Implementation Guide

Connect your application to Salesforce with first-party-data

Background

First party data (FPD) is a simple but powerful set of information you can use to build a customer profile across different systems.

FPD is information that you can obtain from a user's session, once they've consented for you to know who they are.

This can include personal information like their email, and, with time, can play a part in creating experiences that are personal to the consumer, depending on the software.

CIAM solutions like Okta CIC Help maintain a unified customer profile by creating a central resource of users across your upstream and downstream applications, like customer relationship management (CRM), and even customer data platforms (CDPs).

Actions makes it easy to add integrations, and the following Actions template covers how to share the basic, necessary information to create a lead in Salesforce from a single user login.

As part of our extensibility framework, Actions are a drag-and-drop pro-code/no-code logic that you can customize for your own applications and integrations that start with Identity.

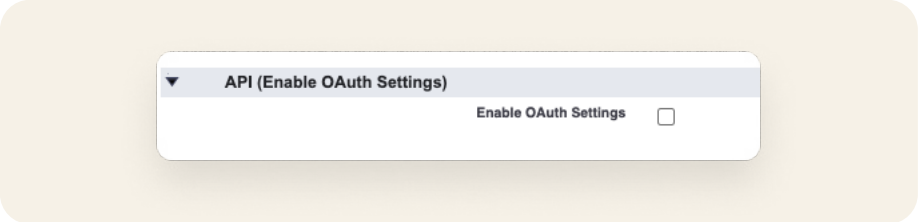
Actions lets you add code to vital points in the authentication pipeline with just javascript — and 2M+ npm modules at your disposal.

Actions templates teach you how to harness the power of Actions, and get to market faster than the competition, addressing common use cases that are vital for organizations today.

Setup in Salesforce

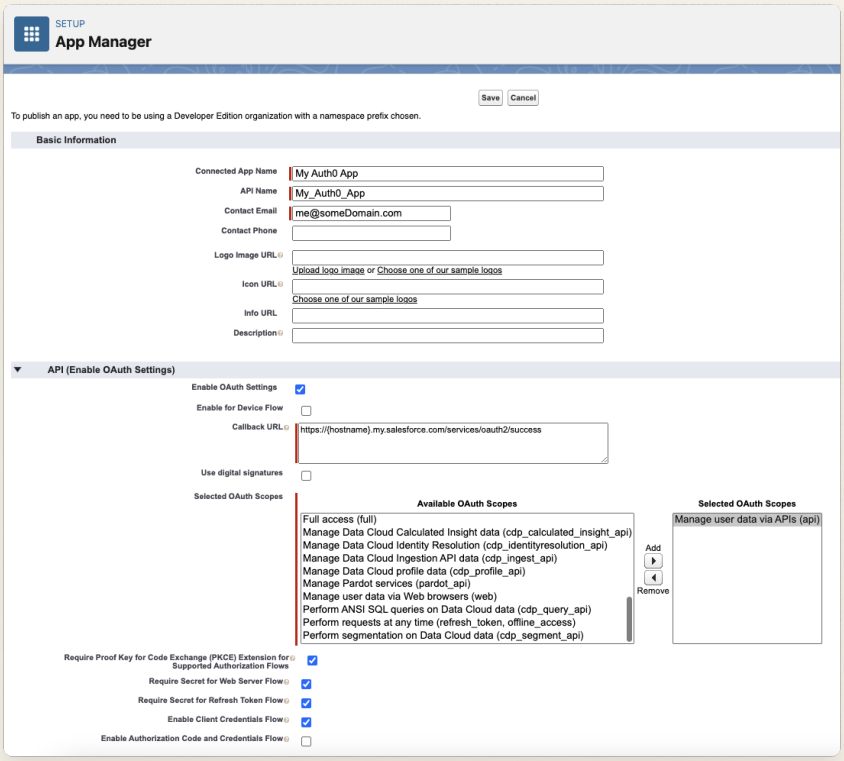
If you haven't already done so, be sure to create an account with Salesforce; it's recommended to create a developer account.

1. Navigate to **Setup**, and go to **App Manager**.
2. Select **Create New Connected App**, enter your app name and contact information.
3. To configure our connected app, select **Enable OAuth Settings**:



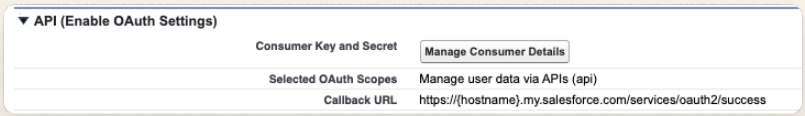
4. For your callback URL, be sure to use your My Domain URL or sandbox URL for your OAuth endpoint in testing, i.e. https://{hostname}.my.salesforce.com/services/oauth2/success. This will add specificity to your request.
5. It's best practice to limit the scope for what your app can do, so, for this use case, select **manage user data via APIs** from the options available.
6. Since we are using a Client Credentials grant, be sure to select enable **Client Credentials Flow**, and accept the security dialog.

Your configuration should look something like this:



7. Hit **save**, and then **continue** on the next screen.

8. Next, select **Manage Consumer Details** to generate a client ID and secret that can be used in your Action to connect with your Salesforce app.



9. You will need to enter a quick verification code in order to generate this ID and secret. Save this information for the next steps in this guide.

The Template: Call Salesforce API to record the contact as a new lead (post-login)

This template creates a lead in Salesforce when a user has logged in, if they haven't been added already.

Note: this will also create a lead for new users, since users are logged in for the first time post-sign up.

This template includes basic information to gather and send to Salesforce on a one-time basis, and the adds a line of metadata in the Auth0 user profile to confirm the lead is recorded `(recordedAsLead: true)`:

```
exports.onExecutePostLogin = async (event, api) => {
// if a lead has already been recorded then end the action
successfully
if (event.user.app_metadata.recordedAsLead) {
return;
}
//Populate the variables below with appropriate values, failing if
any secrets are missing
const sfDomain = event.secrets.SALESFORCE_DOMAIN;
if (!sfDomain) {
console.log(`Unable to create lead: Salesforce domain not
configured`);
return;
}
const sfClientId = event.secrets.SALESFORCE_CLIENT_ID;
if (!sfClientId) {
console.log(
`Unable to create lead: Salesforce client id not configured`
);
return;
}
const sfClientSecret = event.secrets.SALESFORCE_CLIENT_SECRET;
if (!sfClientSecret) {
console.log(
`Unable to create lead: Salesforce client secret not configured`
);
return;
}
const sfCompany = event.secrets.SALESFORCE_COMPANY;
if (!sfCompany) {
console.log(`Unable to create lead: Salesforce company not
configured`);
return;
}

// fetch the token from the cache or regenreate it if it cannot be
retrieved, see
// https://help.salesforce.com/s/articleView?id=sf.remoteaccess_oauth_
endpoints.htm
```

```
const fetchAccessToken = async () => {
const cachedToken = api.cache.get('sf_access_token');
if (cachedToken) {
return cachedToken.value;
}
const sfLogin = `https://${sfDomain}/services/oauth2/token`;
const body = new FormData();
body.set('grant_type', 'client_credentials');
body.set('client_id', sfClientId);
body.set('client_secret', sfClientSecret);
// force hourly refresh on a per-host basis
const expiry = Date.now() + 3600000;
const response = await fetch(sfLogin, {
method: 'POST',
body: body,
});
if (!response.ok) {
throw new Error('Unable to fetch token');
}
const data = await response.json();
api.cache.set('sf_access_token', data.access_token, {
expires_at: expiry,
});
return data.access_token;
};

//See http://www.salesforce.com/us/developer/docs/api/Content/sforce_
api_objects_lead.htm
const createLead = async (access_token) => {
// see https://developer.salesforce.com/docs/atlas.en-us.object_
reference.meta/object_reference/sforce_api_objects_lead.htm
const body = {
LastName: event.user.name || event.user.email,
Company: sfCompany,
Email: event.user.email,
};
const headers = {
'Content-Type': 'application/json',
Authorization: `Bearer ${access_token}`,
};
const sfLead = `https://${sfDomain}/services/data/v59.0/sobjects/Lead`;
const response = await fetch(sfLead, {
method: 'POST',
body: JSON.stringify(body),
headers: headers,
});
if (!response.ok) {
throw new Error('Unable to create lead');
}
};
```

```
try {
const token = await fetchAccessToken();
await createLead(token);

// if no errors were raised, the lead was created
api.user.setAppMetadata('recordedAsLead', true);
} catch (error) {
// fail gracefully *withough* recording that lead generation has
// completed. Failing to handle these errors here would result
// in a failedl login flow, which we do not want.
return;
}
};
```

Setup

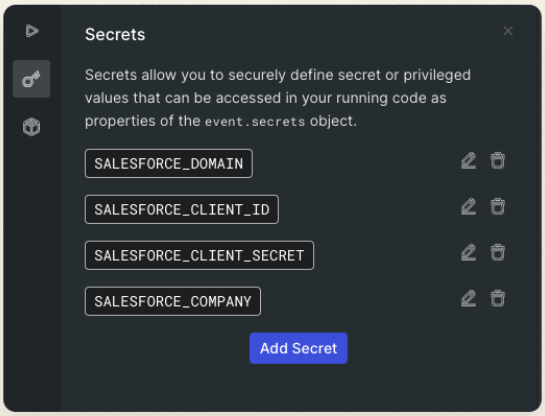
You will want to create a series of event secrets that are referenced in the beginning of this action, including:

SALESFORCE_DOMAIN: {myDomain}.my.salesforce.com

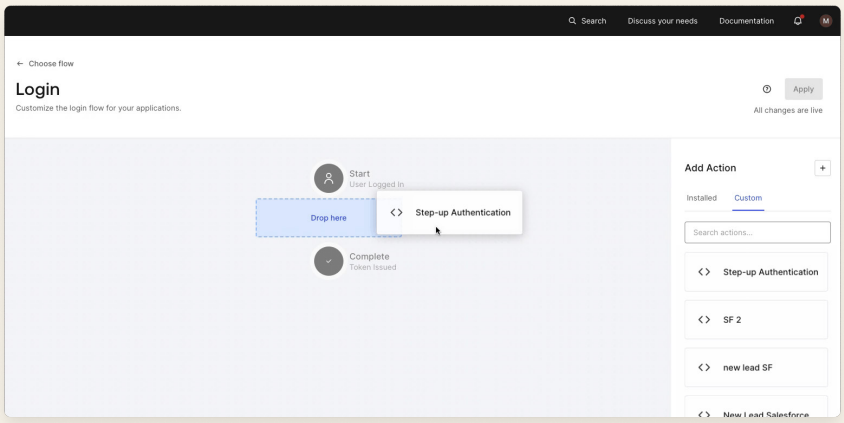
SALESFORCE_CLIENT_ID: your consumer key

SALESFORCE_CLIENT_SECRET: your consumer secret

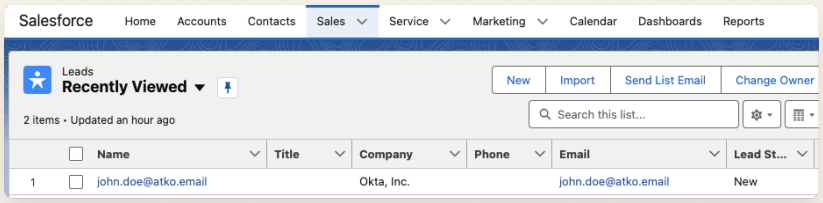
SALESFORCE_COMPANY: your organization's name



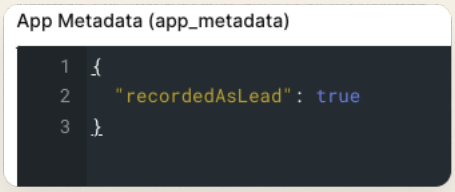
When you've tested and made sure all your values are correct, feel free to deploy and drag-and-drop your action into your **Login** flow, and hit **Apply**:



The next time a user next logs in or signs up, a new lead will be created in Salesforce:



And, you will see the corresponding metadata confirmed in Auth0:



Summary

Actions can support any number of sales and marketing integration use cases.

Our Actions templates help illustrate how quick and easy it is to hook into any system, with just a few clicks.

All of our templates are production-ready and make it easy to come up with your own tailored solutions.

Thought that was easy?

Check out our other implementation guides for more vital business operations made simple with Actions templates:

- Adaptive MFA
- Flag sensitive transactions for step-up
- Prompt with personalized recommendations
- Customize UI for accessibility