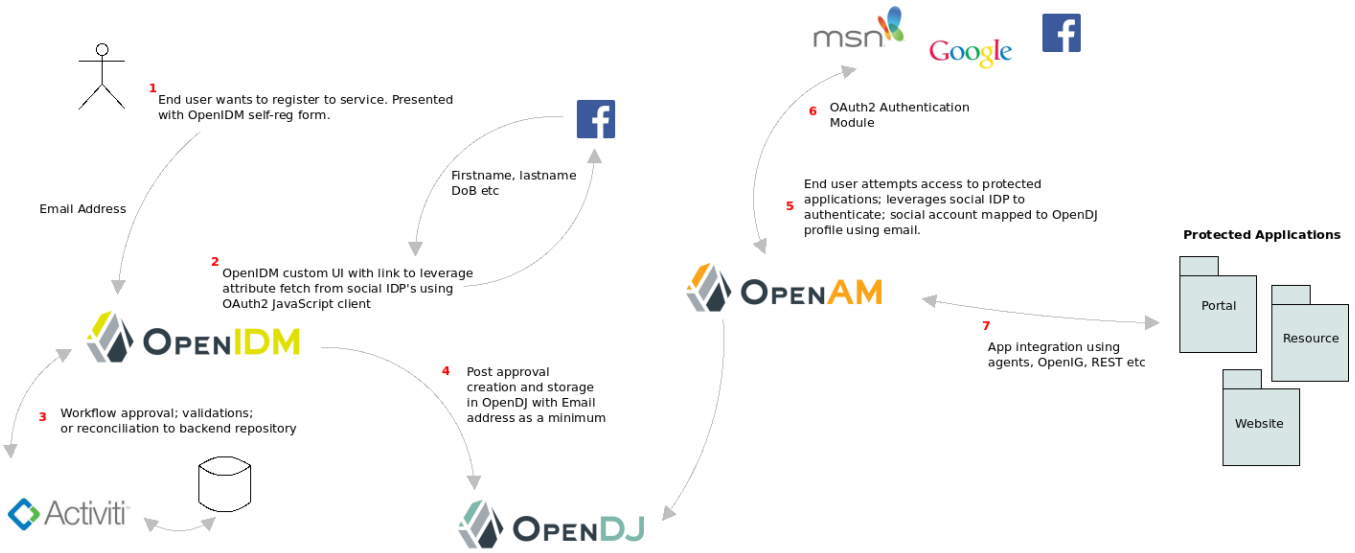
Self-Registration Harvesting Facebook Attributes

A common use case for C2C style applications and consumer facing services, is to allow the registration process to retrieve attributes from social media sites such as Facebook. This is a shortcut user convenience approach to providing data for other services.

In B2B style environments, registration may need other approval, reconciliation or workflow style processes to be triggered, which makes this an ideal use case for integrating with OpenIDM.



Note this example is based on OpenIDM 3.0, due to the improved capability of creating custom user interfaces. The pre-release nightly build is available at ForgeRock.org.

The above diagram is an example flow showing how self-registration in OpenIDM is modified to allow for the pre-population of attributes from Facebook, before provisioning an account in OpenDJ, perhaps after an internal reconciliation or approval processes. The OAuth2 Authentication Module is then used in OpenAM to allow for authentication from the social media source, by using an alias to map into the OpenDJ profile.

Creating a Custom UI for OpenIDM

This step is documented here in full - http://openidm.forgerock.org/doc/integrators-guide/index.html#ui-project-config

Basically copy the ../enduser directory from the ../openidm/ui directory in the OpenIDM deployment area, into a new folder. Edit the ../conf/boot/boot.properties to use the new UI folder location. This provides a nice, clean way to edit the UI with custom pages or styles. The only page we're going to edit is the self-registration page. You can either edit, or create a new page dependent on requirements.

Creating an OpenIDM Application in Facebook

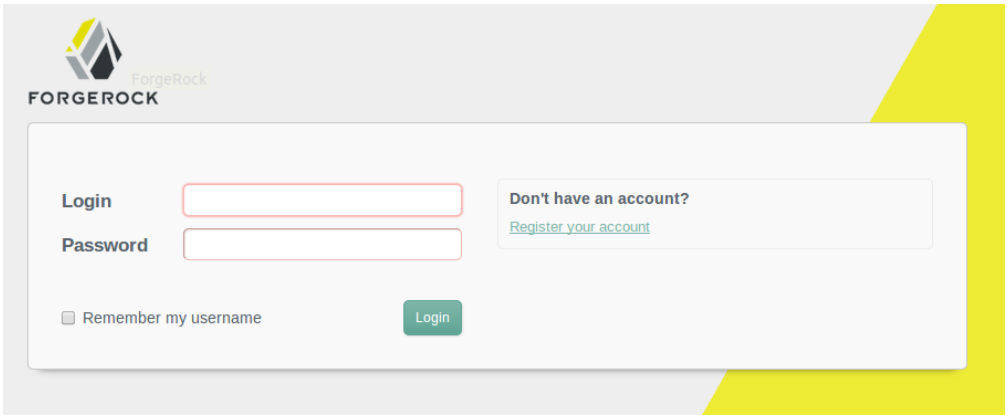
The Facebook development platform requires you to set up an application reference with unique ID and shared secret, to allow your OpenIDM self-reg app to communicate with Facebook and retrieve user attributes. Following the Facebook developer documentation to complete this - https://developers.facebook.com/docs/facebook-login/login-flow-for-web/v2.0

The app will basically need a name and domain associated with it. If you're developing on localhost, simple edit your ../etc/hosts file to give you machine a fully qualified domain. Eg openidm.exampledomain.com instead of just localhost.

Once the app is created, note down the appid - this will be needed to authenticate the OpenIDM self-reg page with Facebook in the next steps...

Enabling Self-Service Registration in OpenIDM

By default, self-service is disabled. To enable, simply edit the ../openidm/conf/ui-configuration.json as per http://openidm.forgerock.org/doc/integrators-guide/index.html#ui-self-registration. Basically change the selfRegistration field from false to true. This is a dynamic change, so simply refresh the OpenIDM login page to see the new registration link.



Adding Facebook Integration to the Self-Service Registration Page

We need to add in some JavaScript libraries and functions to allow communication to the Facebook API. This process is fully documented here at the Facebook developers page - https://developers.facebook.com/docs/facebook-login/login-flow-for-web/v2.0

I created my custom UI in a new directory called myui under ../openidm. The self-service registration file is a file called UserRegistrationTemplate.html.

../openidm/myui/enduser/public/templates/user/UserRegistrationTemplate.html

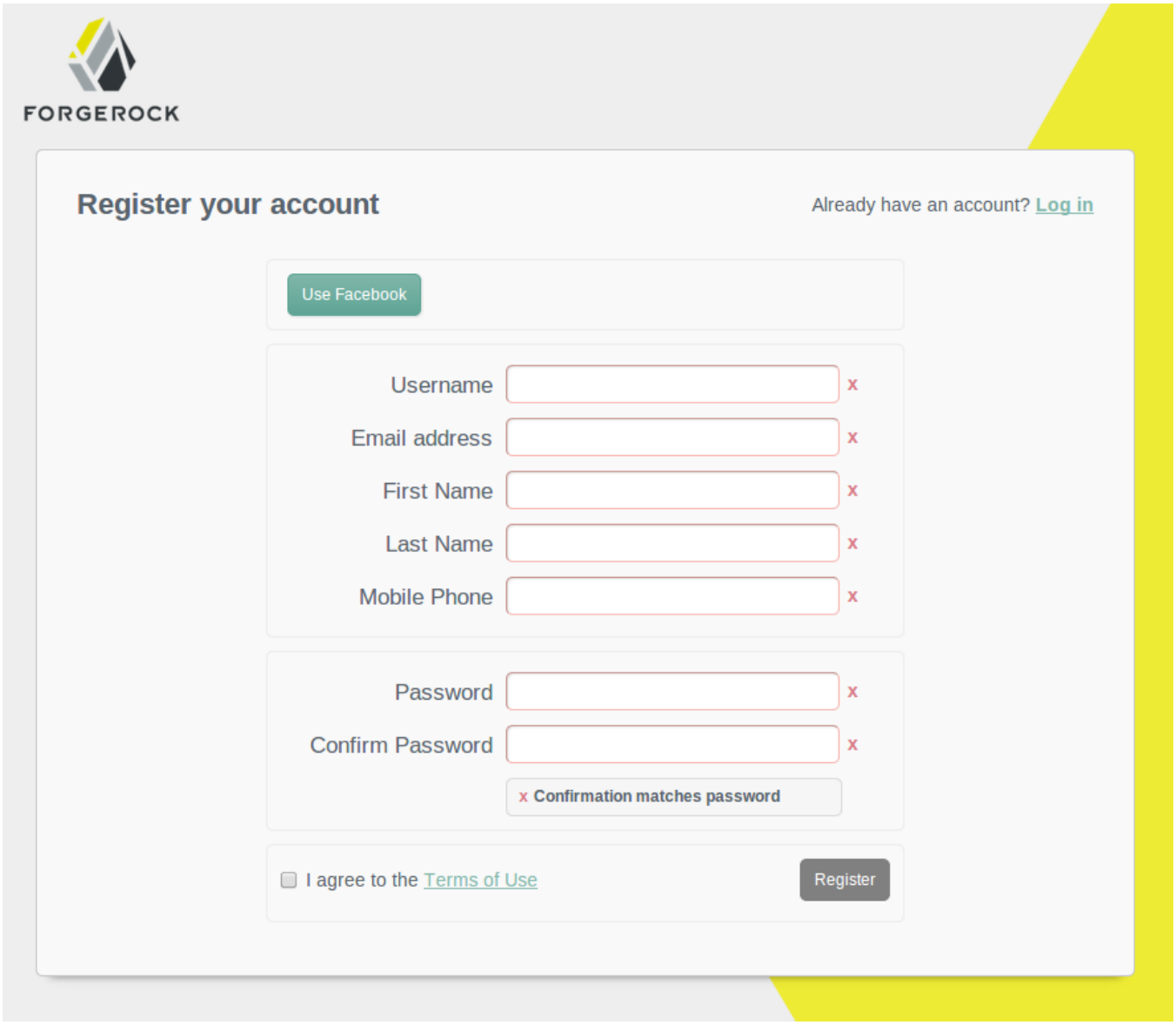
Open this file in your editor and we can add in the necessary JavaScript changes. Follow the detailed Facebook developers notes, but basically add in a section to initialise the Facebook libs in your page:

```
<!--facebook integration smof may30-->
<script>
  window.fbAsyncInit = function() {
    FB.init({
      appId      : '###############',
      xfbml      : true,
      cookie     : true,
      version    : 'v2.0'
    });
  };

  (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 = "//connect.facebook.net/en_US/sdk.js";
        fjs.parentNode.insertBefore(js, fjs);
      }(document, 'script', 'facebook-jssdk'));

</script>
<!--facebook integration smof may30-->
```

Add in your specific Facebook app id. This async load, then gives your page access to the all FB prefixed Facebook functions.

I added a basic button on the self-registration form, that basically starts the Facebook login check process and populates the form fields with some attributes. This is just a copy and paste from an existing button on the page and calls the useFacebook(); function when clicked.

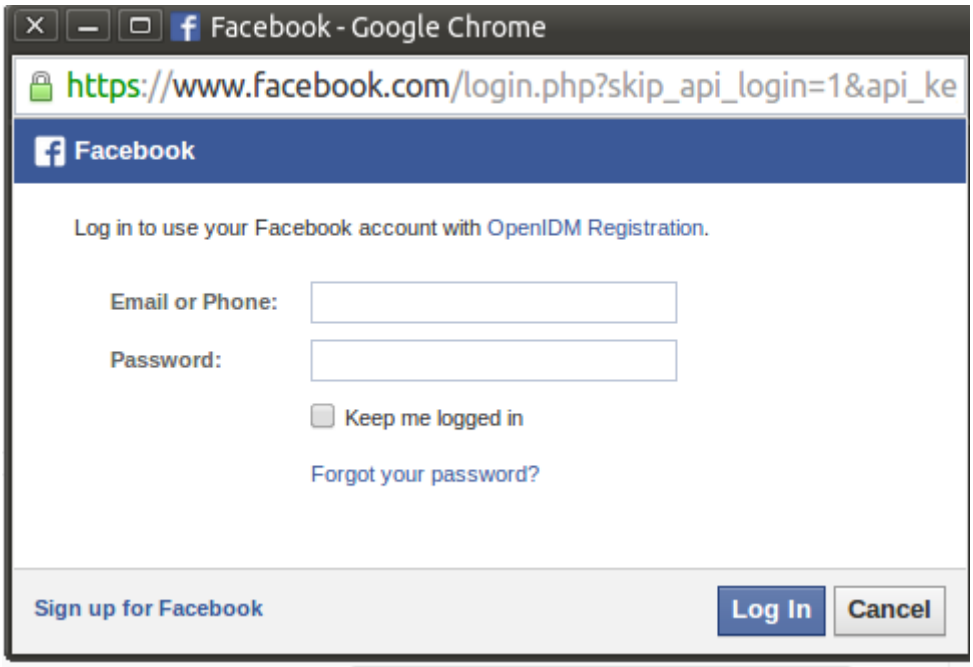
```
<fieldset class="fieldset col0">

    <input type="submit" name="fb_retrieval" class="button" value="Use Facebook" onclick="useFacebook();"/>

</fieldset>
```



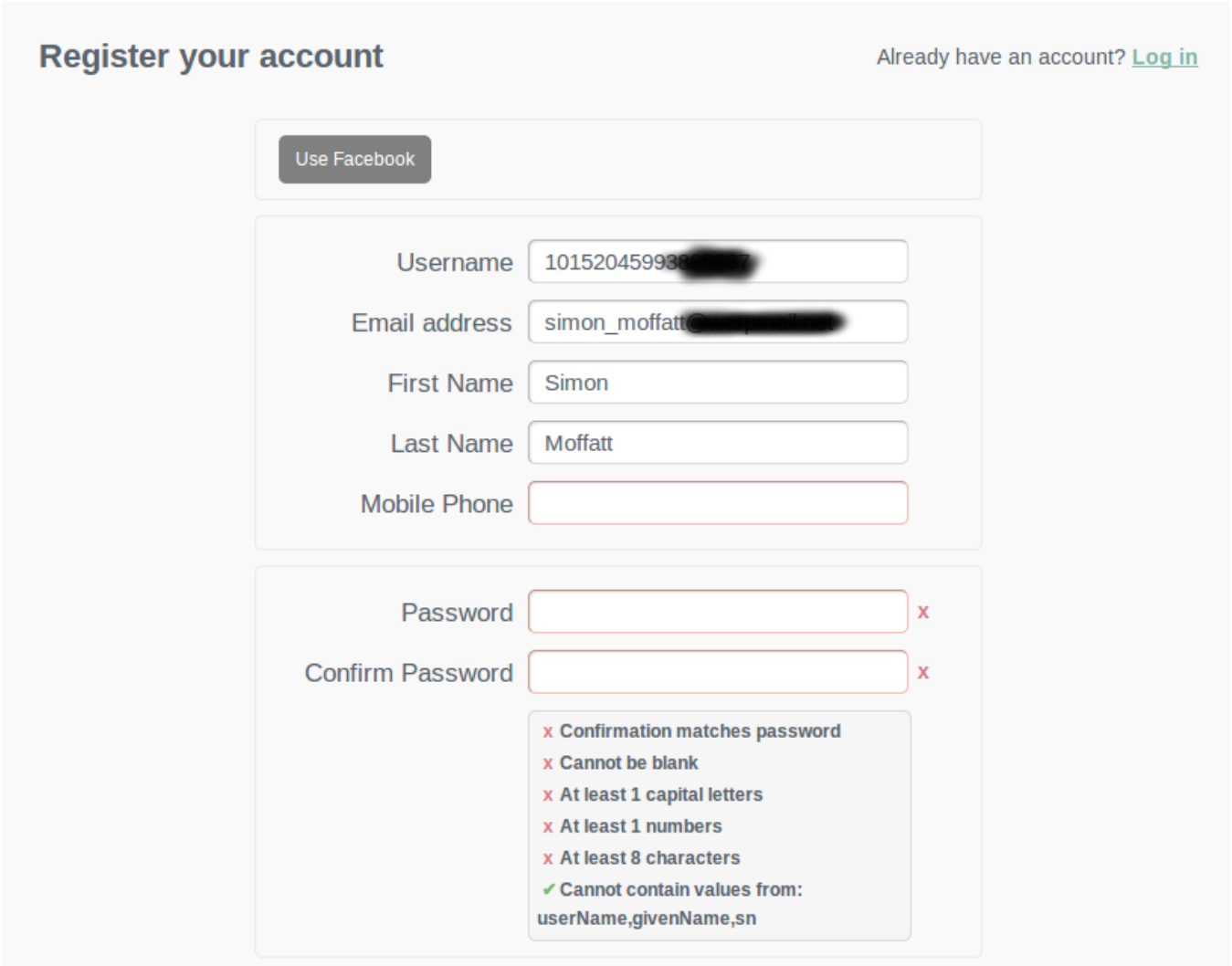
Clicking the Use Facebook button, calls the useFacebook() function, that simply calls the Facebook login function to do a few things - force you to login into Facebook if you're not already logged, and then retrieve 4 attributes. 3 are from the Facebook public profile and a fourth, email, as an extended request. The function then does a basic value replacement on the existing OpenIDM form with the values received from Facebook. I mapped the OpenIDM username to Facebook ID, but you could simply just map the email address for convenience, if your users want to log into OpenIDM and not just be provisioned elsewhere.

```
<script>

//retrieves facebook attributes
function useFacebook(){

    FB.login(function(response) {

       if (response.authResponse) {

               console.log('Welcome!  Fetching your information.... ');
               FB.api('/me', function(response) {

                           document.getElementById("userName").value = response.id;
                           document.getElementById("mail").value = response.email;
                           document.getElementById("givenName").value = response.first_name;
                           /document.getElementById("sn").value = response.last_name;

               });

       } else {

         console.log('Facebook: User cancelled login or did not fully authorize.');
         document.getElementById("fb_error").value = 'Facebook: User cancelled login or did not fully authorize.';

       }
     }, {scope: 'email'});

};


</script>
```

When clicking the Use Facebook link, a dialogue will appear, either asking you to Login to Facebook if you're not already, or to give consent to the OpenIDM registration app, if you are already logged in.



Once approved or logged in, the form updates with the necessary attributes:



I simply mapped the firstname, lastname, email and username, but depending on your form requirements other attributes could be retrieved.

Note at this stage, no password is pulled our or mobile number. Note also, that the details coming from Facebook, may not be good enough to pass the data validation checks done on the form. The password fields would in reality be hidden if using Facebook OAuth2 authentication in OpenAM later in the flow as they would not be required.

The above flow is really for capturing data from other IDP's to populate ether OpenIDM or other downstream systems. In the above example, once the account had been approved, OpenIDM would provision say the email address, firstname and lastname into OpenDJ. The OAuth2 authentication module could simply then map the Facebook.mail to the email in OpenDJ as alias when performing authentication.