Novus APIs and Web Admin

Personify

June 2024

Table of Contents

What are Novus APIs?

The Novus APIs are a set of APIs, each installed as a separate web application, allowing greater flexibility to users for retrieving and managing their data.

There are two types of APIs: Entity APIs and Query APIs. The Entity API is a transactional service, which can be used to create and update records, and the Query API is a read-only service for querying the data. Both types of APIs are built on the standard HTTP request methods: GET, POST, PUT, and DELETE.

Entity APIs

The Entity API is a transactional service, mostly used for creating and updating records. These APIs are based on a domain model defined in the Personify Business Logic Layer. For example, an individual customer is a domain, and for that individual customer you can retrieve address information, communication methods, etc. as part of that domain. This service is also simplified for complex transactions.

Query APIs

The Query API is based on REST with OData operators as querying options. The Query APIs, like the Entity APIs, are also based on a domain model defined in the Personify Business Logic Layer. The Query APIs includes features, such as pagination, aggregation, and other complex data functions, allowing the ability for greater data manipulation and granularity in data fetches. With the Query API it is easy to fetch multiple data elements in one call. For example, if you want to query a customer's record, including their orders, their phone numbers, and their addresses, etc., you can fetch all that information in a single call.

Getting Started

The Novus APIs enables you to have open access to your data when you need it and control who has access to it. The platform provides tools and resources for the APIs that easily allows you to integrate with third party vendors and fetch data as needed.

This section helps you get acquainted with the Web Admin platform and explains how to get access and use the Novus APIs.

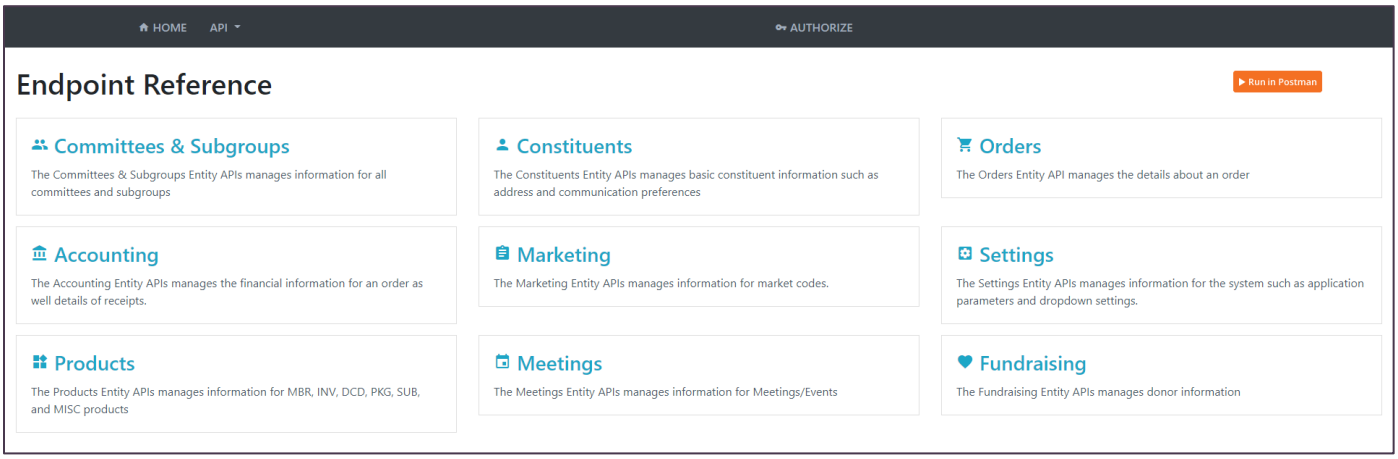
How the platform is organized

The Web Admin is a platform that contains the technical workflows needed by developers and administrators such as the ability to customize the Web Client or manage who has access to the Novus APIs. Within the Web Admin, you can find different areas to manage. The Access Control contains everything needed to secure the APIs tied to an environment. The Web Designer contains workflows needed to customize the Web Client. To see more information about these areas, review the Access Control and Web Designer sections.

© 2021 Personify Inc.

The Novus APIs have been organized into different scopes, or areas, that are commonly used together. These are similar to the various functionality found in the modules of the Web Client, such as Constituents, Orders, and Accounting. For more information about Scopes, review the Scopes section.

All the endpoints are available via a Swagger page, which is a software that has been utilized to provide more information about the APIs. Below is the Home page of the Swagger interface of the Entity API. The API drop-down at the top of the page allows you to navigate to the Query APIs, the interface for Query Services, and the OAuth 2.0 Authorization Requests.

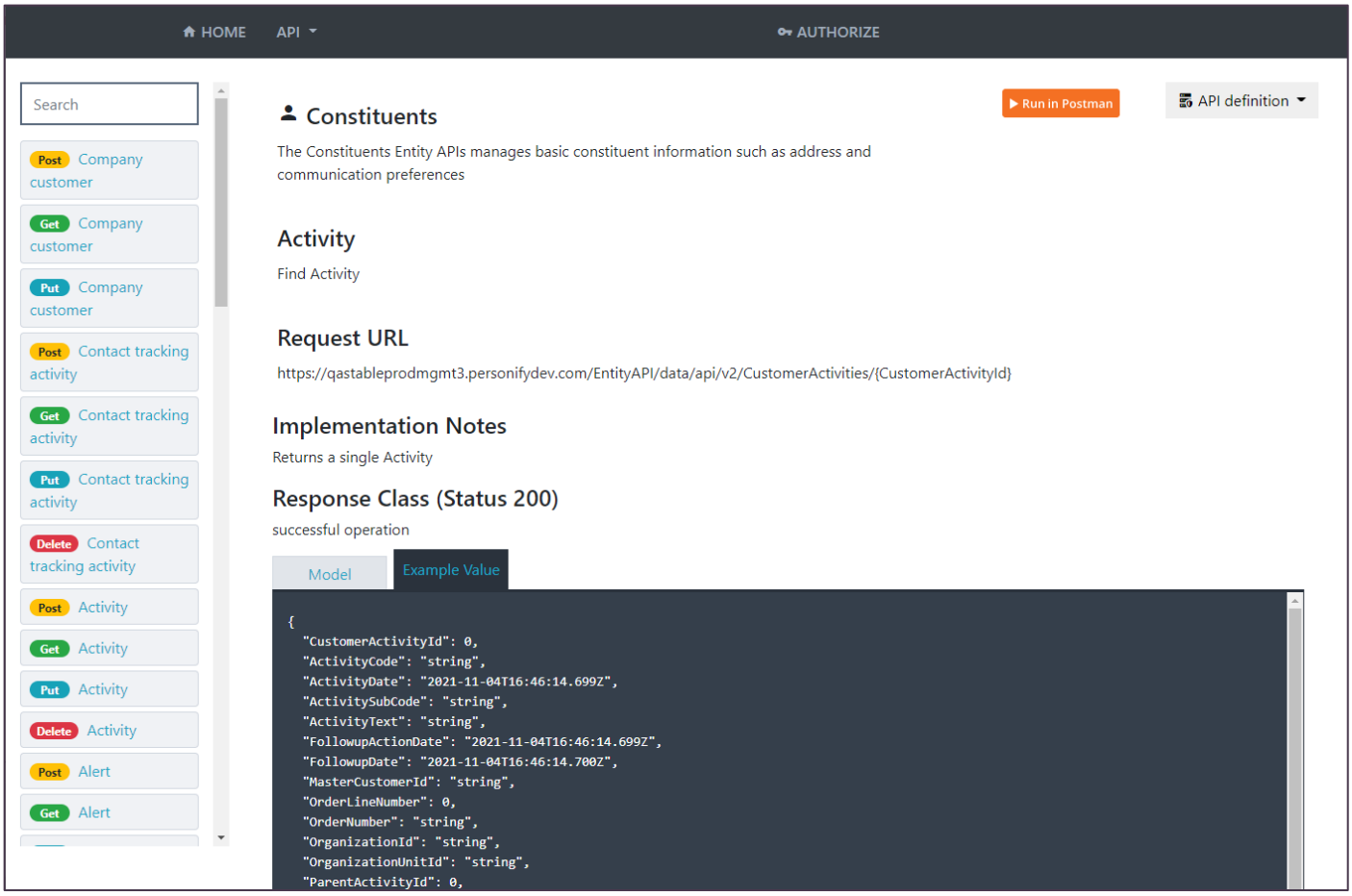


From the Home page, you also have the option to Authorize in order to test the APIs from the Swagger page. Authorization is based on Client ID and Client Secret which can be obtained when a new client is created. Additional information on Authorize and OAuth 2.0 Authorization Requests is available in the Novus API Security section.

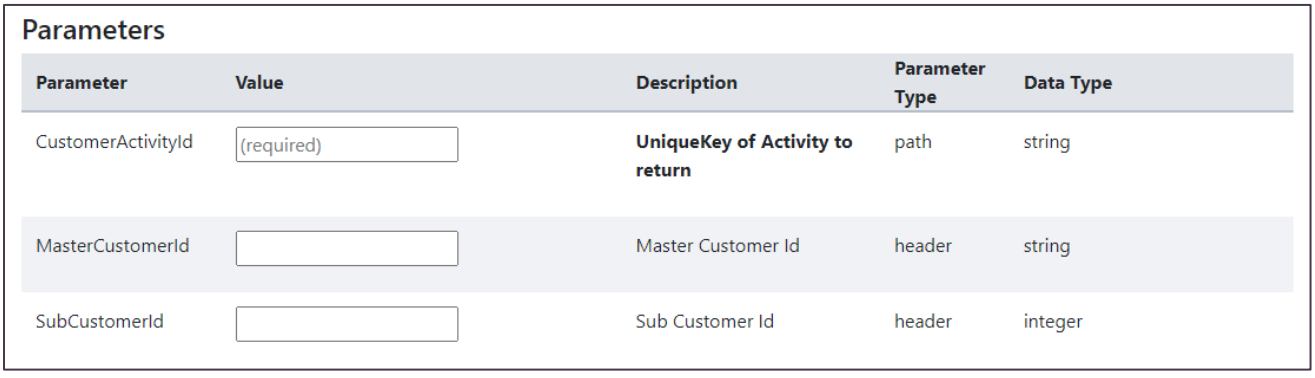
The boxes listed on the page, such as Constituents and Orders, are scopes that have been defined. You can use the scopes to limit access to the APIs. Scopes can be edited and adjusted as needed per client. Clicking on a scope will display a list of endpoints that are available within the scope. The endpoint list is configurable by client so that certain data points can be exposed or hidden for a defined scope. For more information on managing scopes in the Web Admin Portal, please see Managing Scopes.

A Postman collection is also available for download. Clicking on the "Run in Postman" button will prompt you to select either the Web or Desktop application for Postman. This collection contains examples of how to manage and fetch data using the Entity and Query APIs. Before utilizing this collection, follow the How to Grant Access section to generate a client ID and client secret, which is needed for authorization.

© 2021 Personify Inc.



You can select an endpoint to view additional details to the right of the endpoint list. In the Parameters section, you can see the list of parameters and available options within those parameters. All access to the Entity API are by keys that are granted to clients.



Parameters

Parameter	Value	Description	Parameter Type	Data Type
CustomerActivityId	(required)	**UniqueKey of Activity to return**	path	string
MasterCustomerId		Master Customer Id	header	string
SubCustomerId		Sub Customer Id	header	integer

Endpoints within the Query API allow many more options to fetch data than the Entity API. The different parameters are how the data can be filtered if needed. The Postman collection contains a few examples of how these parameters can be used.

© 2021 Personify Inc.

Parameters

Parameter	Value	Description	Parameter Type	Data Type
$select	Nickname TributeFlag MasterCustomerId	selects child properties that need to be considered added in result	query	Array[string]
$expand	ConstituentRoles Coupons CusMatchingGiftPlanInfo CustomerAuditDefaultInfo	expand child collection properties	query	Array[string]
$filter		filter based on child properties	query	string
$orderby	Nickname TributeFlag MasterCustomerId	order by child properties in result	query	Array[string]
$top		number of records to be returned in single page	query	integer
$skip		number of records to be skipped before returning a single page	query	integer
$count		total count of filtered record	query	boolean

How to grant access

Below is the recommended workflow for accessing the Novus APIs. Visit the Access Control section for details on how to do each of these steps in depth.

1. Review the available scopes and add/update as needed.
   a. The Scopes are how the endpoints are grouped together and will be used to determine who has access to what groups of endpoints. These will come predefined, but you can adjust these as needed for your business needs.
2. Create a role(s).
   b. Roles are used to control who has access to what scopes.
3. Create a client
   c. A client, which is assigned a client ID and client secret, is how a user or vendor accesses the APIs.
4. Once you have a client ID and client secret, you can now access the data within the specified environment. You can input these on the Authorize screen in the Swagger page or use them in the Postman collection to begin testing the APIs.

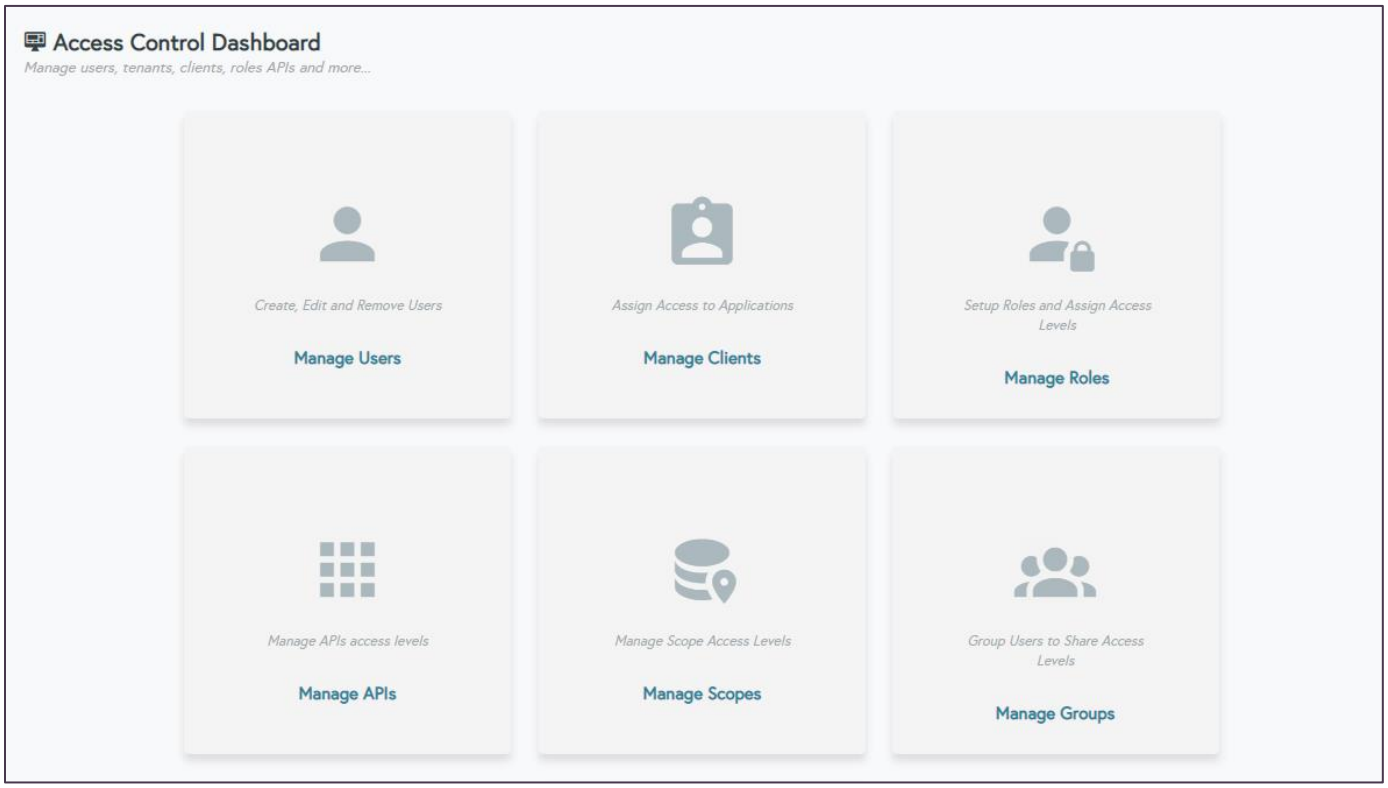
© 2021 Personify Inc.

Access Control

The Web Admin Portal allows clients to provide Novus APIs for vendor access using the Access Control. Each client has access to their own Web Admin Portal.

Access Control Dashboard

The Access Control Dashboard is the landing page for all administrative settings for Web Admin access. After selecting **Access Control Dashboard** from the Web Admin Dashboard, the Access Control Dashboard page displays, as shown below.



From this dashboard, you can navigate to the following areas of the Access Control setup:

- Manage Roles
- Manage APIs
- Manage Scopes
- Manage Clients
- Manage Users
- Manage Groups

© 2021 Personify Inc.

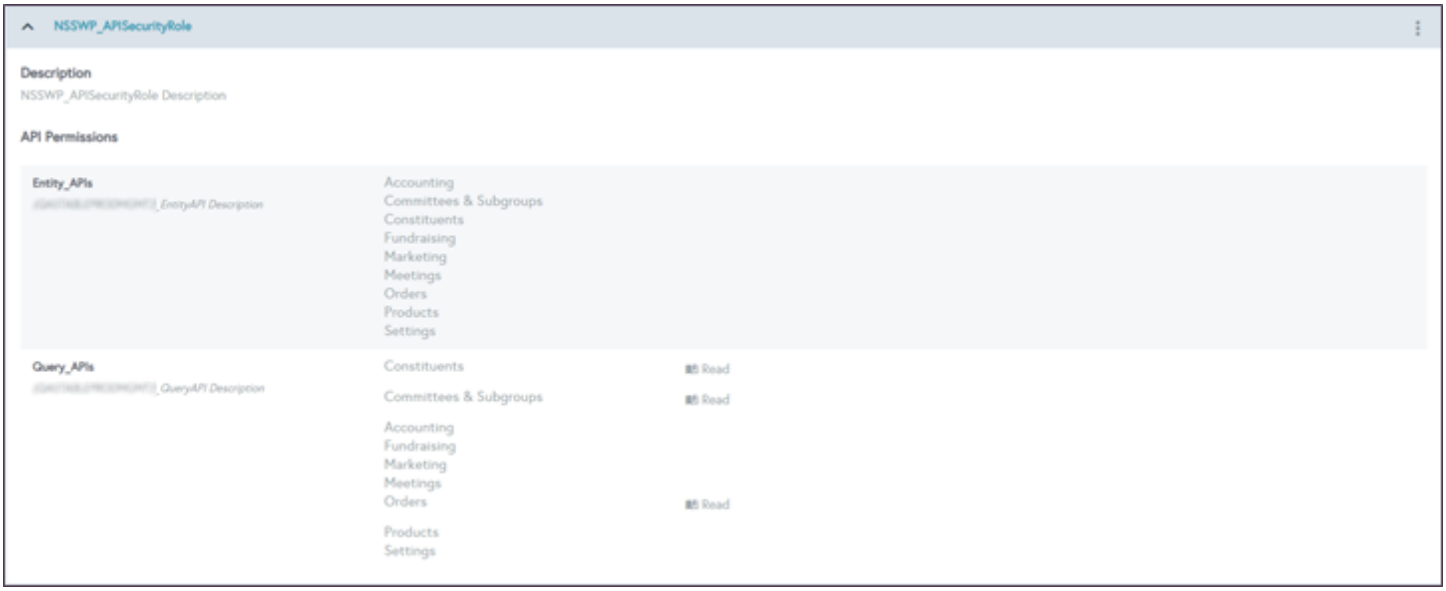
Managing Roles

A scope describes a single permission to a set of APIs. A role is a set of multiple scopes that can assigned to a user.

Roles are how permissions are assigned to the scopes. Roles are not part of the OAuth request, but they are tied to the Client ID and Client Secret. A role can be assigned to more than one client and all permissions of that role will be applied to the client. For more information on clients, please see the Managing Clients section.

NOTE: Roles are used for API and Scope security and Groups are used for Application security (like Web Designer).

The Roles page can be accessed from the Manage Roles tile on the Access Control Dashboard. The grid on the Roles page displays all the roles for the user's specific environment, which is displayed on the top navigation bar. From the grid, each role can be expanded to view additional details about the role, including the Description and API Permissions for the role.



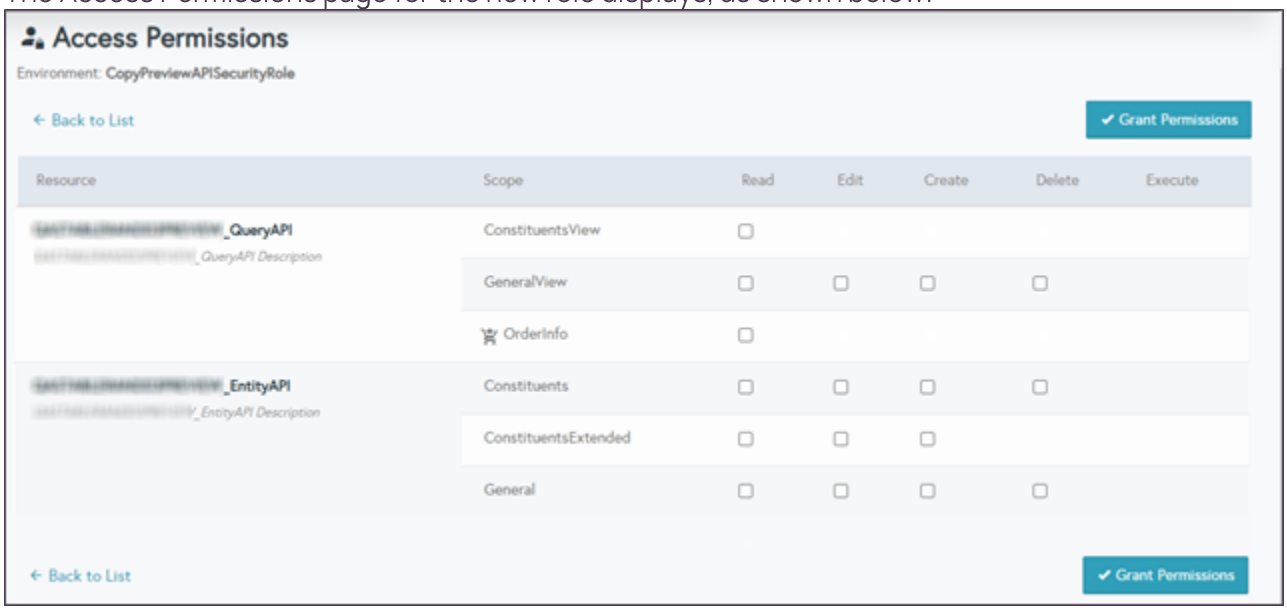
© 2021 Personify Inc.

To create a new role:

1. From the Roles page, select the **+Create New Role** button.

2. The Create a Client Role page displays, as shown below.

3. Enter the role **Name**.
4. Enter the role **Description**.
5. From the **Clone Role** field, select a role you wish to copy permissions from, if necessary.
6. Once you have completed entering the new role's information, select the **Create** button to create the new role.
7. The Access Permissions page for the new role displays, as shown below.



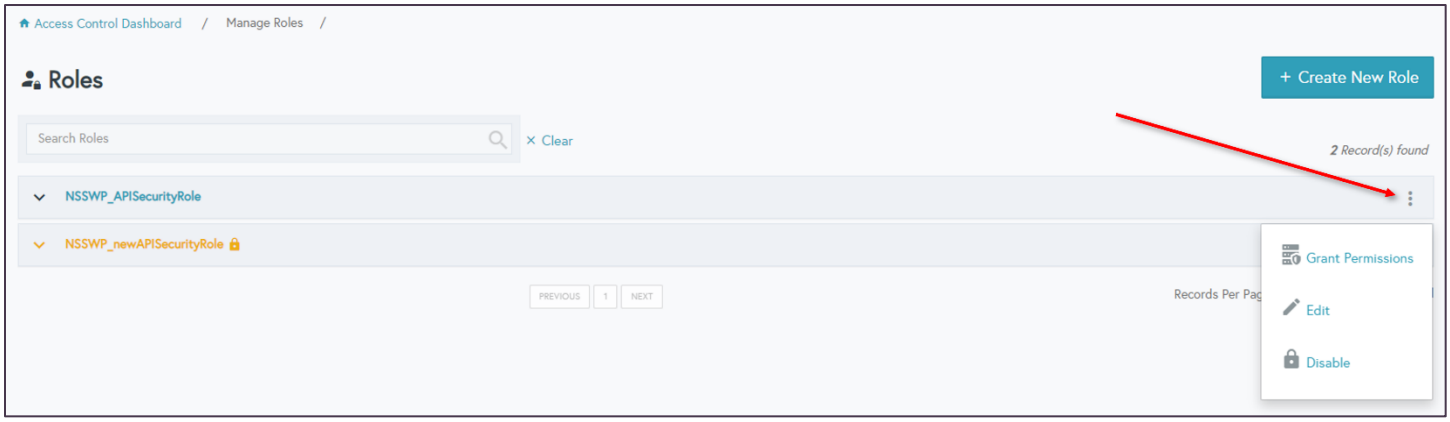
8. Select which Resources and Scopes the new role should have access to. You can select from **Read**, **Edit**, **Create**, and **Delete** access for each Scope by selecting the corresponding checkbox.

© 2021 Personify Inc.

9. Once you have selected which permissions this role should have access to, select the **Grant Permissions** button.

After creating a role, you can click the Edit pencil to edit the role details. Organization administrators can also create roles as necessary from the Roles page.

You can also edit the role information or grant permissions from the contextual menu. This will open the Access Permissions page so adjustments can be made.

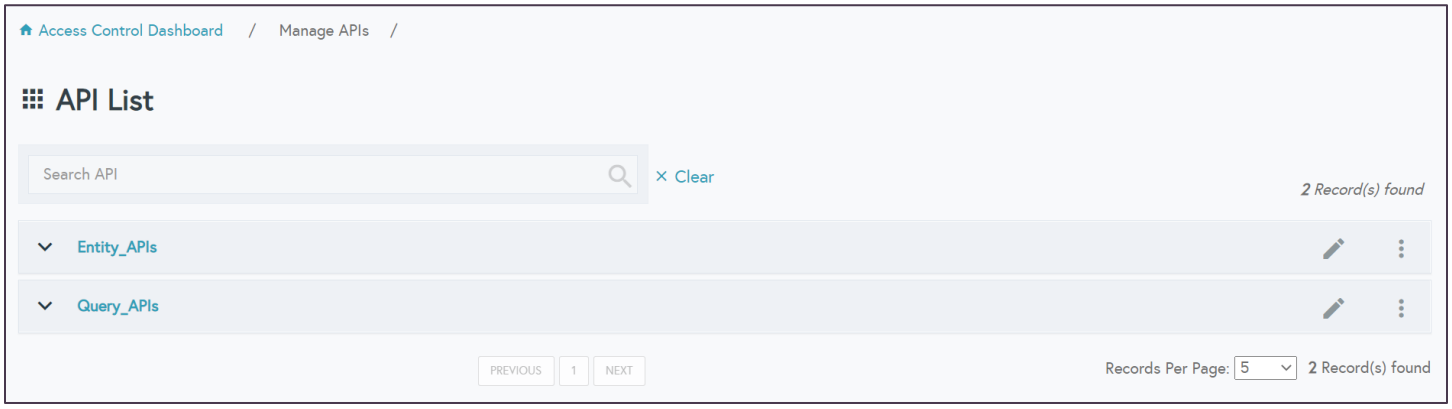


Roles can be deleted but must be disabled first. On the contextual menu of the Role, click on “Disable”. Once the Role has successfully been disabled, click on the contextual menu again to see the “Delete” option.

© 2021 Personify Inc.

Managing APIs

The API List page allows users the ability to view pre-defined APIs, as well as the scopes that they can manage under each API. To learn more about managing scopes, please see the Managing Scopes section. There are two types of APIs that are defined, the Entity API and the Query API.

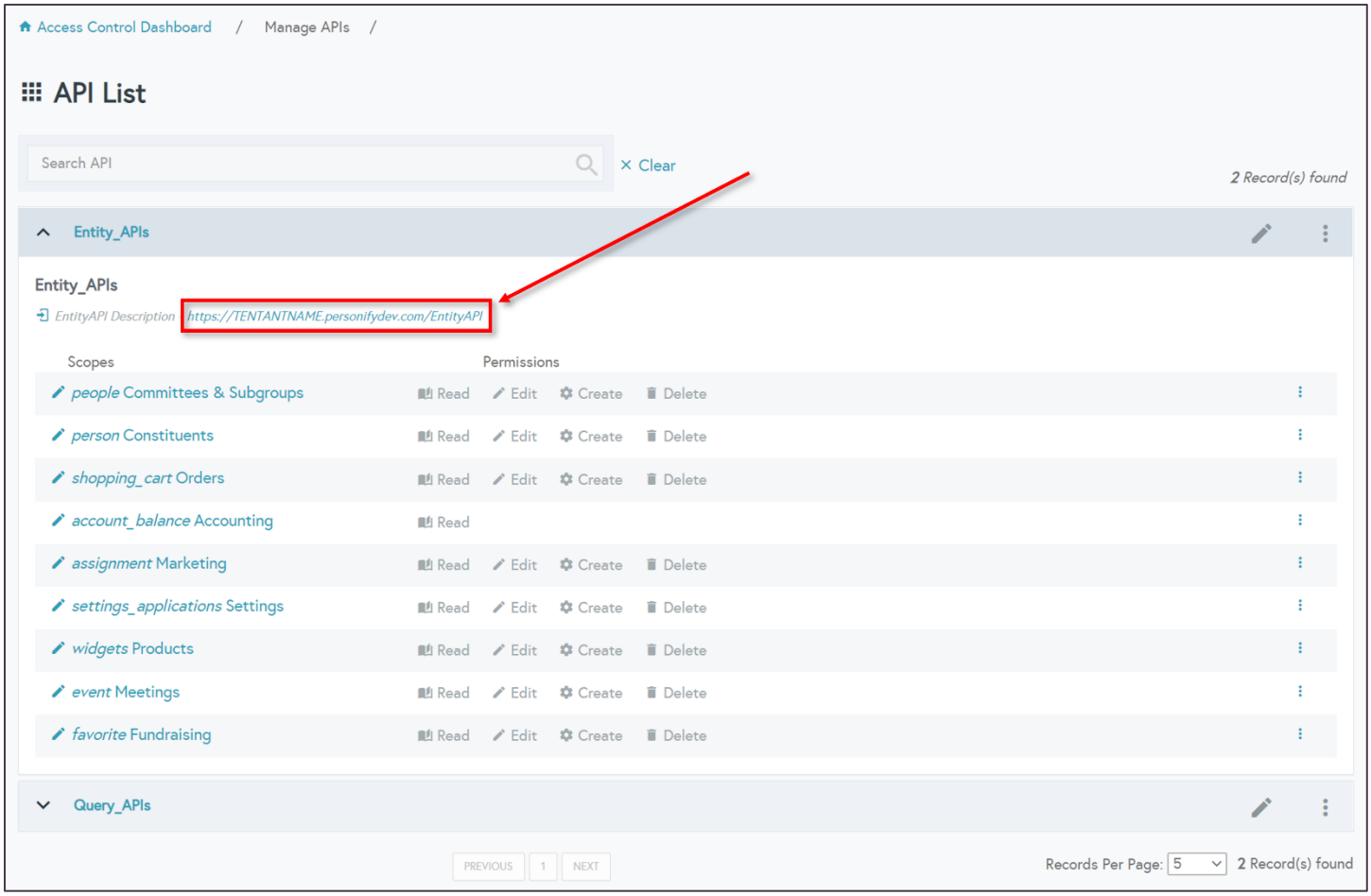


Entity APIs allow users to easily perform the standard CRUD operations (create, retrieve, update, delete). These have been built on the REST framework and use the standard HTTP methods (GET, POST, PUT, and DELETE). These APIs are based on a Domain model so that users can streamline their calls when managing data. For more information on Entity APIs, please see the Entity APIs section.

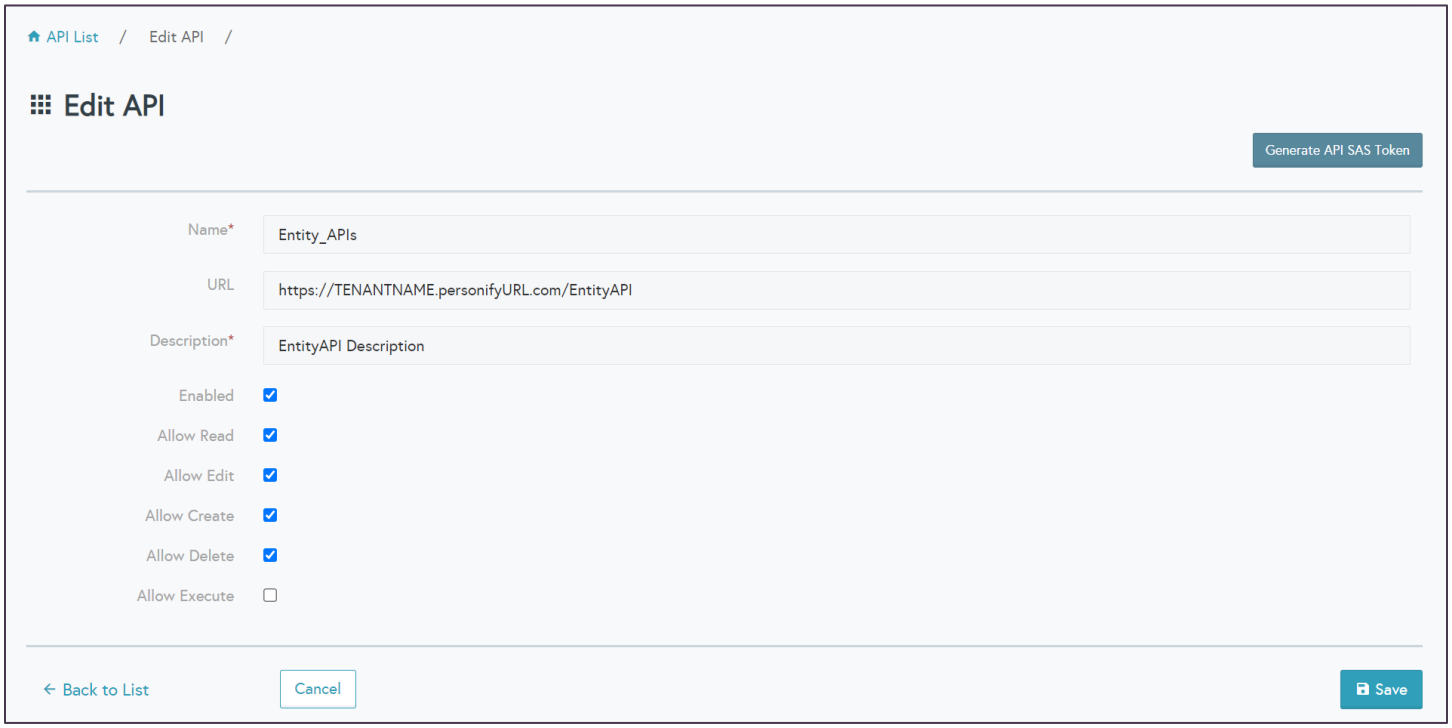
Query APIs allow users to get information quickly and fetch multiple data elements in a single call. These APIs are read-only services that are built on the REST framework and OData services. The OData foundation allows users to construct more complex queries such as pagination, aggregation, and filters. For more information on Query APIs, please see the Query APIs section.

Within the APIs is a link to the Swagger API page. Only endpoints that have the **Standalone** toggle set to No will be available, which can be reviewed in the API section of the Web Designer. The swagger page displays additional information about the endpoints. The swagger interface for the APIs is reviewed in the How the platform is organized section.

© 2021 Personify Inc.



Select the edit pencil to open the Edit API page, where a user can set the permissions for an API.

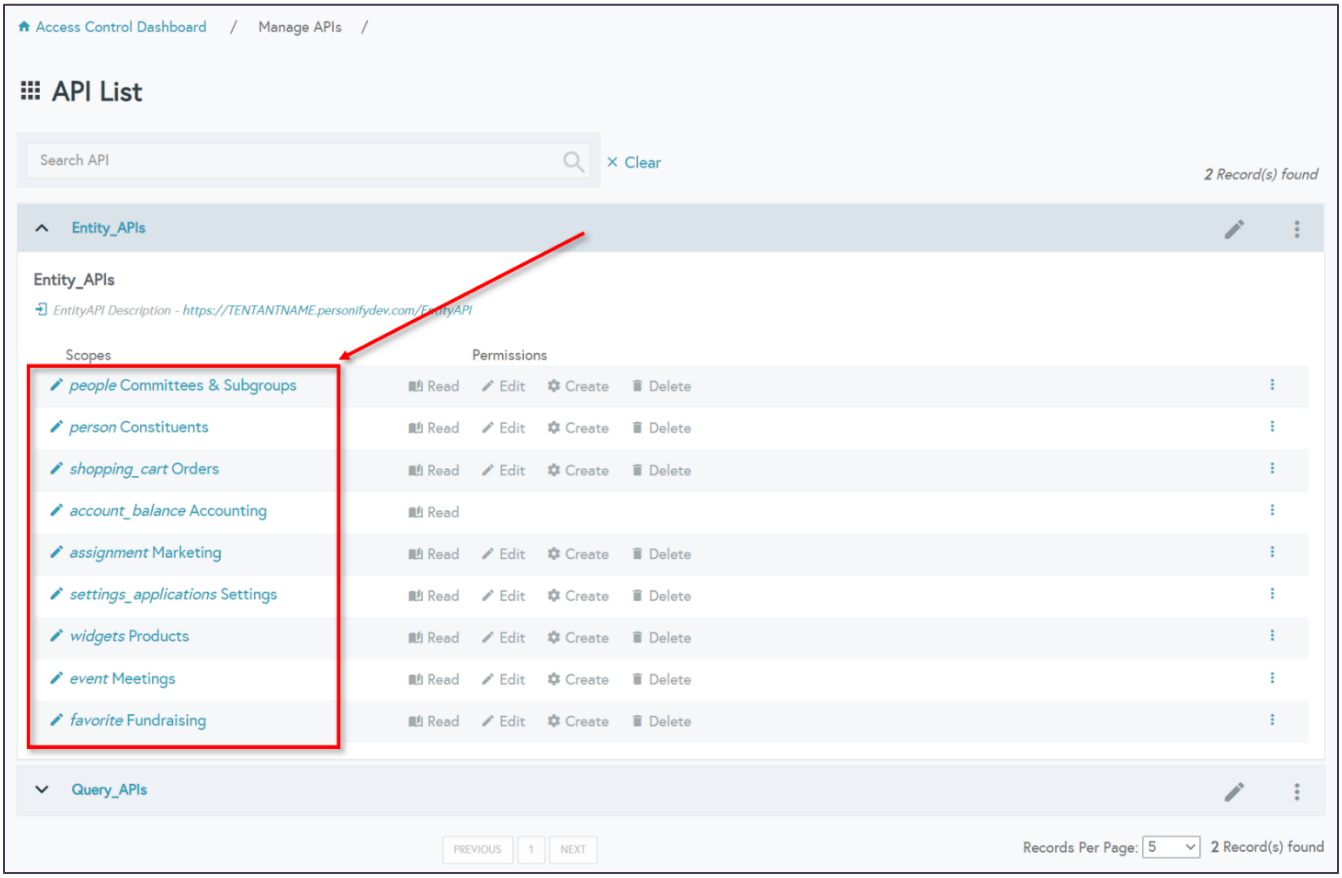


© 2021 Personify Inc.

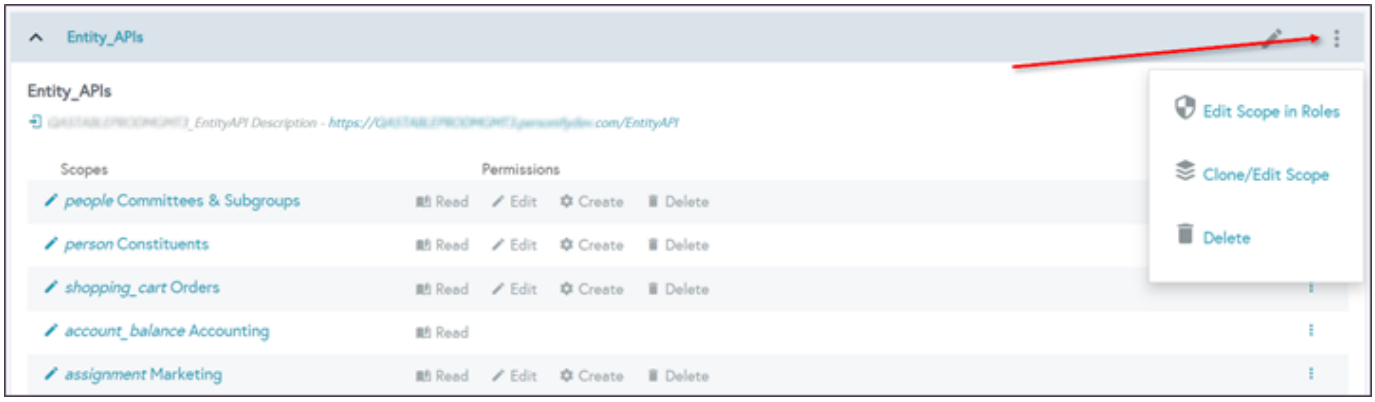
Query APIs are Read-Only APIs, therefore only the **Allow Read** checkbox is checked.

NOTE: When creating or editing an API, the system does not distinguish between "Entity" or "Query" APIs, therefore all operations are allowed. However, if the defined API is a Query API, only the "Read" option will matter.

From the API List page, you can also quickly navigate to edit scopes by selecting the scope from the API expand panel in the grid. For more information on editing and managing scopes, please see Managing Scopes.

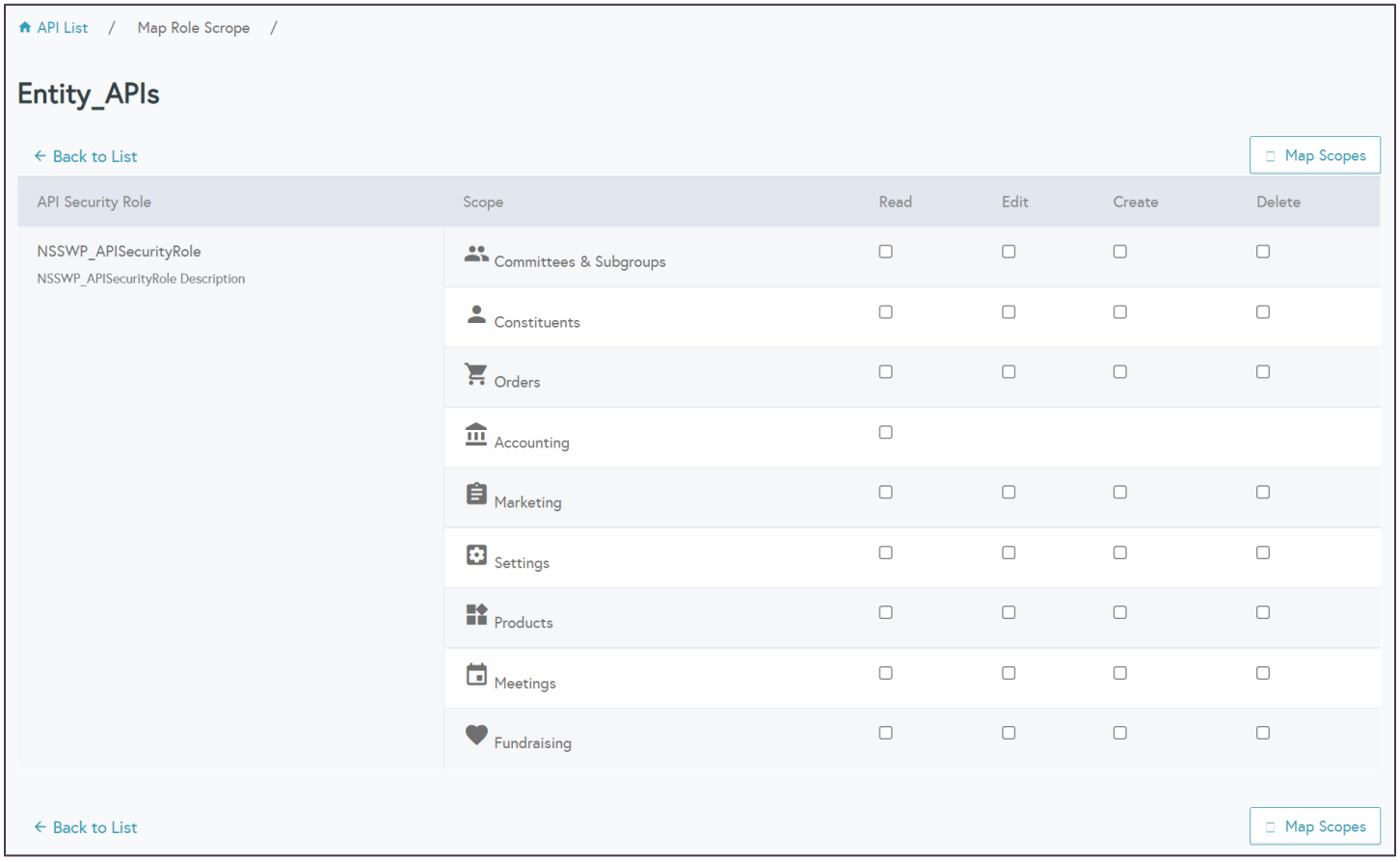


From the API List page, you can also quickly navigate to map role scopes for the selected API by selecting the **Edit Scope in Roles** option from the contextual menu.



© 2021 Personify Inc.

For more information on creating and managing roles, please see Managing Roles.

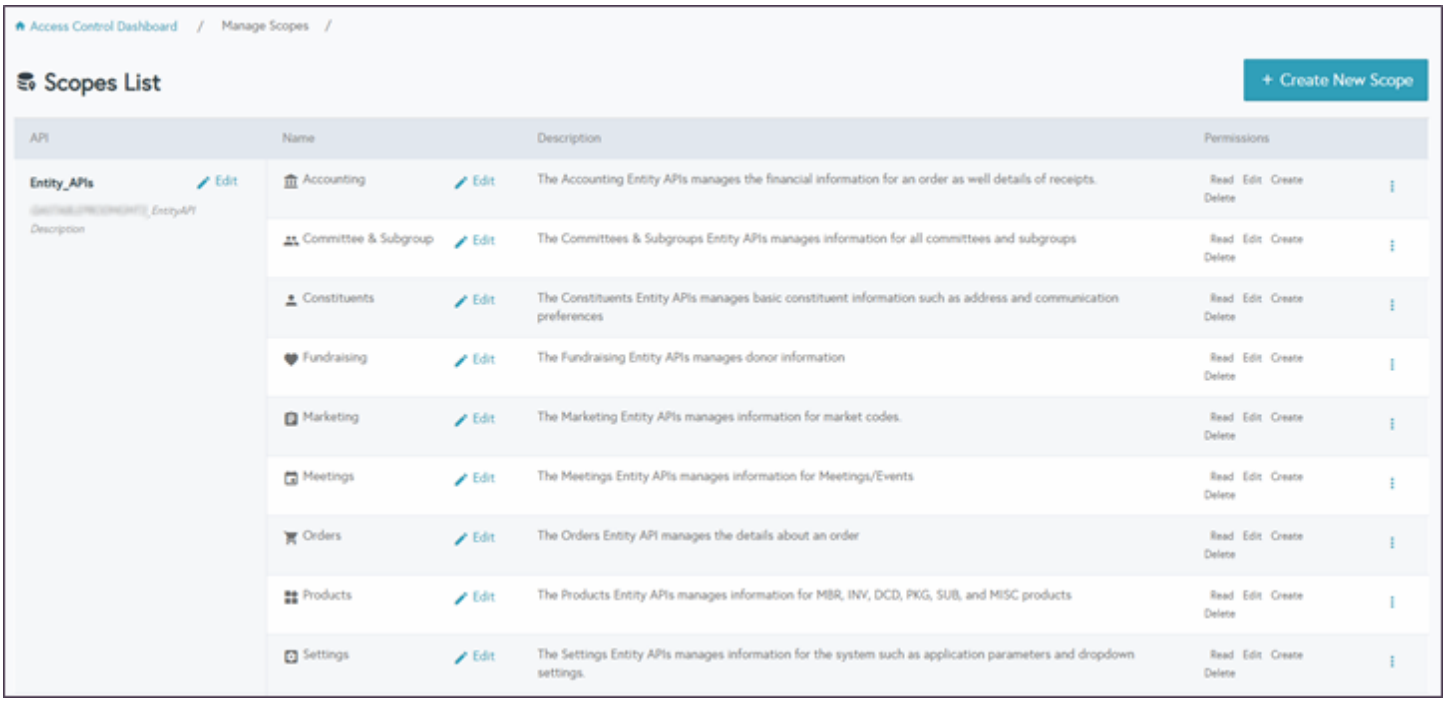


© 2021 Personify Inc.

Managing Scopes

Scope is a mechanism in OAuth 2.0 to limit an application's access to a user's account. An application can request one or more scopes. This information is then presented to the user on the consent screen, and the access token issued to the application will be limited to the scopes granted. Scopes are a way to control access and help the user identify the permissions they are granting to the application.

There are two separate ways within the Web Admin Portal to access and manage scopes. From the Access Control Dashboard, select **Manage Scopes** to view the Scopes List page; alternatively, select Manage APIs from the Access Control Dashboard to view the API List page, where you can use the expand panel in the grid to view all scopes listed under an API.



Web Admin provides several predefined scopes, but clients can create their own new scopes as needed.

The OAuth spec allows the authorization server or user to modify the scopes granted to the application compared to what is requested. OAuth does not define any particular values for scopes, as it is highly dependent on the service's internal architecture and needs.

The ultimate job of an OpenID Connect/OAuth token service is to control access to resources. Scope is a way to limit an application's access to a user's data. Rather than granting complete access to a user's account, it is often useful to give applications a way to request a more limited scope of what they are allowed to do on behalf of a user.

Some third-party vendors or applications/integrations only use OAuth to identify the user, so they only need access to a user's ID and basic customer information. Other third-party vendors may need to know more

© 2021 Personify Inc.

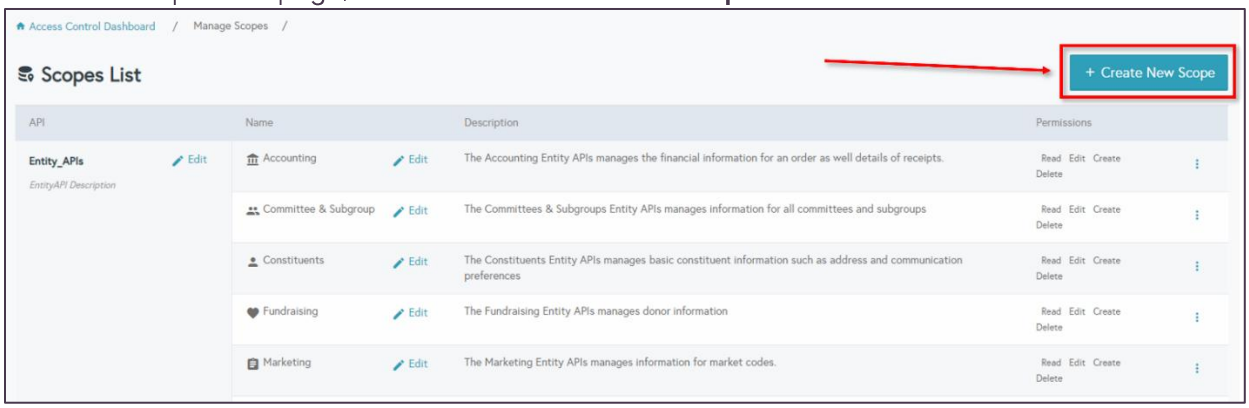
sensitive information such as financial information, or they may need the create orders or modify customer data. Scope is a way to control access and help the user identify the permissions they are granting to the vendor.

Defining Scopes

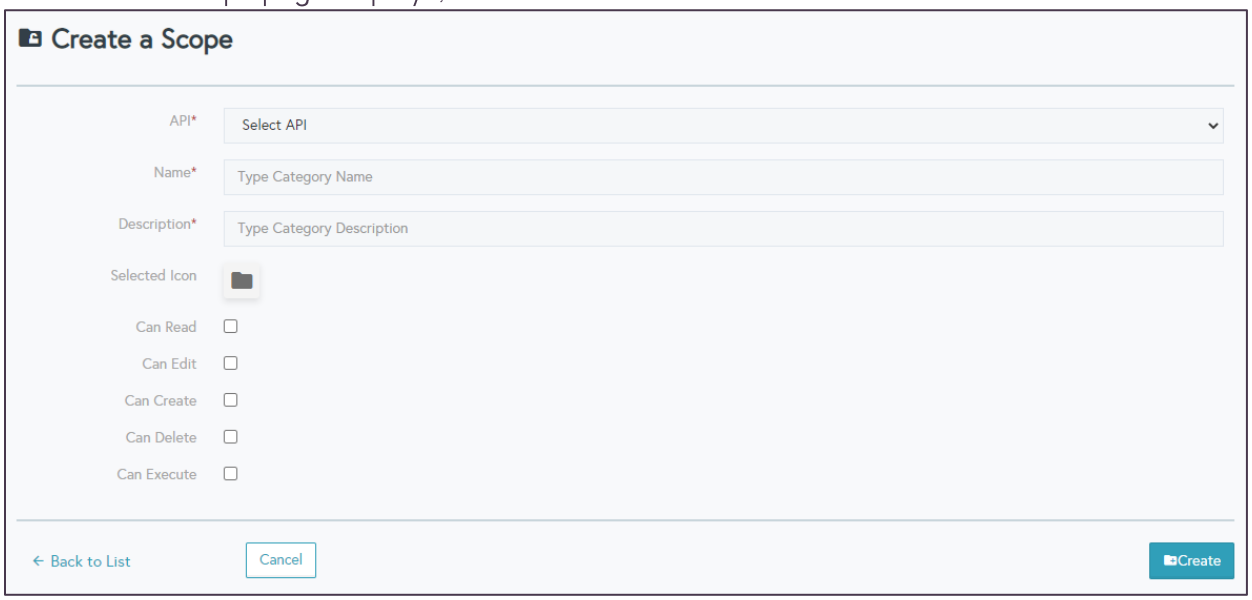
The challenge when defining scopes for your service is to not to define too many scopes. Users need to be able to understand the scope of the authorization they are granting, and this will be presented to the user in a list. When presented to the user, they need to understand what is going on. A significant use of scope is to selectively enable access to a user's account based on the functionality needed.

To create a new scope:

1. From the Scopes List page, select the **+Create New Scope** button.



2. The Create a Scope page displays, as shown below.



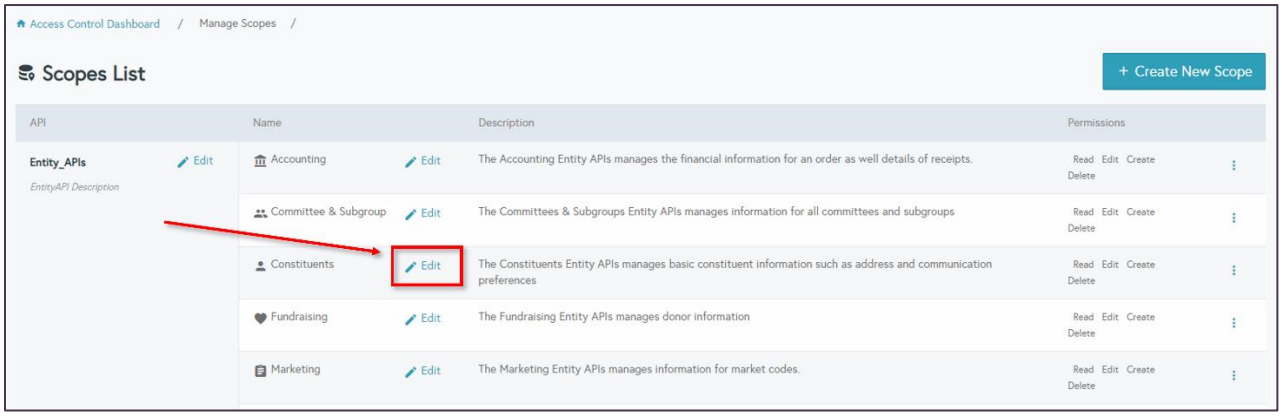
3. Select the **API** you wish to create a scope for from the drop-down – you will have two options, either the Entity or Query APIs.
3. Enter a **Name** for the scope.
4. Enter a **Description** for the scope.
5. Select a different **Icon**, if necessary. The selected icon will display on the Swagger page.

© 2021 Personify Inc.

6. Select the scope permission checkboxes, as necessary. The options available will depend on the API type selected in the API drop-down. For example, if a Query API is selected, only the Can Read option will be available. If the Entity API is selected, all below options will be available.
   a. Can Read
   b. Can Edit
   c. Can Create
   d. Can Delete
   e. Can Execute
7. Once you have completed the information for your new scope, select the **Create** button.
8. The scope is created and displays on the Scopes List page. You can now assign endpoints to the scope.

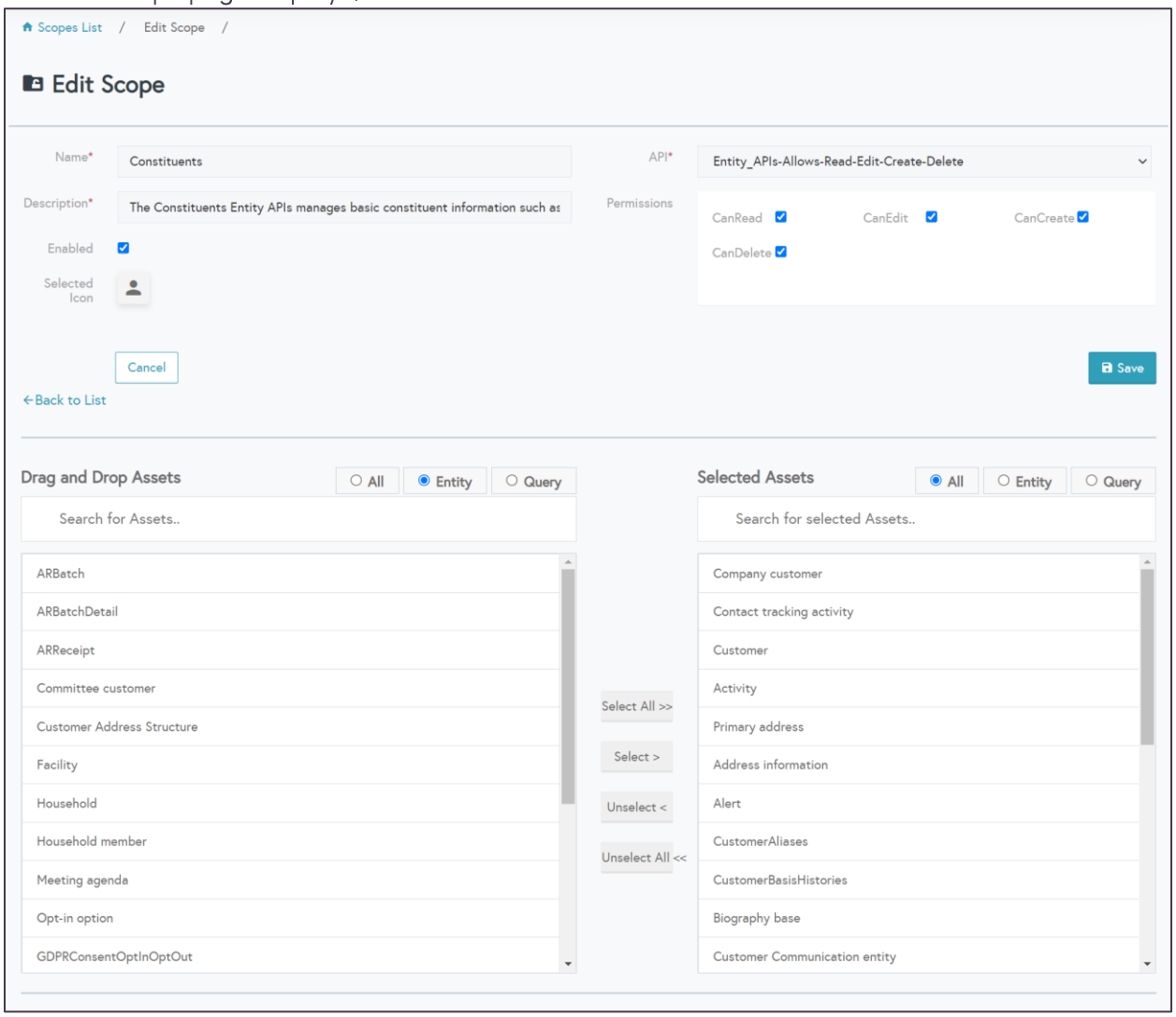
To edit and/or assign endpoints to a scope:

1. From the Scopes List page, select the **Edit** pencil hyperlink next to the corresponding scope you wish to edit.



© 2021 Personify Inc.

2. The Edit Scope page displays, as shown below.



3. At the top of the page, you can edit the information entered when the scope was created, such as the Name, Description, Icon, API, and API Permissions.
3. The **Enabled** checkbox determines if the scope is active or not. If this checkbox is unchecked, the scope will be disabled and will not display in the Swagger interface.
4. At the bottom of the Edit Scope page, you can define which endpoints are available within the scope. The endpoints listed on the right-hand side are the ones that have been selected for the scope. You can drag and drop endpoints from the list to the left to the selected list to the right, or you can use the **Select/Unselect** buttons to move endpoints between the two lists.

NOTE: You should only select either Entity or Query endpoints for a scope, but not both.

5. Once you have completed making your desired changes and assigning your desired endpoints, select the **Save** button to save your changes and return to the Scopes List.

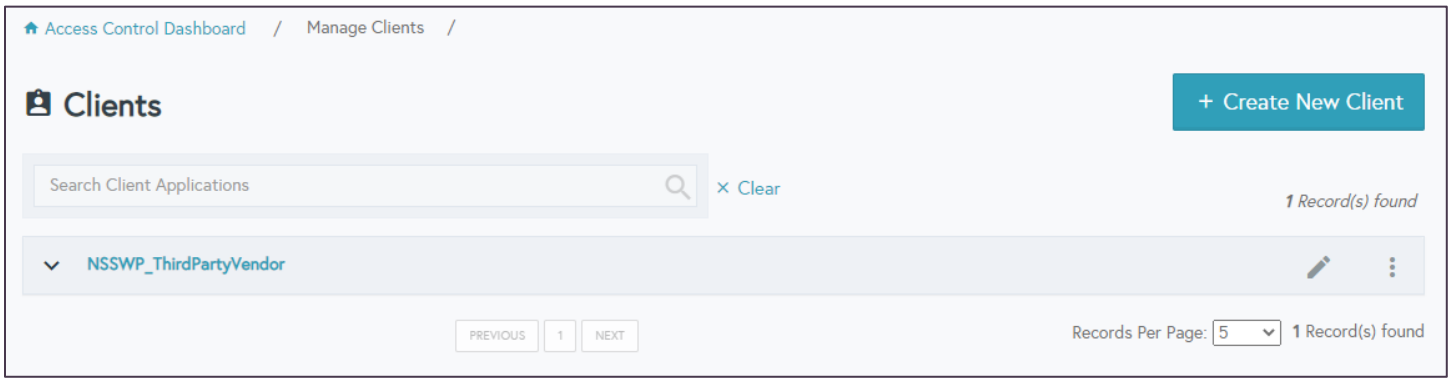
© 2021 Personify Inc.

Managing Clients

Organizations can create and manage client applications to give access to their various vendors. When creating a new client, organizations can select which role should be assigned to the client to give access to defined APIs and scopes.

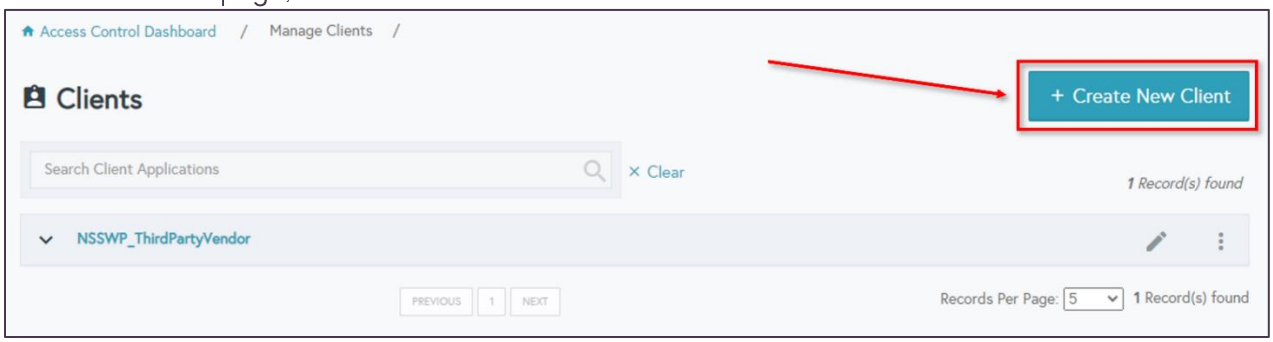
To access and manage clients in the Web Admin Portal, select **Manage Clients** from the Access Control Dashboard. From the Clients page, you can create a new client, as well as edit existing client settings or delete clients.

Web Admin collects basic information about an application, such as the Client application name before issuing the Client ID and Client Secret. It is recommended that you require the developer to register one or more redirect URLs for the application for security purposes.



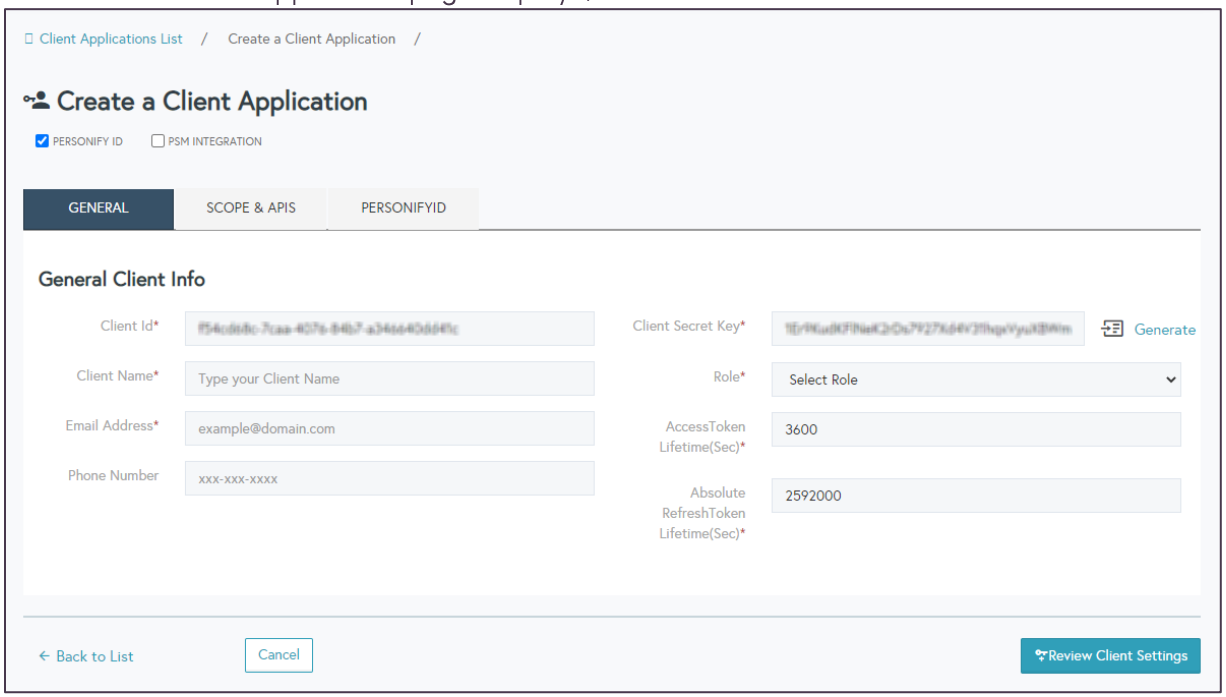
To create a new client:

1. From the Access Control Dashboard, select Manage Clients.
   The Clients page displays.
2. From the Clients page, select the **+Create New Client** button



© 2021 Personify Inc.

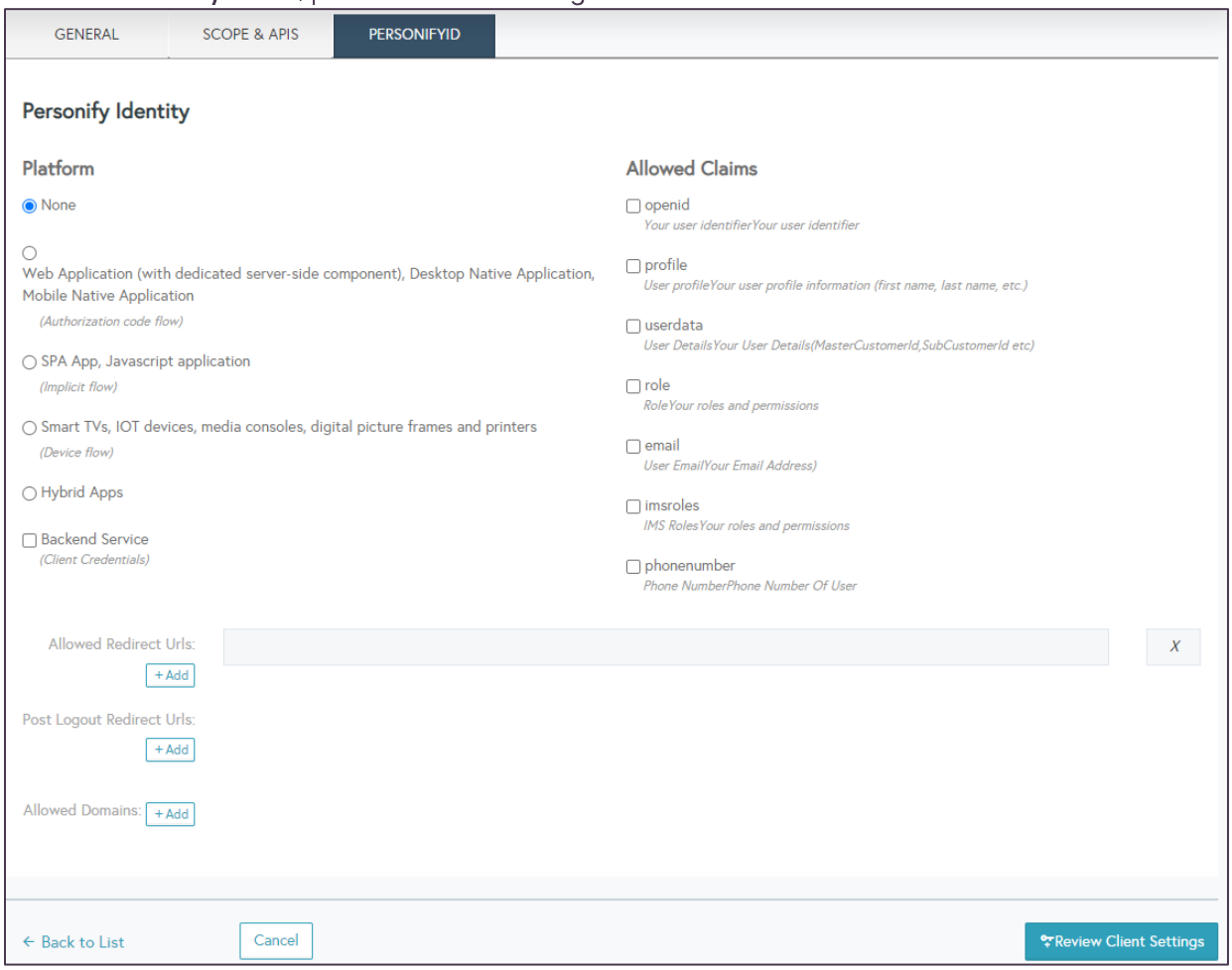
3. The Create a Client Application page displays, as shown below.



3. On the **General** tab, perform the following:
4. The **Client Id** and **Client Secret Key** are generated automatically. You can select the **Generate** button to regenerate the ID and Secret Key if necessary.
5. Enter the **Client Name**.
6. Enter the client's **Email Address**.
7. Enter the client's **Phone Number**, if necessary.
8. Select the client's **Role** from the drop-down.
   The selected role is how the system will determine the client's access.
9. The **AccessToken Lifetime** and **Absolute RefreshToken Lifetime** values will default. Change these values as necessary.
10. On the **Scope & APIs** tab, perform the following:
    The ultimate job of an OpenID Connect/OAuth token service is to control access to resources. Scope is a way to limit an application's access to a user's data. Rather than granting complete access to a user's account, it is often useful to give applications a way to request a more limited scope of what they are allowed to do on behalf of a user.
    a. Toggle on **Allow All Scopes** if you want to allow this client to access to all scopes.
    b. Check the checkbox next to each API you wish to allow access to.

© 2021 Personify Inc.

11. On the **PersonifyID** tab, perform the following:



a. Select the **Platform.**
   The platform is the type of application you are creating. The system will only issue a client secret for "web server" or Server-Side applications (the Web Application option below). Selecting a Platform reduces the likelihood of leaked secrets.
   The following options are available:
   i. Authorization Code Flow - **Web Application**, Server-Side Application
      This is the most commonly used platform.
   ii. Implicit Flow - Single-page Application (**SPA**)
      This option uses an ID token instead of authentication code.
   iii. Device Flow – **Smart TV**, Alexa, Amazon Fire Stick, etc.
      This option uses a code to authenticate the user and authorize the application or device.
   iv. **Hybrid** Application
      This option uses both an authentication code and an ID token. Personify eBusiness uses this option.
   v. **Backend Service**
      This option is used to directly integrate with the data without authenticating a user. For example, this option is used by Postman. You can have both a front-end option (Web Application, SPA, Smart TV, Hybrid App) and this Backend Service option enabled at the same time.

b. Select the **Allowed Claims**.
   The most common Allowed Claims are **openid**, **profile**, and **email**.

© 2021 Personify Inc.

The Allowed Claims determine what information you would like to share with the application you are creating. The options you select here will depend on what the consuming application needs, and what you would like to allow the application to use.
Some applications only use OAuth to identify the user, so they only need access to a user ID and basic profile information. Other applications may need to know more sensitive information such as the user's birthday, or they may need the ability to post content on behalf of the user, or modify profile data.

c. Enter the **Redirect URLs**.
For more information on Redirect URLs, please see the section below.
d. **Allowed Domains** are only used for single-page applications.

12. Once you have completed entering the client information, select the **Review Client Settings** button. The Review Settings page displays.
13. On the Review Settings page, you can review your selected options for the new client and make changes or correct any errors before completing client setup. Once you are satisfied with your client settings, select the **Save Settings** button to save your changes and return to the Clients page.

© 2021 Personify Inc.

Redirect URLs

Redirect URLs are a critical part of the OAuth flow. After a user successfully authorizes an application, the authorization server will redirect the user back to the application with either an authorization code or access token in the URL. Because the redirect URL will contain sensitive information, it is critical that the service does not redirect the user to arbitrary locations.

The best way to ensure the user will only be directed to appropriate locations is to require the developer to register one or more redirect URLs when they create the application.

REDIRECT URL REGISTRATION

To ensure the security of the service, you must require developers to register one or more redirect URLs for the application. The authorization server will never redirect to any other location.

If an attacker can manipulate the redirect URL before the user reaches the authorization server, they could cause the server to redirect the user to a malicious server which would send the authorization code to the attacker. For public clients without a client secret, all that is needed is the Client ID and authorization code to obtain an access token. If an attacker can obtain an authorization code, they could then exchange it for an access token for public clients. This is another reason it is important for the authorization server to know whether the application is public or private during application registration.

REDIRECT USERS AFTER LOGOUT

You can redirect users to a specific URL after they logout. You will need to register the redirect URL in your tenant or application settings. IDP only redirects to allow list URLs after logout. If you need different redirects for each application, you can add the URLs to the allow list in your application settings.

Add a returnTo query string parameter with the target URL as the value. Encode the target URL being passed in. For example, to redirect the user to http://www.example.com after logout, make the following request: https://YOUR_DOMAIN/v2/logout?returnTo=http%3A%2F%2Fwww.example.com.

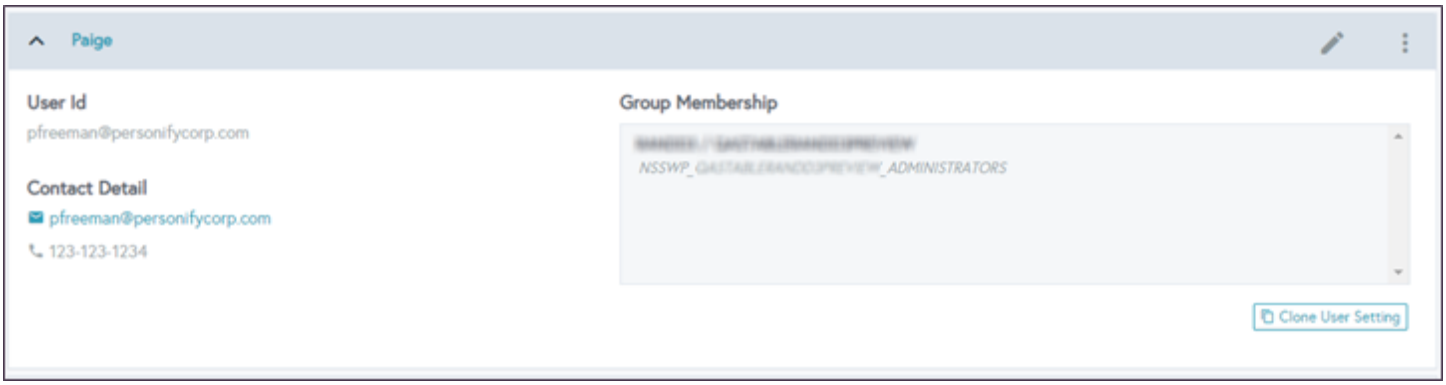
VALID REDIRECT URLS

When you build the form to allow developers to register redirect URLs, you should do some basic validation of the URL that they enter. Registered redirect URLs may contain query string parameters.

© 2021 Personify Inc.

Managing Users

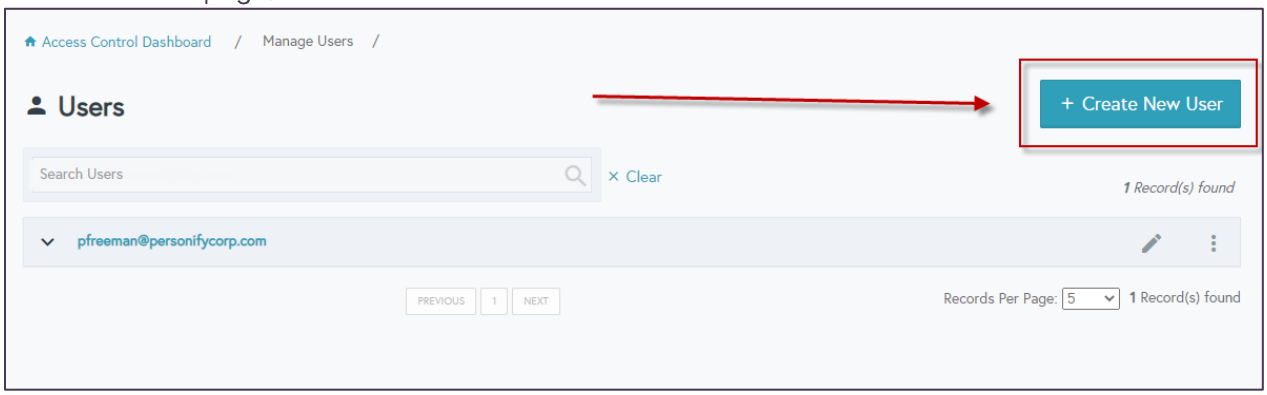
The Users page can be accessed from the Manage Users tile on the Access Control Dashboard. When a user first navigates to the Users page, the grid displays all the users for the user's specific environment, which is displayed on the top navigation bar. From the grid, each user can be expanded to view additional details about the user, including the User Id, Contact Detail, and Group Membership for the user.



Users can be deleted from the contextual menu. Click the Edit pencil to edit the user account details. Organization administrators can also create additional users as necessary from the Users page.

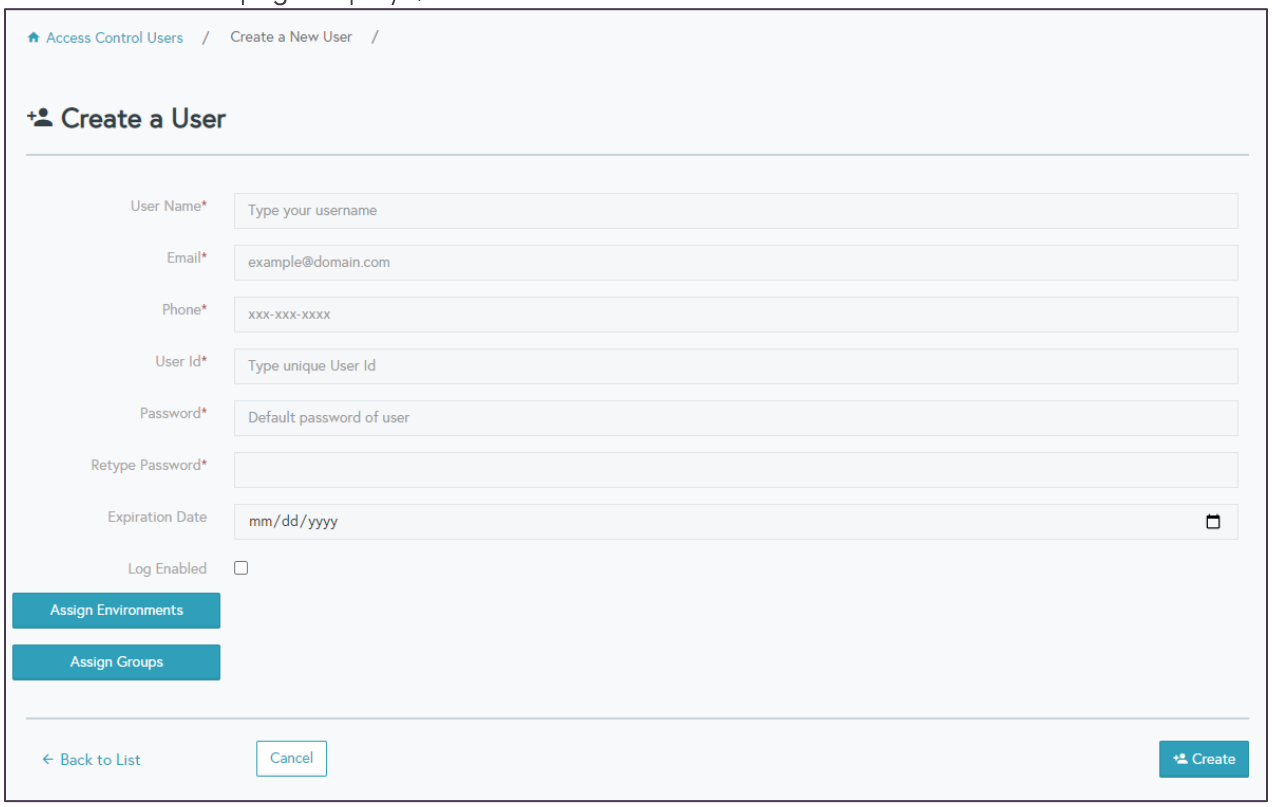
To create a new user:

1. From the Users page, select the **+Create New User** button.



© 2021 Personify Inc.

2. The Create a User page displays, as shown below.



3. On the Create a User page, perform the following:
   a. Enter the user's **User Name**.
      This is the name of the user.
   b. Enter the user's **Email** address.
      This email address will be used to send the Reset Password notification.
   c. Enter the user's **Phone** number.
      This is required for two-factor authentication purposes.
   d. Enter the user's **User Id**.
      The new user will use this ID to log in to the Web Admin Portal.
   e. Enter the user's **Password**.
      The new user will use this password to log in to the Web Admin Portal. You can also send a Reset Password email to the user after creation to have the user set their own password.
   f. Enter the user's password again in the **Retype Password** field.
   g. Enter an **Expiration Date** for the user, if necessary.
      If an expiration date is selected, the user account will be deactivated on the selected date.
   h. Leave the **Log Enabled** checkbox unchecked.
      This option is used by Personify developers for troubleshooting purposes only.
4. Select the **Assign Environments** button to select which environments this user should have access to.
   If you add multiple environments, use the radio button to denote which environment should be the default environment upon login.
5. Select the **Assign Groups** button to select which group permissions should be assigned to this new user.
   If you assign the user to a group that is associated with an environment, that environment will be automatically added to the user.

© 2021 Personify Inc.

6. Once you have completed entering the new user's information, select the **Create** button to create the new user.

You can also create multiple users all with the same settings using the Clone User Setting button on the Users page.

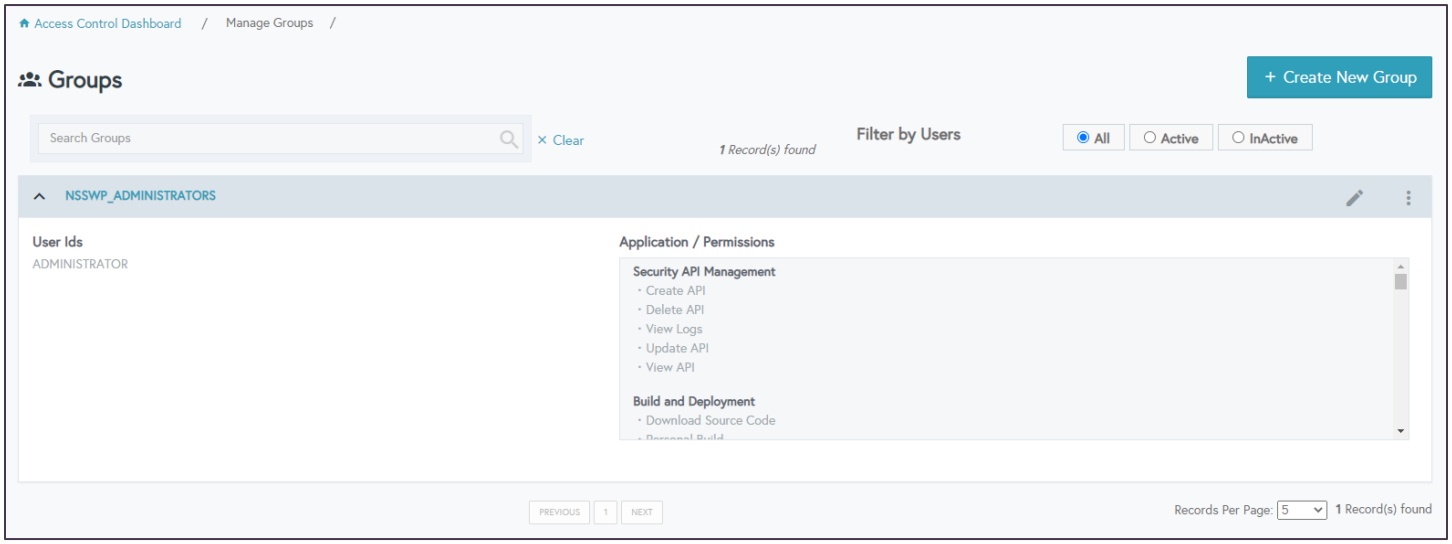
© 2021 Personify Inc.

Managing Groups

NOTE: Roles are used for API and Scope security and Groups are used for Application security (like Web Designer).

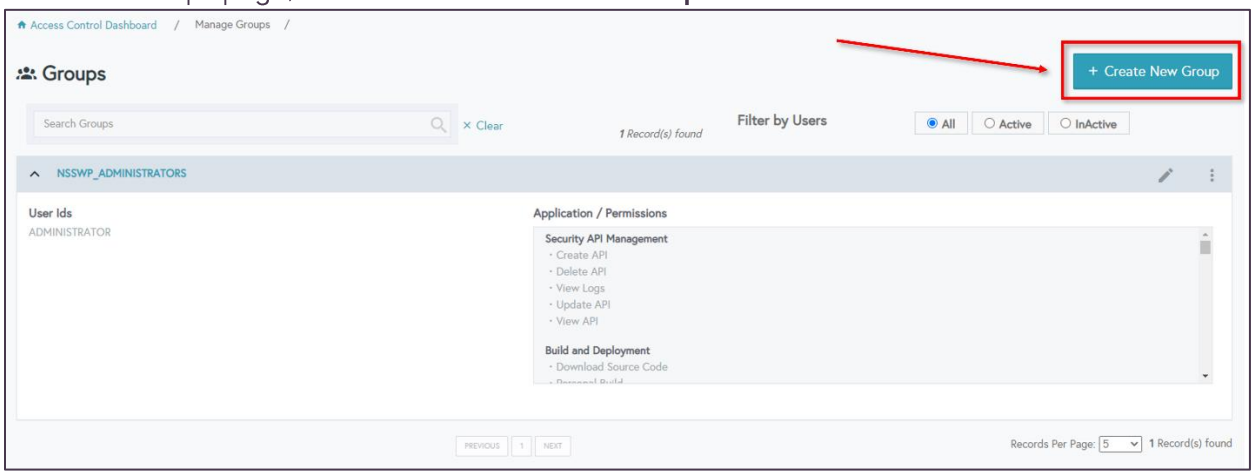
All users are created within a group. Clients can create however many groups they would like (such as "Administrators," "Power Users," etc.). Groups are how a client can define security around applications.

To access and manage groups in the Web Admin Portal, select **Manage Groups** from the Access Control Dashboard. From the Groups page, you can create a new group, as well as edit existing groups or delete groups.



To create a group:

1. From the Groups page, select the **+Create New Group** button.



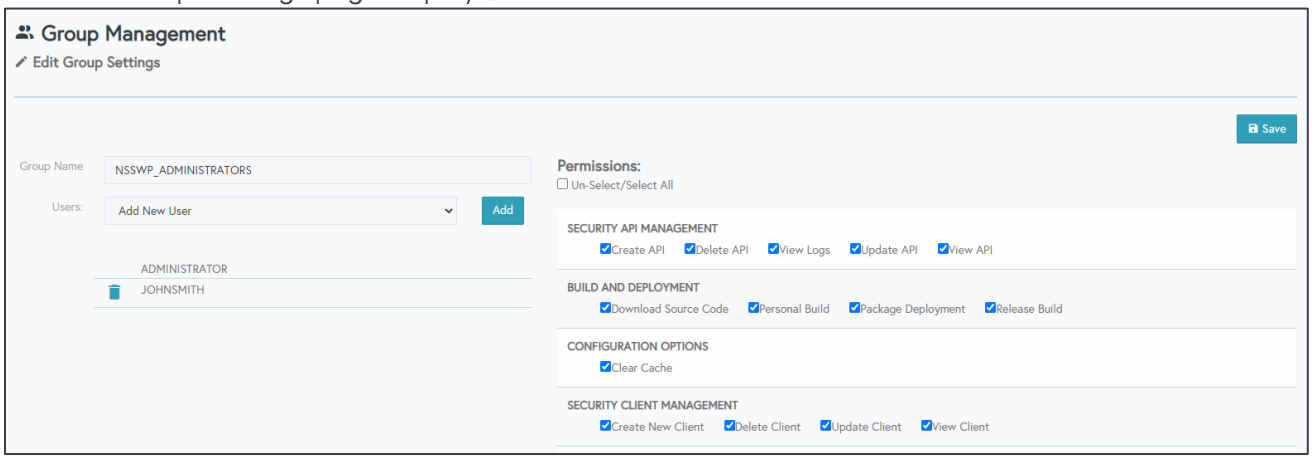
© 2021 Personify Inc.

2. The Create Group page displays, as shown below.

3. Enter the **Group Id**.
4. Enter the **Group Name**.
5. Select the **Environment** from the drop-down.
6. Select the **Create** button to create your group and return to the Groups page.

To add users to a group and/or select group permissions:

1. From the Groups page, select the edit pencil next to the corresponding group you wish to edit.
2. The Edit Group Settings page displays, as shown below.



3. Add users to the group using the **Users** drop-down. Select a user and then select the **Add** button to add the user to the group.
3. Delete users from the group using the trash can icon next to the user in the list.
4. Select the permissions to assign to this group using the checkboxes on the right of the page.
5. Once you have added your users and/or your permissions, select the **Save** button to save your changes and return to the Groups page.

Security Permissions

Below are the available security permissions that can be set at the Group level:

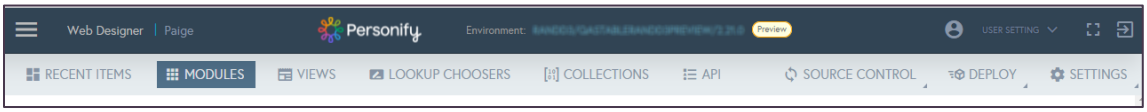
© 2021 Personify Inc.

Web Designer

Web Designer is a tool in the Web Admin Portal that can be used to create and edit a variety of pages and objects that are used by ThreeSixty Web Client.

Overview & Navigation

The Web Designer's main sections are accessible in the ribbon located below the top navigation bar, which contain links to direct the user to different areas of the Web Designer.

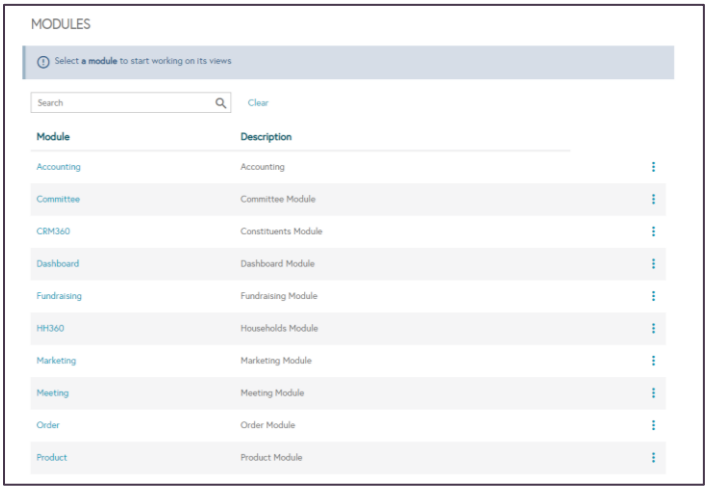


Recent Items

The default page of Web Designer, it shows the most recently edited Views.

Modules

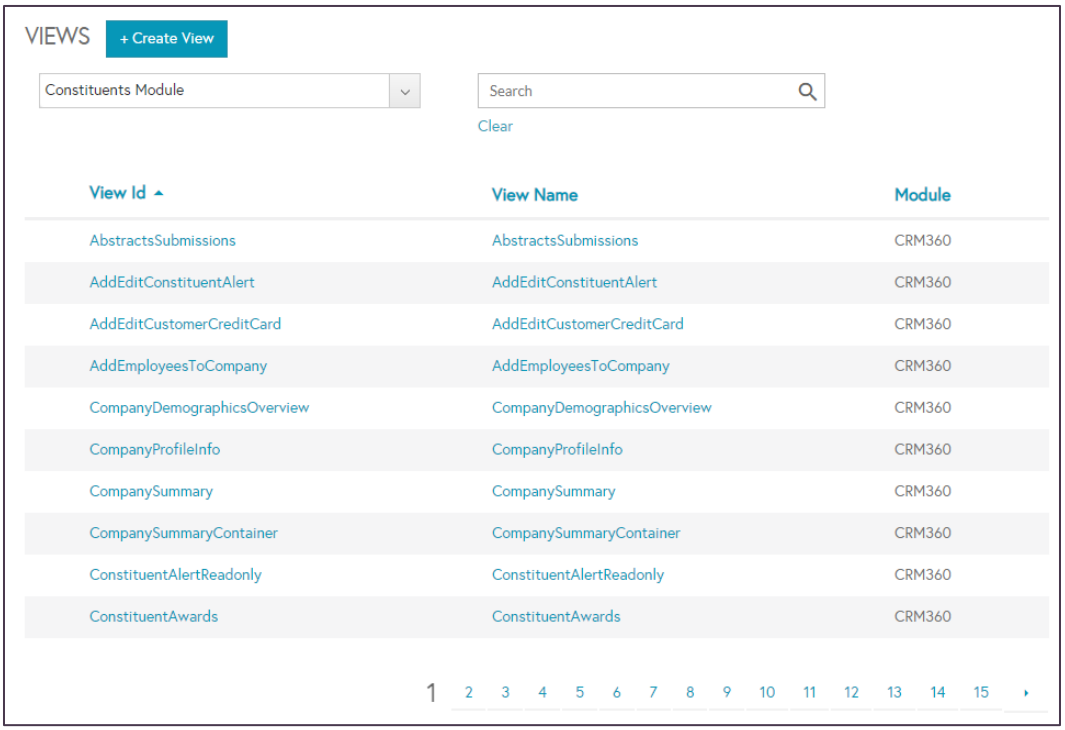
These modules are direct analogs of the modules on the left-hand toolbar in ThreeSixty Web Client, and they usually have a strong relation to API Namespaces of the same name. This page shows a list of the names and descriptions of all available modules and links to their view lists, which also accessible via the Views link on the ribbon.



© 2021 Personify Inc.

Views

This page lists all views associated with a specific module that can either be selected from the drop-down or selected via the Modules page.



Selecting a linked View Id or View Name in the list will take you to the canvas editor page, which allows you to edit the contents of the view by dragging and dropping its elements. Selecting the Create View button opens the view creation wizard.

A "view" represents a visible portion of ThreeSixty Web Client that can be customized to display or edit data or contain other views. A view usually consists of multiple xml files: a view file, a resource file, a repository file, and/or a viewmodel file.

- The View file is responsible for determining the placement of html controls on the page, like textboxes, links, and labels.
- The resource file can be used to specify what text is written on the page and resource references are denoted by a "$" at the front. For example, a button might have a text value of "$btnSave" which references a key-value pair in the resource file
- The repository file specifies which API Objects/Collections/Schemas that data will be sourced from
- The viewmodel file is a mapping between the datasources from the repository and the UI controls on the page via the "PropertyName" tag.
- There are also JavaScript Extension file that can be added to some views that need custom UI interactions not supported by the standard control palette and it gets imported along with the generated JavaScript upon runtime.

There are primarily two types of Views: Simple Views (also known as Shared or CRUD Views), located in the "Shared Module," and Regular Views (also known as non-shared, partial, or non-simple Views).

© 2021 Personify Inc.

Simple Views are comprised of only a View File and optionally a Resource File. They can be accessed from any module in ThreeSixty Web Client if they are passed the correct context data (for example, MasterCustomerId).

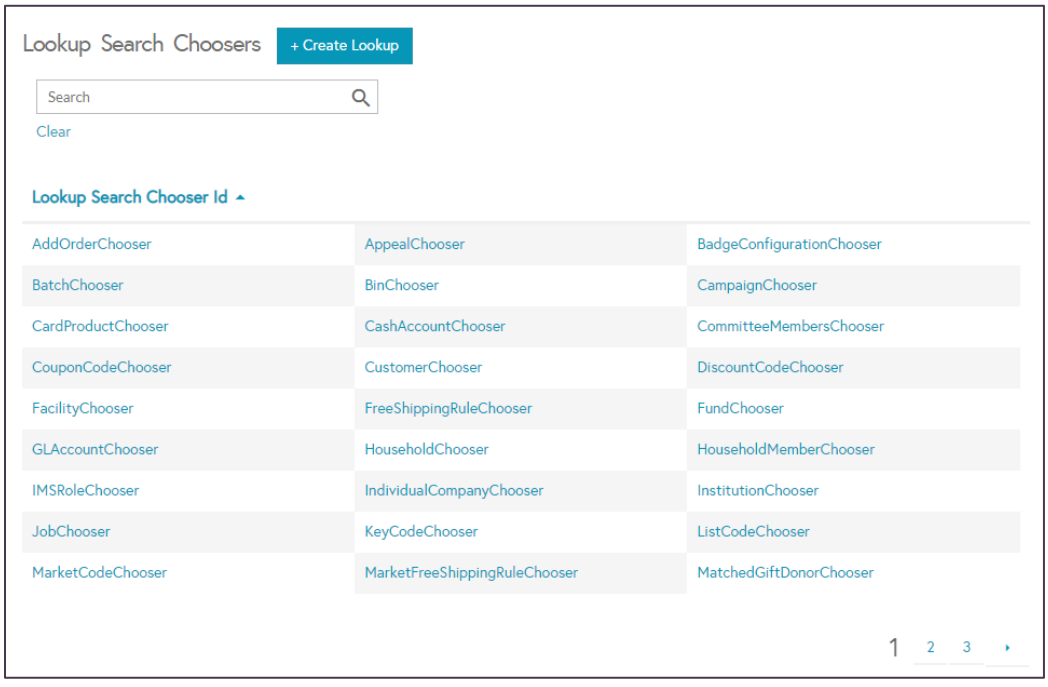
- A simple view can take on three forms: List, AddEdit, and ReadOnly. Visually, they appear as three different pages but they all reside within the same xml file.
- The list form for a BusinessObjectCollection will show one BusinessObject per row. There are links on each row to transition the list form to a read-only or edit form. There is a create link above the list to transition the page to an add form.
- The read-only form displays non-editable data whereas the add/edit form contains input controls to edit data. The add page uses the same xml as the edit form. The isCreate flag is used to determine if the title of the page has "Add" or "Edit" when viewed in the Web Client, which is set based on whether the user reached the page by clicking a table-row-link or clicking on a create button. Clicking on the cancel or save button will return the user to the list page (or the read-only page if the user is working with a BusinessObject instead of a Collection).

Regular or Non-Simple Views usually are confined to usage within one module and employ all four xml types to work correctly (view, resource, repository, and viewmodel).

Lookup Choosers

There are special controls within ThreeSixty Web Client that can be used to select business objects and can provide a clean interface for searching and selecting these objects. An example is if we wanted to create a page with product selection, the page could have a product lookup control that has a searchbox with a drop-down that shows images of its search results. If we wanted to customize what we show in this drop-down, we will have to access the product lookup control's internal code from the Lookup Choosers page.

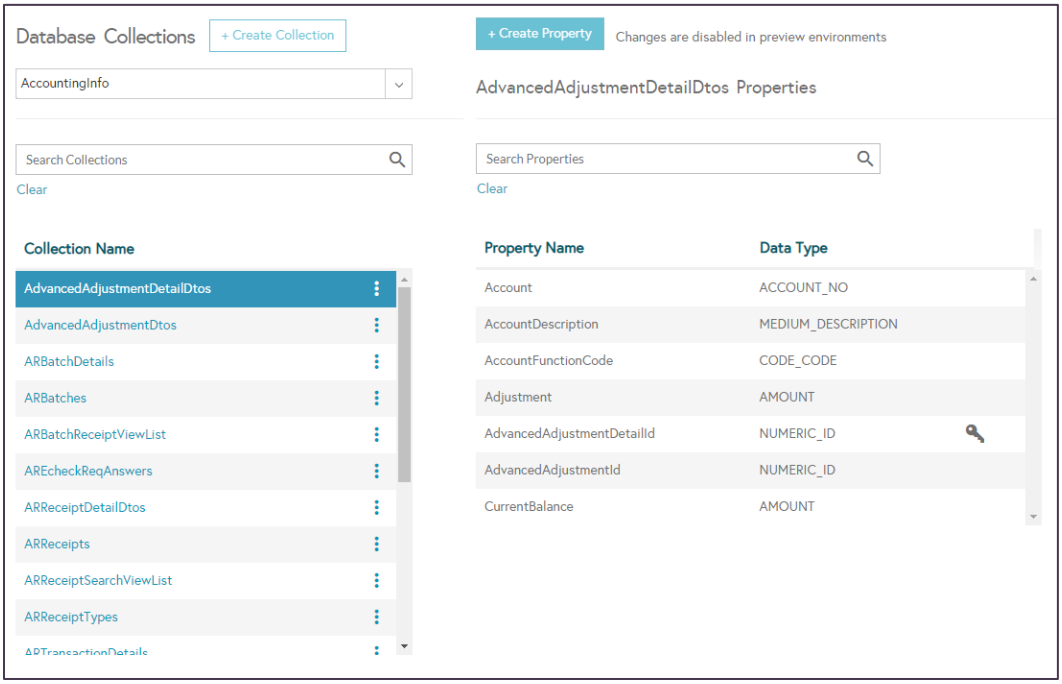
Lookup Choosers consist of two xml files, a "Chooser" and "Config" file. The xml syntax in these files is different from that used in the View files.



© 2021 Personify Inc.

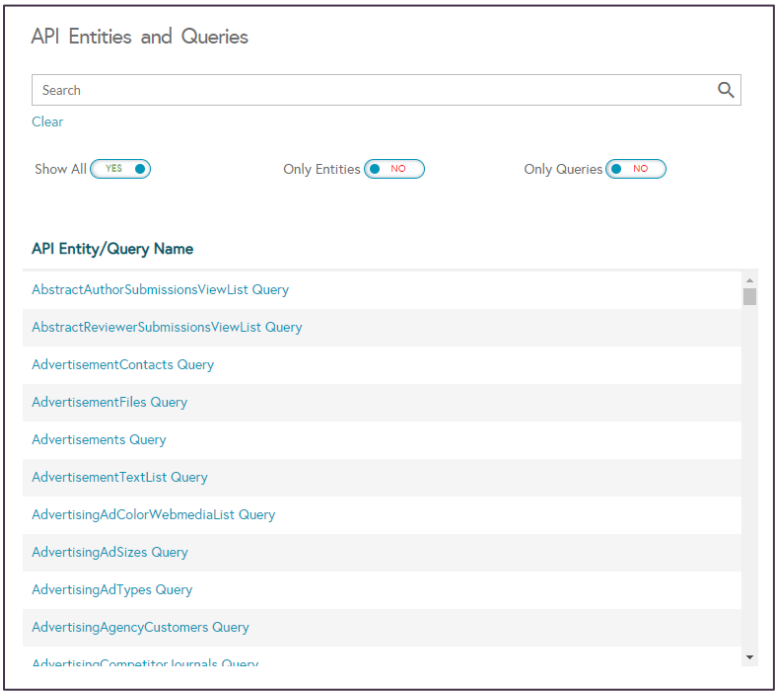
Collections

BusinessObject Collections and BusinessObject properties can be viewed on the Collections page. Custom properties can be edited and added where applicable and new collections can be created under the specified namespaces. Entities and Queries can also be accessed and created via each row in the collection list.



API

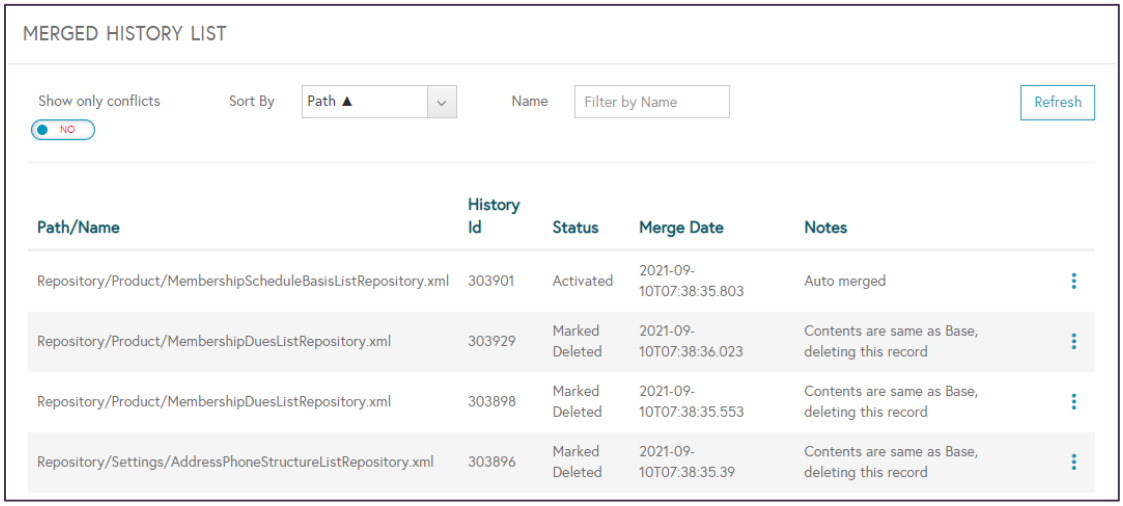
All existing API entities and queries can be viewed and edited on the API page. However, adding any entities or queries must be done via Collections.



© 2021 Personify Inc.

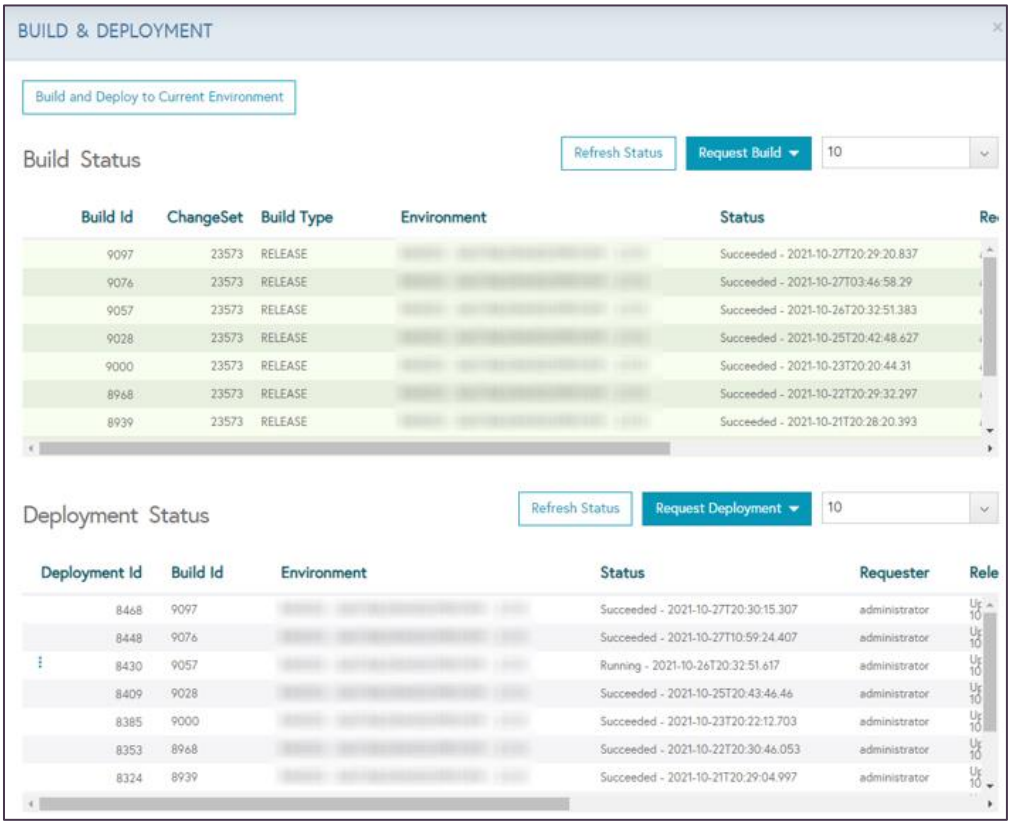
Source Control

Any API or View changes must be checked into Source Control before being built into a package and deployed on a server. Any pending changes can also be undone if no longer needed. There is also a Conflict Resolution page for facilitating environmental base upgrades.



Deploy

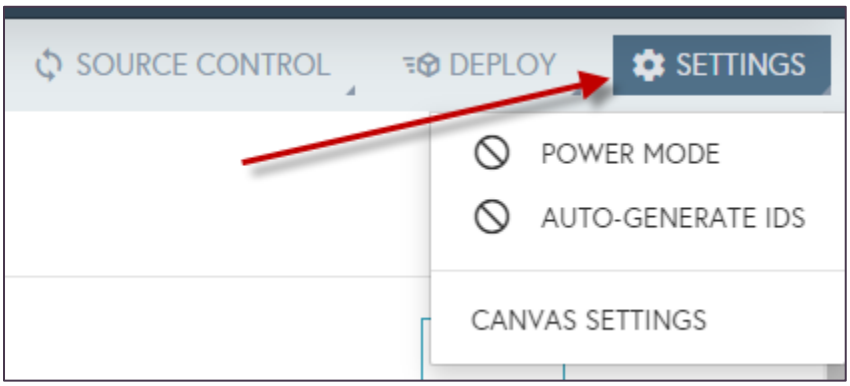
This dialog displays the status of all builds and deployments for the currently selected environment, which is displayed above the ribbon in the center. New builds and deployments can also be requested and old deployments can be cancelled from here.



© 2021 Personify Inc.

Settings

Settings has a small number of options for the view canvas. Auto-Generating IDs will append randomized numbers to new controls added to a page. Selecting Power Mode show the tabs for accessing extended JavaScript, repositories, and the markup editor, which provides direct access to the view/viewmodel/resource/repository xml code.



© 2021 Personify Inc.

Novus API Security

The Novus API security is based on the OAuth2 protocol. Personify does not support passing client credentials in the headers; instead, you must pass the Client ID and Client Secret. After passing the Client ID and Client Secret, the Novus APIs return a JWT (JSON Web Token) token, which will be used for any subsequent requests.

The options that Personify provides for OAuth2 are available within the Swagger interface, accessible from the OAuth 2.0 option on the API drop-down menu. The options that are used for the Novus APIs are the AccessToken and RefreshToken.

Access Token and Refresh Token

Access tokens enable clients to securely call the APIs, and are used by the APIs to perform authentication and authorization. The AccessToken contains the requests made to get the token, including Client ID, Client Secret, and Audience. The Audience is important as it identifies the API and scopes for the token. For example, when you select Authorize from the Home menu of the API, the Swagger interface knows that you are requesting authorization for all the scopes that are defined for the API. If you open a particular scope and then get the authorization, then the Swagger interface is only going to get the authorization for that scope – meaning if you attempt to move between scopes, you won't be authorized for the other scopes.

You can adjust the lifetime of an access token to control how often the client application expires the application session, and how often it requires the user to re-authenticate (either silently or interactively). Whereas the access token has a short lifecycle, the refresh token has a longer lifecycle. If the current access token becomes invalid or expires, the refresh token can be used to request a new access token.

© 2021 Personify Inc.

Client Credentials Flow

You can use the OAuth 2.0 client credentials grant specified in RFC 6749, sometimes called *two-legged OAuth*, to access web-hosted resources by using the identity of an application. This type of grant is commonly used for server-to-server interactions that must run in the background, without immediate interaction with a user. These types of applications are often referred to as *daemons* or *service accounts*, which is how the Novus APIs were built. Refer to the Daemon and Service-Side Application section for further information about these types of applications.

The OAuth 2.0 client credentials grant flow permits a web service (confidential client) to use its own credentials, instead of impersonating a user, to authenticate when calling another web service. Because the applications own credentials are being used, these credentials must be kept safe - *never* publish that credential in your source code, embed it in web pages, or use it in a widely distributed native application.

In the client credentials flow, permissions are granted directly to the application itself by an administrator. When the application presents a token to a resource, the resource enforces that the application itself has authorization to perform an action since there is no user involved in the authentication.

DAEMONS AND SERVER-SIDE APPLICATIONS

Applications that have long-running processes or that operate without interaction with a user also need a way to access secured resources. These applications can authenticate and get tokens by using the application's identity, rather than a user's delegated identity, with the OAuth 2.0 client credentials flow. You can prove the application's identity using a client secret or certificate.

GET A TOKEN

After you have acquired the necessary authorization for your application by creating and setting up a client from the Access Control of the Web Admin panel, proceed with acquiring access tokens for APIs. To get a token by using the client credentials grant, send a POST request to the /token IDP (Identity Provider).

Access token request with a shared secret:

```
POST /{tenant}/connect/token HTTP/1.1          //Line breaks for clarity
Host: login.personifygo.com
Content-Type: application/x-www-form-urlencoded

client_id=535fb089-9ff3-47b6-9bfb-4f1264799865
&scope=tenantname_entityapi
&client_secret=sampleCredentia1s
&grant_type=client_credentials

# Replace {tenant} with your tenant!
curl -X POST -H "Content-Type: application/x-www-form-urlencoded" -d 'client_id=535fb089-9ff3-
47b6-9bfb-4f1264799865&scope=tenantname_entityapi
```

© 2021 Personify Inc.

```
&client_secret=qWgdYAmab0YSkuL1qKv5bPX&grant_type=client_credentials'
'https://login.personifygo.com/{tenant}/connect/token'
```

Parameter	Condition	Description
tenant	Required	The directory tenant the application plans to operate against, in GUID or domain-name format.
client_id	Required	The application ID that is assigned to your application.
scope	Required	The value passed for the scope parameter in this request should be the resource identifier.
client_secret	Required	The client secret that you generated for your application.
grant_type	Required	Must be set to client_credentials.

Successful Response

A successful response looks like this:

```
{
  "token_type": "Bearer",
  "expires_in": 3599,
  "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Ik1uQ19WWmNBVGZNNXBP..."
}
```

Parameter	Description
access_token	The requested access token. The application can use this token to authenticate to the secured resource, such as to a web API.
token_type	Indicates the token type value. The only type that the IDP platform supports is bearer.
expires_in	The amount of time that an access token is valid (in seconds).

© 2021 Personify Inc.

USE A TOKEN

Once you have authenticated and acquired a token, use the token to make requests to the Novus APIs. You must use the token to make calls/with every resource request. When the token expires, repeat the request to the /token endpoint to acquire a fresh access token.

```
GET /EntityAPI/data/api/v2/ContactTrackingActivities
Host: https://tenantname.personifygo.com
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Ik5HVEZ2ZEstZnl0aEV1Q...

# Pro tip: Try the following command! (Replace the token with your own.)

curl -X GET -H "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbG...."
'https://tenantname.personifygo.com/EntityAPI/data/api/v2/ContactTrackingActivities'
```

Validating Tokens

Not all applications should validate tokens. Only in specific scenarios should applications validate a token:

- Web APIs must validate access tokens sent to them by a client. They must only accept tokens containing their audience claim.
- Confidential web applications like ASP.NET Core must validate ID tokens sent to them via the user's browser in the hybrid flow, before allowing access to a user's data or establishing a session.

If none of the above scenarios apply, your application will not benefit from validating the token, and may present a security and reliability risk if decisions are made based on the validity of the token. Public clients like native applications or SPAs don't benefit from validating tokens - the application communicates directly with the IDP, so SSL protection ensures the tokens are valid.

APIs and web applications must only validate tokens that have an aud claim that matches their application; other resources may have custom token validation rules.

VALIDATING THE SIGNATURE

A JWT contains three segments, which are separated by the '. 'character. The first segment is known as the **header**, the second as the **body**, and the third as the **signature**. The signature segment can be used to validate the authenticity of the token so that it can be trusted by your application.

You can acquire the signing key data necessary to validate the signature by using the OpenID Connect metadata document located at: https://login.personifygo.com/{tenantId}/.well-known/openid-configuration

This metadata document:

- Is a JSON object containing several useful pieces of information, such as the location of the various endpoints required for doing OpenID Connect authentication.
- Includes a jwks_uri, which gives the location of the set of public keys that correspond to the private keys used to sign tokens. The JSON Web Key (JWK) located at the jwks_uri contains all the public key information in use at that moment in time. The JWK format is described in RFC 7517. Your application can

© 2021 Personify Inc.

use the kid claim in the JWT header to select the public key, which corresponds to the private key that has been used to sign a particular token. It can then do signature validation using the correct public key and the indicated algorithm.

HANDLING AUTHENTICATION ERROR CODES IN YOUR APPLICATION

The OAuth2.0 spec provides guidance on how to handle errors during authentication using the error portion of the error response.

Here is a sample error response:

```
{
  "error": "invalid_scope",
  "error_description": " The provided value for the input parameter 'scope' is not valid.",
  "timestamp": "2016-01-09 02:02:12Z"
}
```

Error Code	Description	Client Action
invalid_request	Protocol error, such as a missing required parameter.	Fix and resubmit the request.
invalid_grant	Some of the authentication material (auth code, refresh token, access token, PKCE challenge) was invalid, could not be parsed, missing, or otherwise unusable.	Try a new request to the /authorize endpoint to get a new authorization code. Consider reviewing and validating that application's use of the protocols.
unauthorized_client	The authenticated client is not authorized to use this authorization grant type.	This usually occurs when the client application is not registered in Web Admin or is not added to the tenant.
invalid_client	Client authentication failed.	The client credentials are not valid. To fix, the application administrator updates the credentials.
unsupported_grant_type	The authorization server does not support the authorization grant type.	Change the grant type in the request. This type of error should occur only during development and be detected during initial testing.

© 2021 Personify Inc.

References

- *OAuth 2.0 Scopes*, https://oauth.net/2/scope/.
- *OAuth 2.0 Simplified*, https://oauth.com/.
  - Scope - Defining Scopes
  - Client Registration - Registering a New Application
  - Redirect URLs - Redirect URL Registration
- *Microsoft Identity Platform Documentation*, https://docs.microsoft.com/en-us/azure/active-directory/develop/.
  - Authentication & Authorization error codes
  - Identity platform access tokens
  - OAuth 2.0 authorization code flow
  - OAuth 2.0 client credentials flow
  - Application types
- *What is OAuth 2.0 Client credentials flow and when would ...*, https://www.bettercoder.io/job-interview-questions/2431/what-is-oauth-20-client-credentials-flow-and-when-would-you-use-it.

© 2021 Personify Inc.