned.

To do a curl on the Get Updates API the following command is used.

```
curl -k "https://onesocial.dyndns.org/api/v1/getupdates" -d
         "userid=master&pwd=master"
```

Code Snippet

```
if ($xml) {
  print "<message>";
  if (isset($oneResponse['id'])) {
  print "<id><![CDATA[{$oneResponse['id']}]]></id>";
}
  print "<image><![CDATA[{$oneResponse['image']}]]></image>";
  print "<user><![CDATA[{$oneResponse['user']}]]></user>";
  print "<text><![CDATA[{$oneResponse['text']}]]></text>";
  if ($oneResponse['app'] == Constants::FACEBOOK_APP_ID) {
       print "<app>Facebook</app>";
  } else {
       print "<app>Twitter</app>";
  }
} else {
  print "<tr><td valign='top'><img height='50px' width='50px' align='top'
src='" . $oneResponse['image'] . "'/></td>";
  print "<td colspan='2' valign='top'><b>" . $oneResponse['user'] .
"</b>&nbsp;&nbsp;" . $oneResponse['text'] . "<br/>";
}
```

Following are the rewrite rules set in apache to support these REST APIs:

1. RewriteRule ^/api/v1/getupdates /Main.php?format=xml

2. RewriteRule ^/api/v1/setupdates /Main.php?format=xml

9. Test Cases

9.1 Unit Testing

Unit testing was done on the code using different inputs, which validated the code. This will be briefed in the following sections. The code is written in a modular fashion; hence each function was individually tested and combined them and tested as a whole. Login and user authentication testing was one of the modules that were tested. PHP test scripts that fed different inputs like proper user credentials and also incorrect user name and password generated output and feedback messages. Users were notified while creating a login for OneSocial if a user id already exists with the same name. Users were also notified if they tried to login with an incorrect user-name or password.

9.2 API Testing

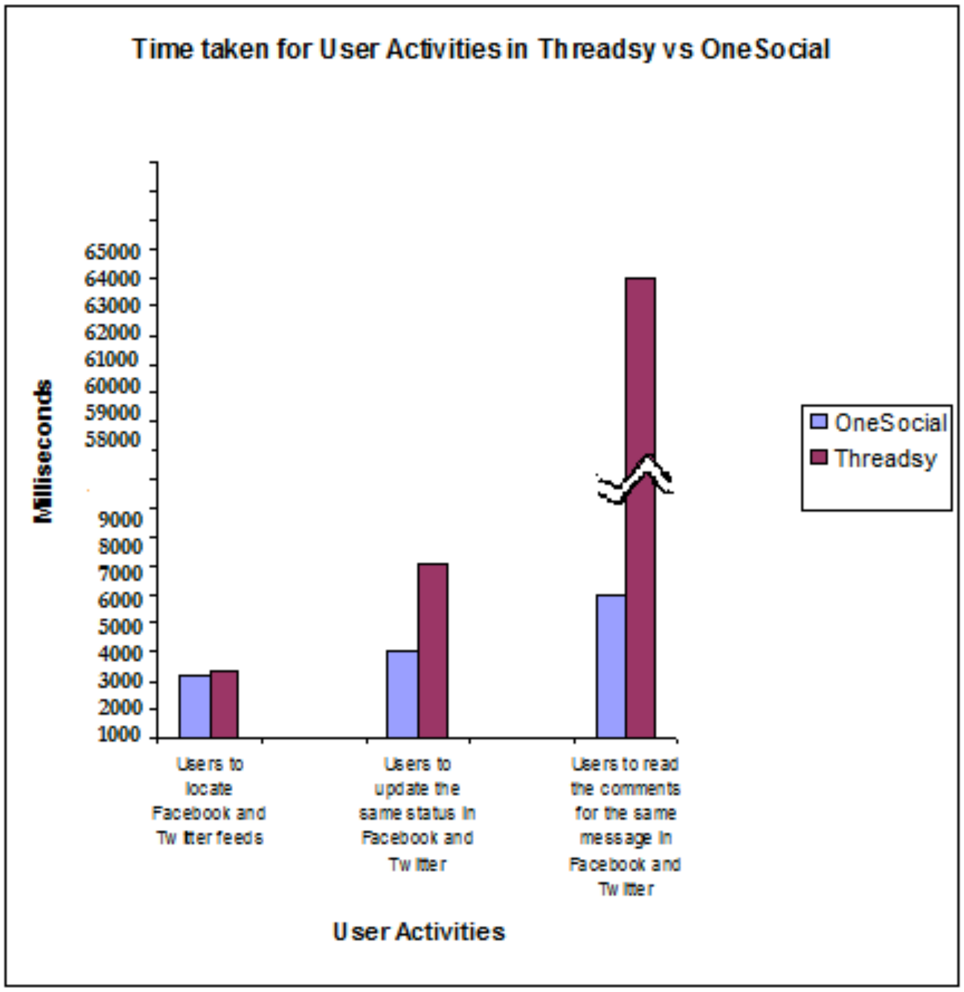
This project utilized a lot of Facebook and Twitter open source APIs. All the APIs used were first tested with curl commands and small php scripts to ensure their validity. These APIs are used by Facebook developers around the world who might attach the Facebook feeds to their website homepages. Facebook and Twitter constantly keep updating some of the API formats or the structure of the data returned by these APIs. For example, Facebook had changed its API from REST to Graph APIs. This greatly improved the performance of the code and helped while integrating with the Twitter API. Twitter, on the other hand, had changed the structure of the data returned by the API. The json structure had an element tag named "id" and "in_reply_to_status_id" which were used for the status id and to mention the id of the status to which it is in reply to. Twitter recently changed the tags to "id_str" and "in_reply_to_status_id_str" to expand the ids to the large number of user load. This change could break the code hence PHP scripts were used to test the APIs every time they curl and changes need to be made based the API modifications.

If Facebook or Twitter is down, OneSocial should not face the same fate. Hence while curling each API a timeout of specific number of seconds is maintained and hence if either of the sites are down, OneSocial does not wait and continues retrieving feeds from the other one. The timeout seconds also depend on the API that needs to be curled. For a read stream the timeout is 3 seconds and for an update stream it is 2 seconds. For showing the login url of Facebook and Twitter the timeout is just 500 milli seconds.

9.3 Usability Testing

Since OneSocial is a website, it requires a good amount of usability testing. Many aspects that are familiar to the user were utilized while designing the website having Login page and Registration page similar to other websites and displaying the messages and comments similar to Facebook. Images and designing was custom done using Photoshop and Microsoft Paint. Further, the website was compared with Threadsy, a competitor to analyze its effectiveness and the performance. A group of 5 users was picked with varying age, sex, computer knowledge, and social networking interests and was asked to perform 3 different tasks to analyze the usability and user performance testing of OneSocial when compared to Threadsy.

First, the users were asked to search for a particular message and comment from Facebook and Twitter and via Threadsy and OneSocial. The users were able to locate the messages within 2 seconds as shown in the below chart. It was the same time taken by both Threadsy and OneSocial.



Next, the users were asked to post the same message to Facebook and Twitter using both Threadsy and OneSocial. Users had to repeat the message for Facebook and Twitter and ended up typing the same message twice while using Threadsy. Users just had to type the message once in the case of OneSocial. This can be noticed in the chart where there is a considerable difference between the performance of OneSocial and Threadsy.

Thirdly, users were asked to read the status and its comments they updated previously to Facebook and Twitter. While using Threadsy, users had to move back and forth to read the status and comments, as they were not threaded together as the same message, which was possible in OneSocial. As seen in the above chart OneSocial had performed the threading in 4 to 5 seconds whereas for Threadsy it was around a minute.

10. Conclusion

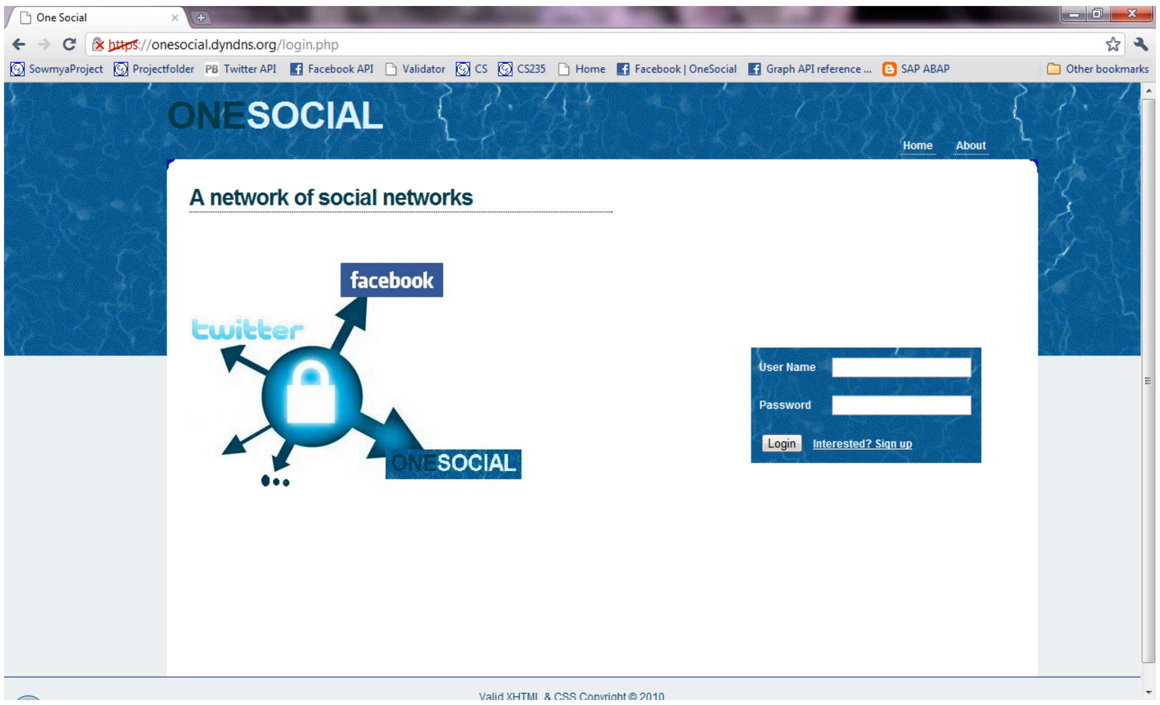
While Facebook and Twitter have changed the lives of millions of users and how to share social information in the Internet, people still find it hard to manage these different social networks and stay up-to-date with their friends in different networks. Bing Social and threadsy.com were able to combine the feeds from Facebook and Twitter. However, they lacked various other features that OneSocial supports to address the issue of maintaining multiple networks. OneSocial was able to fetch feeds from both Facebook and Twitter and, more importantly, combines and threads these two feeds together, which enables users to maintain context in all conversations and eventually give a better social networking experience. Any updates from OneSocial were also able to successfully update in both Facebook and Twitter.

The published REST APIs were tested for various user accounts and were seen to return the expected combined threaded views of the streams. One Social was tested in various browsers such as Internet Explorer, Safari, Firefox and Chrome to find that all its CSS and Javascript were working as expected. Both the website and the REST APIs were tested when Twitter was slow and completely down, and the requests timed-out as expected and made sure that Twitter did not taken OneSocial down along with it.

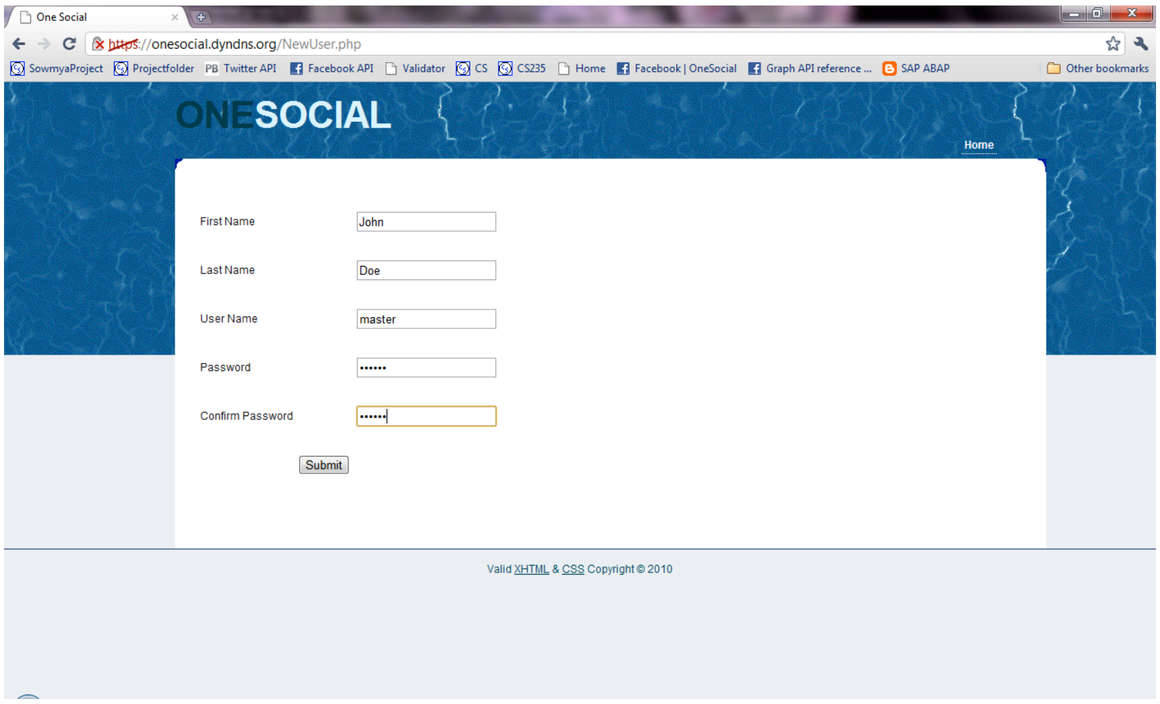
A group of 5 students from SJSU was chosen to test One Social. Based on their usage patterns, it was found that they were able to manage and stay up-to-date with their friends by using One Social.

APPENDIX

Login Page of OneSocial

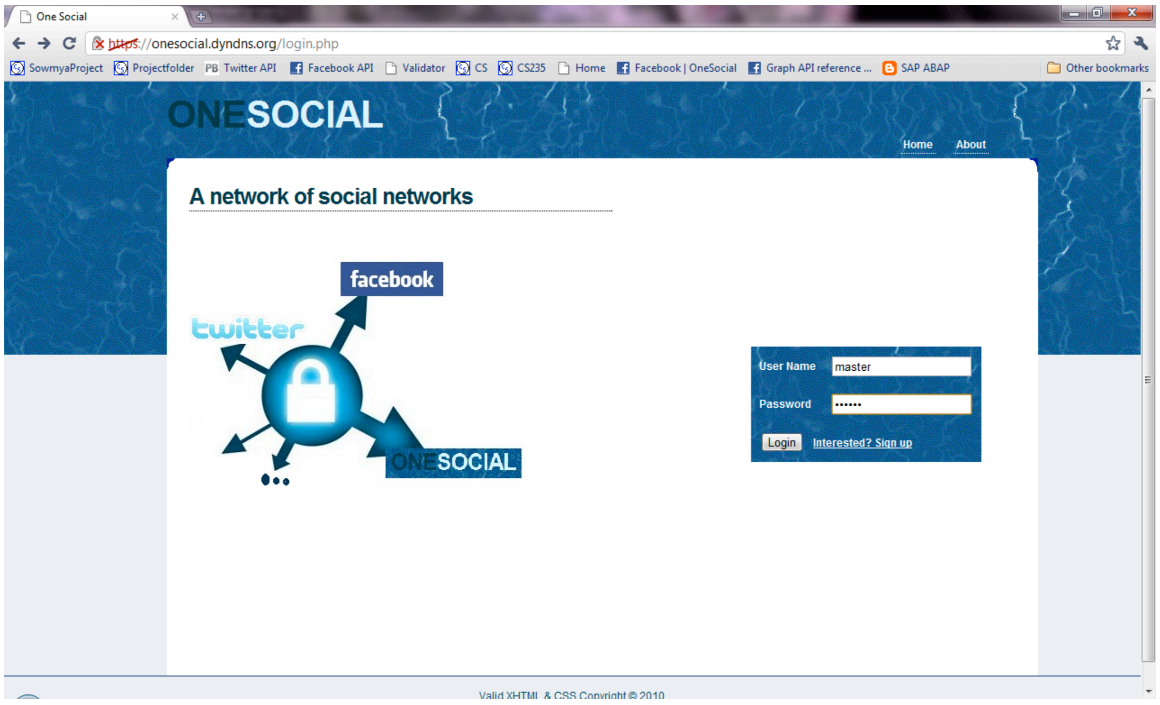


Create a New User

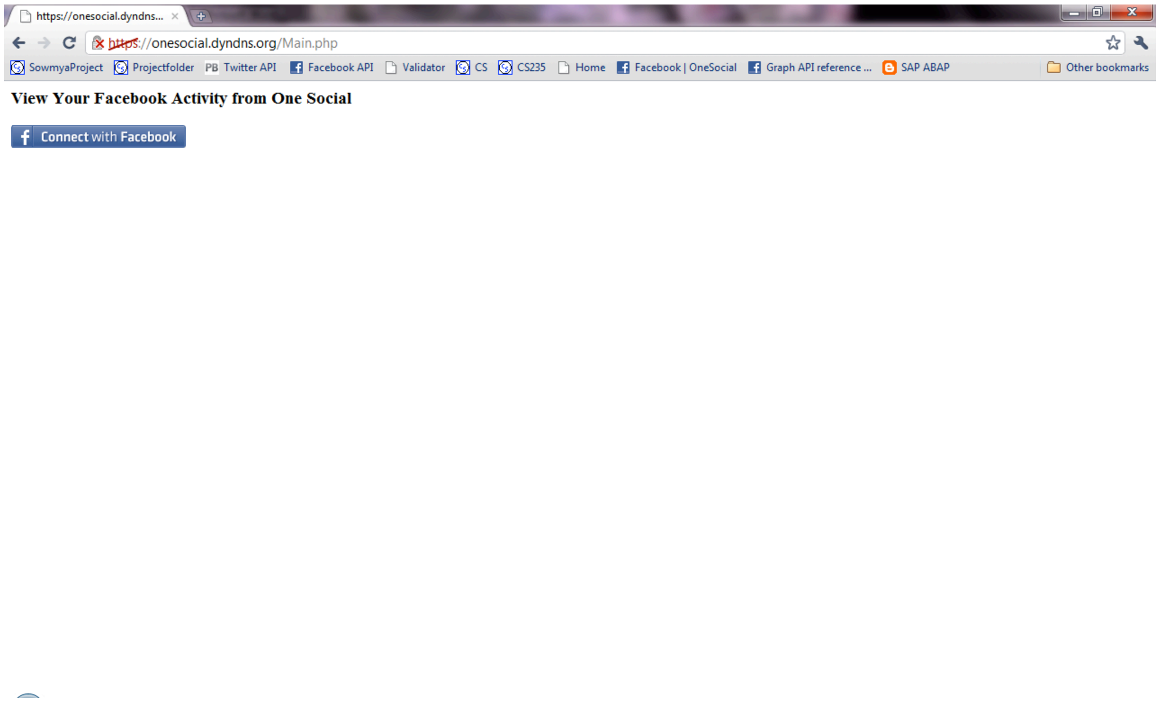


First Name	John
Last Name	Doe
User Name	master
Password	••••••
Confirm Password	••••••

Login with the User id and Password

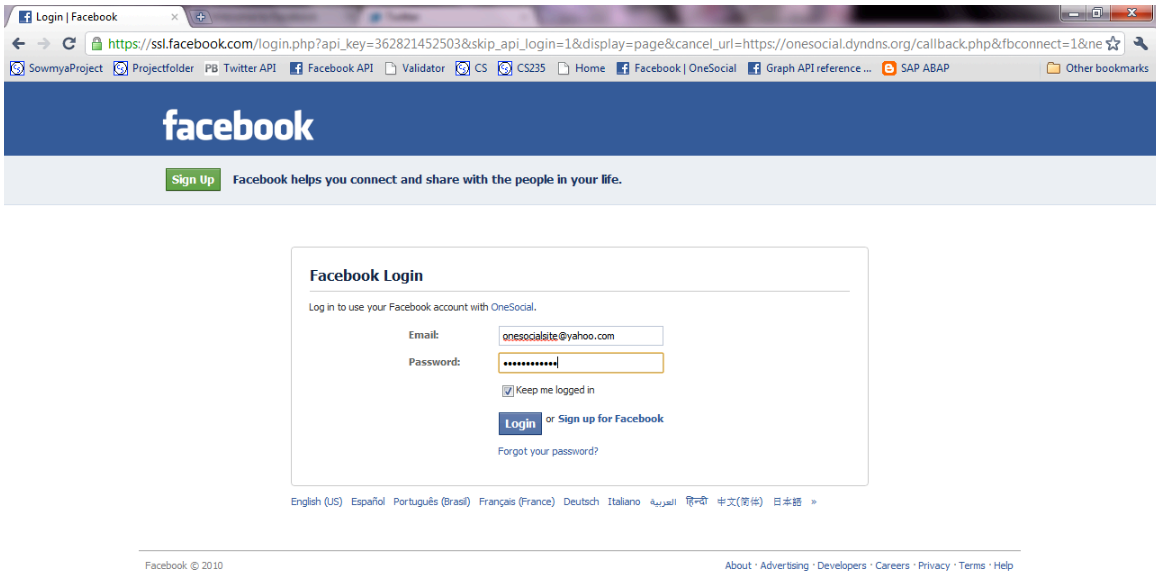

One Social
https://onesocial.dyndns.org/login.php
SowmyaProject Projectfolder Twitter API Facebook API Validator CS CS235 Home Facebook | OneSocial Graph API reference ... SAP ABAP Other bookmarks
ONESOCIAL
Home
About
A network of social networks
facebook
twitter
ONESOCIAL
User Name
master
Password
••••••
Login
Interested? Sign up
Valid XHTML & CSS Copyright © 2010

Link OneSocial Account with Facebook Account


https://onesocial.dyndns...
https://onesocial.dyndns.org/Main.php
SowmyaProject Projectfolder Twitter API Facebook API Validator CS CS235 Home Facebook | OneSocial Graph API reference ... SAP ABAP Other bookmarks
Connect with Facebook

View Your Facebook Activity from One Social

Login to Facebook



https://ssl.facebook.com/login.php?api_key=362821452503&skip_api_login=1&display=page&cancel_url=https://onesocial.dyndns.org/callback.php&fbconnect=1&ne

Login | Facebook
SowmyaProject Projectfolder Twitter API Facebook API Validator CS CS235 Home Facebook | OneSocial Graph API reference ... SAP ABAP Other bookmarks
facebook
Sign Up

Facebook helps you connect and share with the people in your life.

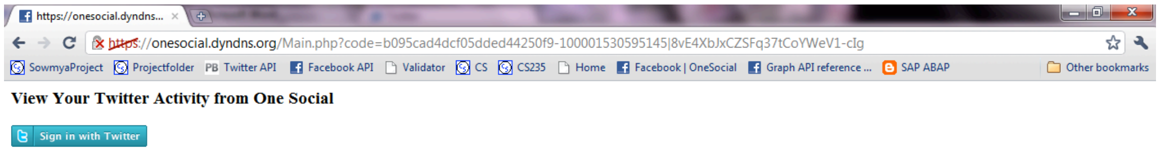
Facebook Login

Log in to use your Facebook account with OneSocial.

Email:	onesocialsite@yahoo.com
Password:	••••••••••••

Keep me logged in
Login
or Sign up for Facebook
Forgot your password?
English (US) Español Português (Brasil) Français (France) Deutsch Italiano العربية हिन्दी 中文(简体) 日本語 »
Facebook © 2010
About · Advertising · Developers · Careers · Privacy · Terms · Help

Link OneSocial Account with twitter Account

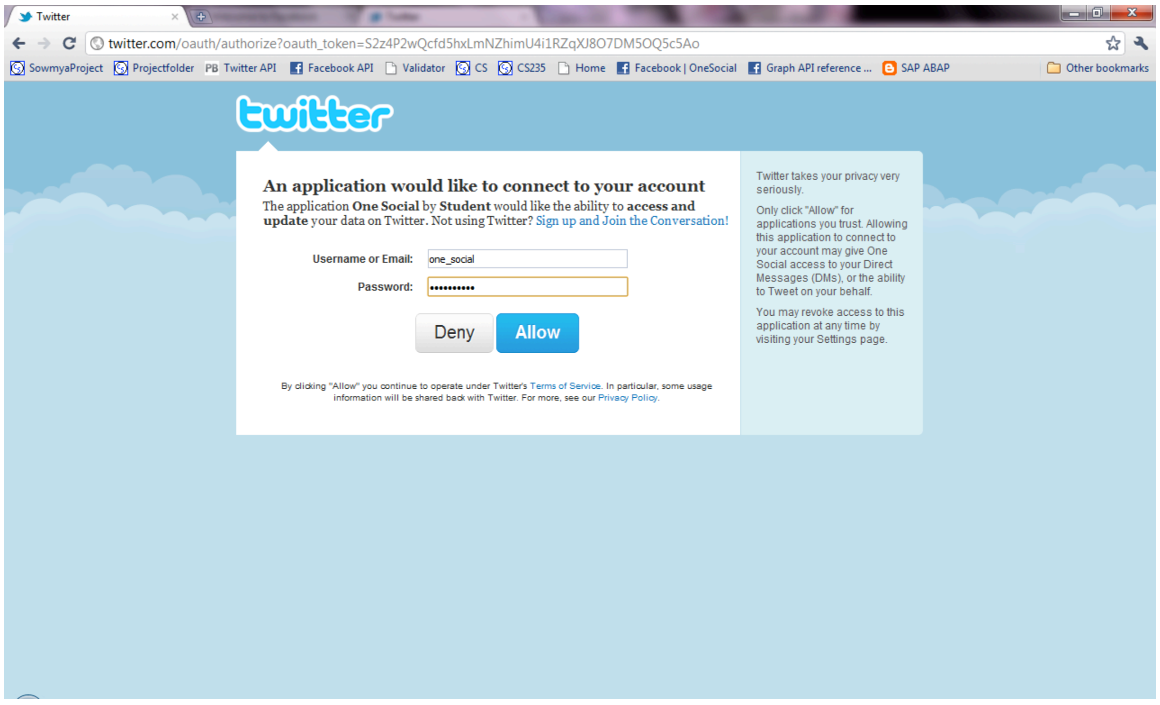


~~https~~://onesocial.dyndns.org/Main.php?code=b095cad4dcf05dded44250f9-100001530595145|8vE4XbJxCZSFq37tCoYWeV1-cIg

https://onesocial.dyndns...
SowmyaProject Projectfolder Twitter API Facebook API Validator CS CS235 Home Facebook | OneSocial Graph API reference ... SAP ABAP Other bookmarks

View Your Twitter Activity from One Social

Login to Twitter account



twitter.com/oauth/authorize?oauth_token=S2z4P2wQcfd5hxLmNZhimU4i1RZqXJ8O7DM5OQ5c5Ao

An application would like to connect to your account

The application **One Social** by **Student** would like the ability to **access and update** your data on Twitter. Not using Twitter? Sign up and Join the Conversation!

Username or Email: one_social

Password: ••••••••••

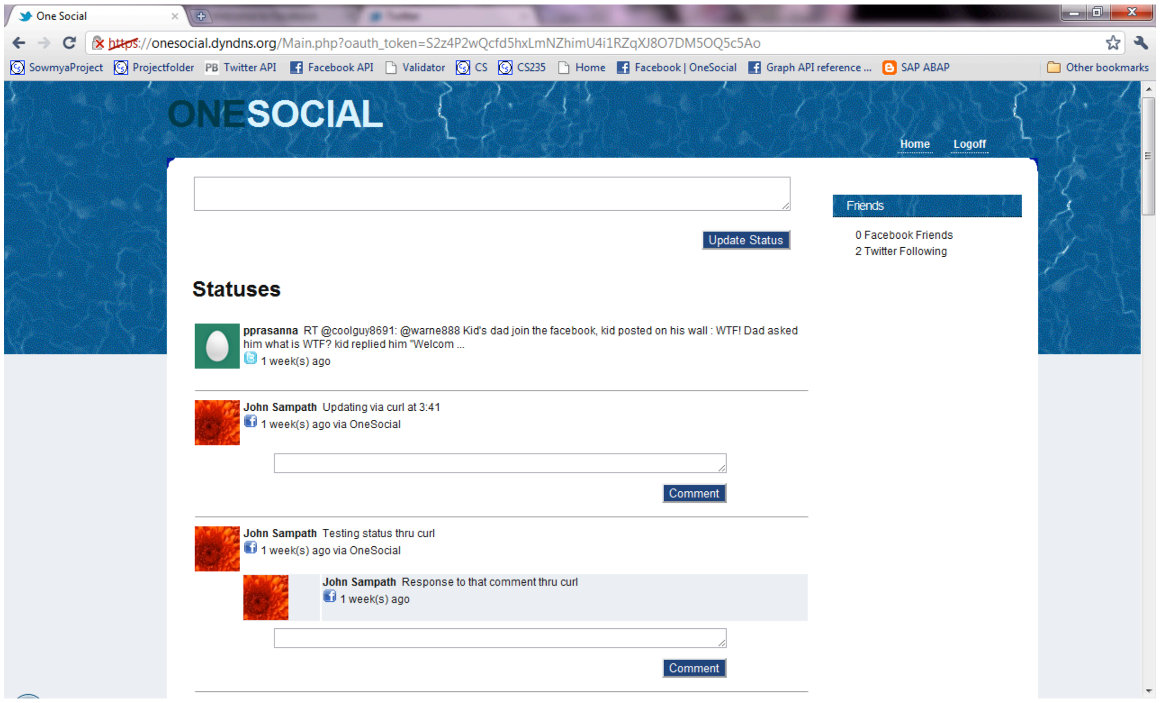
By clicking "Allow" you continue to operate under Twitter's Terms of Service. In particular, some usage information will be shared back with Twitter. For more, see our Privacy Policy.

Twitter takes your privacy very seriously.

Only click "Allow" for applications you trust. Allowing this application to connect to your account may give One Social access to your Direct Messages (DMs), or the ability to Tweet on your behalf.

You may revoke access to this application at any time by visiting your Settings page.

Feeds from Facebook and Twitter threaded together



https://onesocial.dyndns.org/Main.php?oauth_token=S2z4P2wQcfd5hxLmNZhimU4i1RZqXJ8O7DM5OQ5c5Ao

0 Facebook Friends
2 Twitter Following

Statuses

pprasanna RT @coolguy8691: @warne888 Kid's dad join the facebook, kid posted on his wall : WTF! Dad asked him what is WTF? kid replied him "Welcom ...
1 week(s) ago

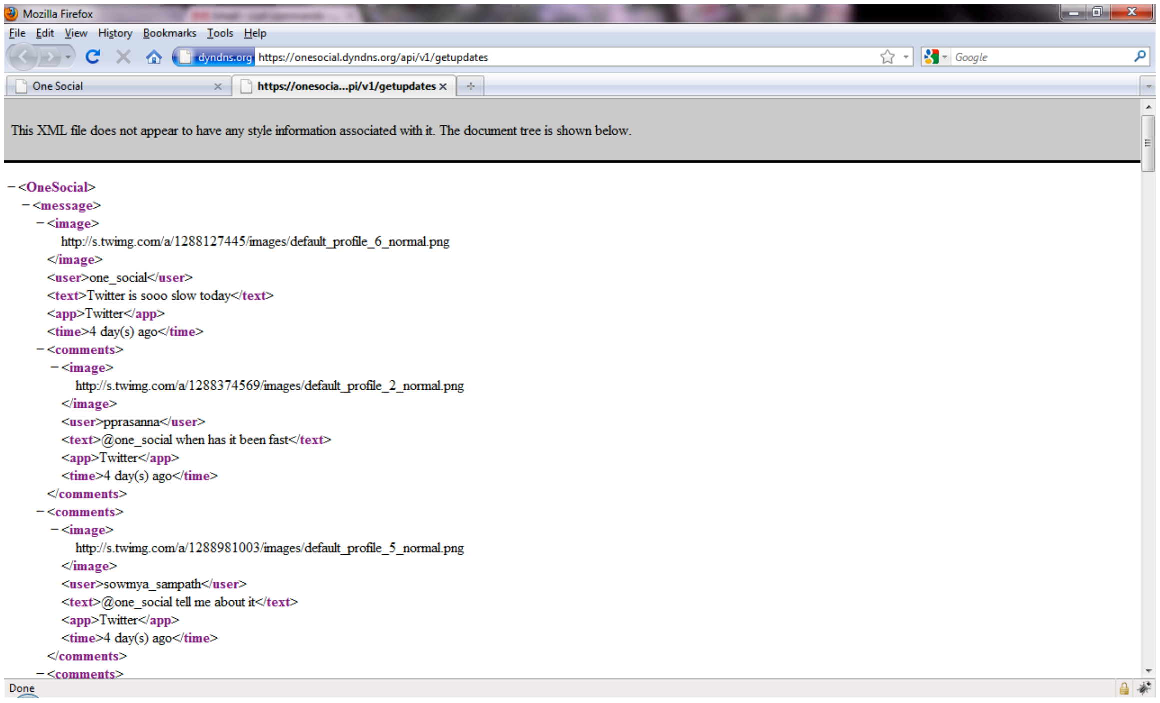
John Sampath Updating via curl at 3:41
1 week(s) ago via OneSocial

John Sampath Testing status thru curl
1 week(s) ago via OneSocial

John Sampath Response to that comment thru curl
1 week(s) ago

Publishing REST API

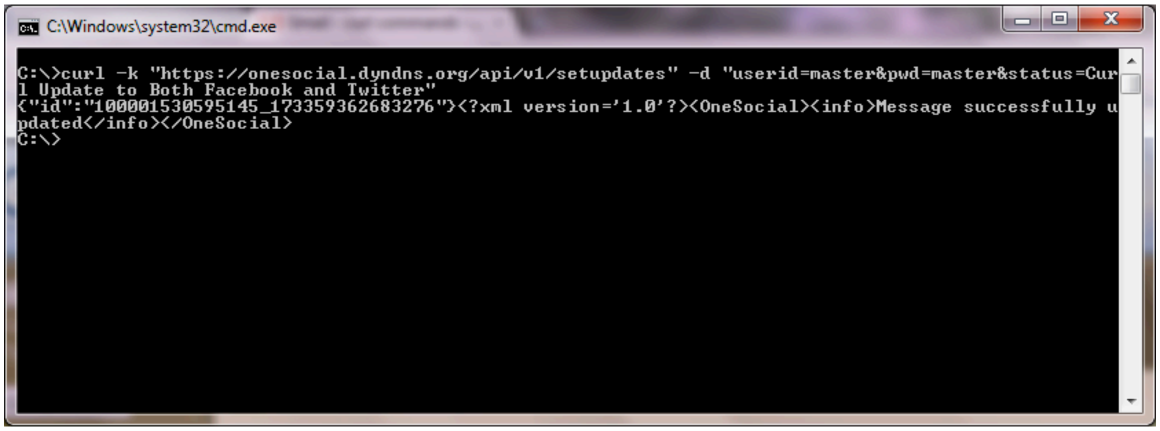
Curl – Get Updates from both Facebook and Twitter


Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://onesocial.dyndns.org/api/v1/getupdates
One Social
https://onesocia...pi/v1/getupdates
Done

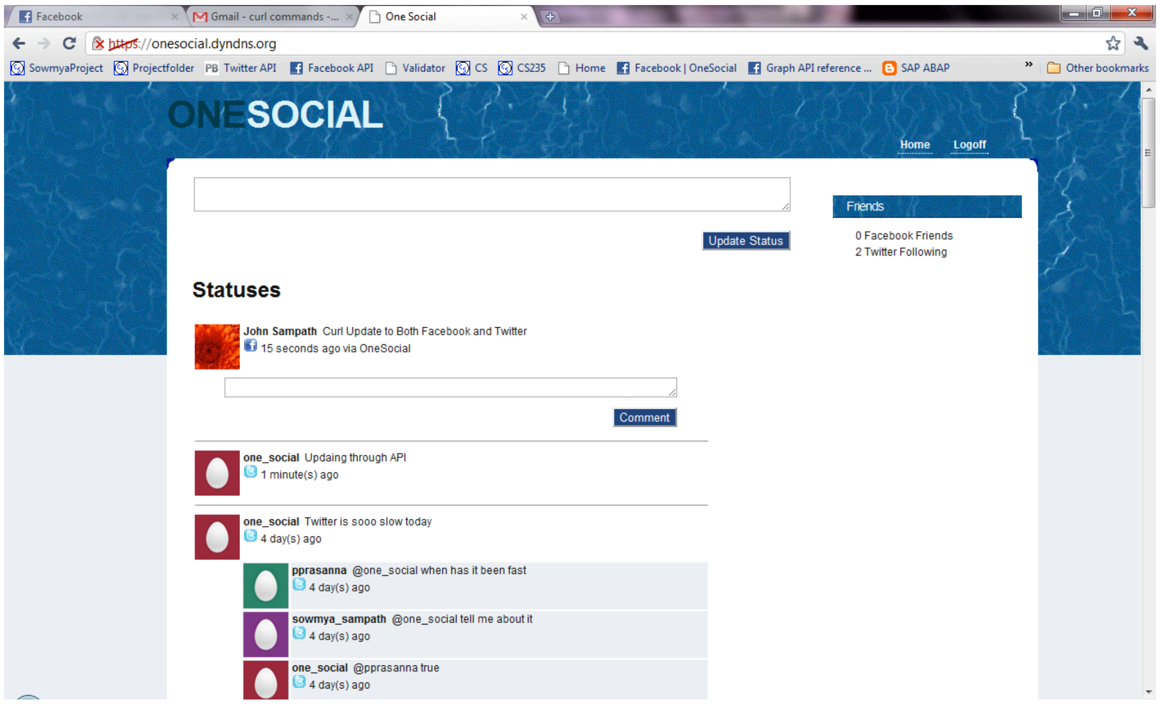
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
− <OneSocial>
  − <message>
    − <image>
        http://s.twimg.com/a/1288127445/images/default_profile_6_normal.png
      </image>
      <user>one_social</user>
      <text>Twitter is sooo slow today</text>
      <app>Twitter</app>
      <time>4 day(s) ago</time>
    − <comments>
      − <image>
          http://s.twimg.com/a/1288374569/images/default_profile_2_normal.png
        </image>
        <user>pprasanna</user>
        <text>@one_social when has it been fast</text>
        <app>Twitter</app>
        <time>4 day(s) ago</time>
      </comments>
    − <comments>
      − <image>
          http://s.twimg.com/a/1288981003/images/default_profile_5_normal.png
        </image>
        <user>sowmya_sampath</user>
        <text>@one_social tell me about it</text>
        <app>Twitter</app>
        <time>4 day(s) ago</time>
      </comments>
    − <comments>
```

Curl – Set status to both Facebook and Twitter



```
C:\>curl -k "https://onesocial.dyndns.org/api/v1/setupdates" -d "userid=master&pwd=master&status=Cur
l Update to Both Facebook and Twitter"
{"id":"100001530595145_173359362683276"}<?xml version='1.0'?><OneSocial><info>Message successfully u
pdated</info></OneSocial>
C:\>
```



0 Facebook Friends
2 Twitter Following

Statuses

John Sampath Curl Update to Both Facebook and Twitter
15 seconds ago via OneSocial

one_social Updaing through API
1 minute(s) ago

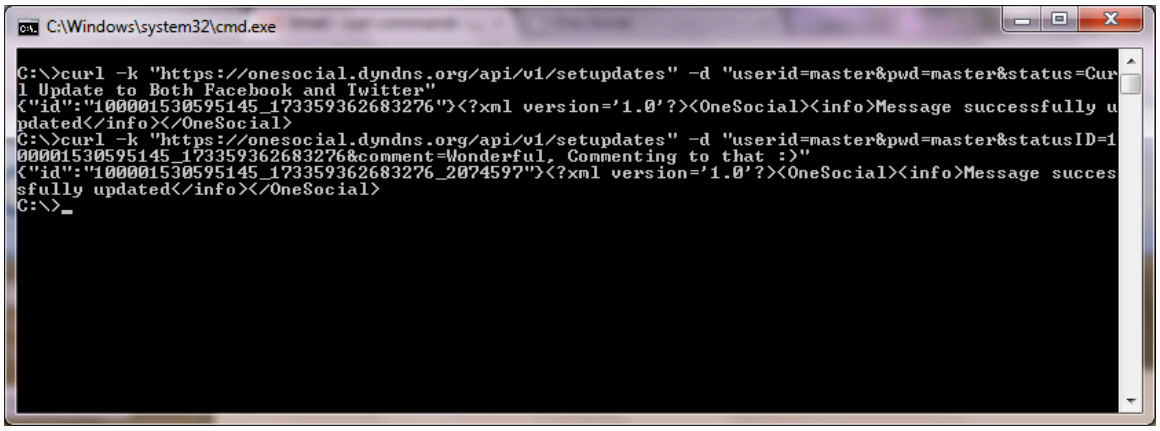
one_social Twitter is sooo slow today
4 day(s) ago

pprasanna @one_social when has it been fast
4 day(s) ago

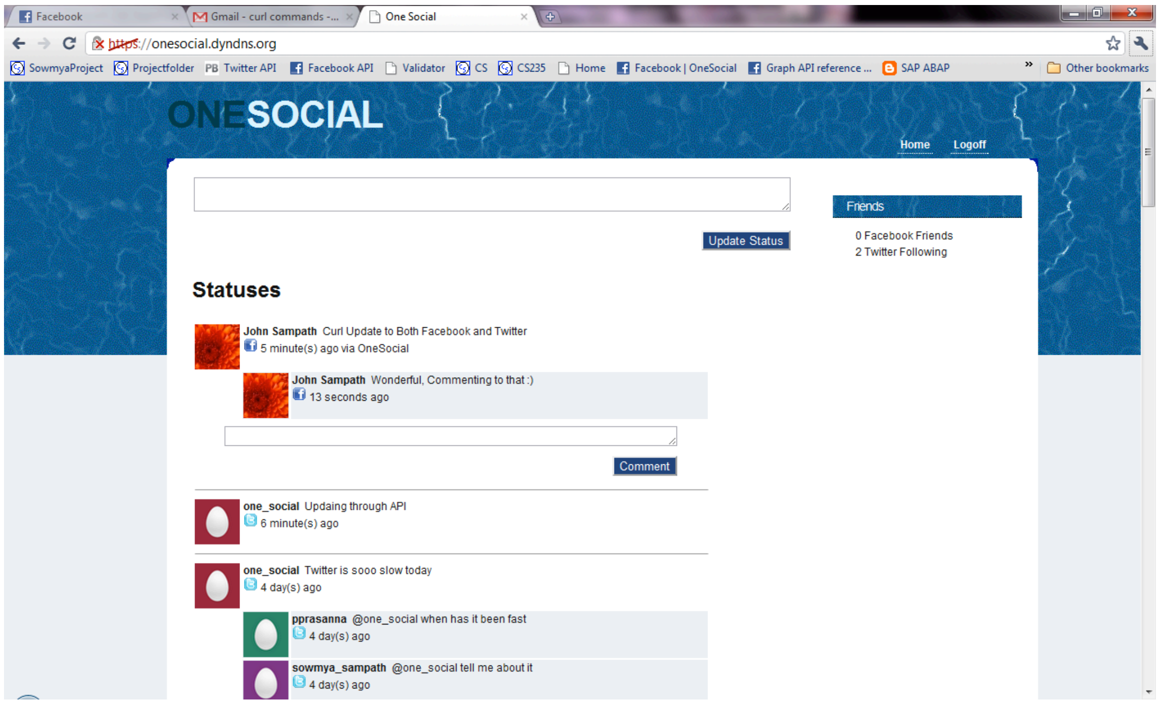
sowmya_sampath @one_social tell me about it
4 day(s) ago

one_social @pprasanna true
4 day(s) ago

Curl – Set status to both Facebook and Twitter (to the above status)



```
C:\>curl -k "https://onesocial.dyndns.org/api/v1/setupdates" -d "userid=master&pwd=master&status=Cur
l Update to Both Facebook and Twitter"
{"id":"100001530595145_173359362683276"}<?xml version='1.0'?><OneSocial><info>Message successfully u
pdated</info></OneSocial>
C:\>curl -k "https://onesocial.dyndns.org/api/v1/setupdates" -d "userid=master&pwd=master&statusID=1
00001530595145_173359362683276&comment=Wonderful, Commenting to that :)"
{"id":"100001530595145_173359362683276_2074597"}<?xml version='1.0'?><OneSocial><info>Message succes
sfully updated</info></OneSocial>
C:\>_
```



0 Facebook Friends
2 Twitter Following

Statuses

John Sampath Curl Update to Both Facebook and Twitter
5 minute(s) ago via OneSocial

John Sampath Wonderful, Commenting to that :)
13 seconds ago

one_social Updaing through API
6 minute(s) ago

one_social Twitter is sooo slow today
4 day(s) ago

pprasanna @one_social when has it been fast
4 day(s) ago

sowmya_sampath @one_social tell me about it
4 day(s) ago

References

[1] "Model-View-Controller". In Wikipedia. Retrieved November 29, 2010 from http://en.wikipedia.org/wiki/Model-View-Controller

[2] Facebook Developers. 2010. 29 November 2010. <http://developers.facebook.com/>

[3] Harris, Matt. "Twitter API Wikis/FrontPage". Ed. Harris, Matt. 24 Aug. 2010. <http://apiwiki.twitter.com/w/page/22554648/FrontPage>

[4] Graph API. 2010. <http://developers.facebook.com/docs/api>

[5] Twitter API Documentation. 2010. <http://dev.twitter.com/doc>

[6] Crum, Chris. "Bing iPhone App Upgraded with Social Search Results. Facebook and Twitter Friends in Results". June 22, 2010. <http://www.webpronews.com/topnews/2010/06/22/bing-iphone-app-upgraded-with-social-search-results>

[7] Threadsy. 2010. <http://www.threadsy.com/>

[8] Facebook. 2010. <http://www.facebook.com/>

[9] Twitter. 2010. <http://www.twitter.com/>

[10] Reuters 2010. <http://www.reuters.com/article/idUSTRE52A6E420090311?feedType=RSS&feedName=internetNews>

[11] Marketing Charts 2009. <http://www.marketingcharts.com/direct/social-network-traffic-surpasses-web-based-emails-in-uk-2333/>

[12] Sync blog 2009. <http://www.sync-blog.com/sync/2010/07/is-email-on-its-way-out.html>

[13] Wikipedia 2010. <http://en.wikipedia.org/wiki/Social_network_service>

[14] Facebook 2010. <http://www.facebook.com/press/info.php?statistics>

[15] Techcrunch 2010. <http://techcrunch.com/2010/06/08/twitter-190-million-users/>

[16] Digital Network 2010.

<http://digitalgoesinternational.wordpress.com/2010/07/07/social-media-in-spain-72-of-the-youth-use-more-than-one-social-network/>