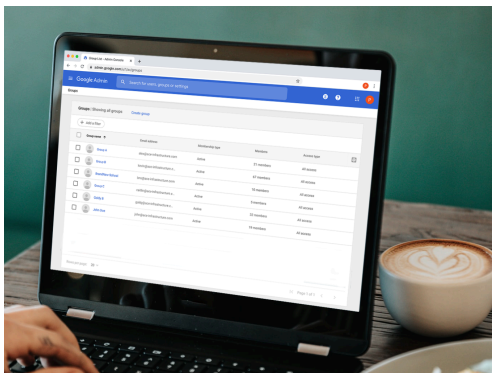
chrome

Getting started with the Chrome Enterprise Core **API**

Last updated Oct 2024



This document will guide you through the process of setting up the API in Chrome Enterprise Core. It also includes a section on how to set up the integration with the Postman API Platform. By using Postman you can accelerate the design, mocking, and testing of your scripts to make the process easier than ever to use Chrome Enterprise Core APIs. By using Cloud management APIs, you can push and pull data into the console at scale and enhance the level of reporting that you can gather from your enrolled browsers. This document assumes the following:

- Have access to the Google admin console
- A super admin account for the setup
- At least one (or more) device(s) enrolled into Chrome Enterprise Core for testing

Here are some useful links:

Chrome Browser Enterprise Github
Readme file for the API
Google Cloud Platform Console
Postman API Platform

How Scripts Authenticate with the admin console

Authorize with consent

Each time you request a token, the admin will be prompted to authorize the request via the API.

- Postman scripts are developed using this authorization method. For validating API calls, this is the recommended method for your integration development work

---

Decide which APIs scopes (functionality) you want to use

The API provides four different APIs that you can enable. Depending on what requests or actions you want to take via API could determine which scopes you enable. You can also enable them with read or write access (or both).

Complete descriptions of what each can do is located in the Github documentation.

Here is a brief overview of each one:

Scope	Description
Write scopes	
https://www.googleapis.com/auth/admin.directory.device.chromebrowsers	**Chrome Enterprise Core** Lets you view and modify enrolled browsers and enrollment tokens
https://www.googleapis.com/auth/admin.directory.orgunit	**Org Units** - lets you view and modify organizational units
https://www.googleapis.com/auth/chrome.management.policy	**Chrome Policy** - lets you view and modify Chrome policies for devices and users
Read only scopes	
https://www.googleapis.com/auth/admin.directory.device.chromebrowsers.readonly	**Chrome Enterprise Core** Get detailed information on enrolled browsers and enrollment tokens (read-only)
https://www.googleapis.com/auth/chrome.management.reports.readonly	**Reports** - Chrome versions and installed apps (read-only)

https://www.googleapis.com/auth/chrome.management.appdetails.readonly	**App Details** - get detailed information about requested or specified apps (read-only)
https://www.googleapis.com/auth/chrome.management.policy.readonly	**Chrome Policy** - lets you view Chrome policies for devices and users (read-only)
https://www.googleapis.com/auth/admin.directory.orgunit.readonly	**Org Units** - lets you view organizational units (read-only)
https://www.googleapis.com/auth/admin.reports.audit.readonly	**Admin Console Reports** - lets you view activities done by administrators using the Admin console and oAuth token activities (read-only)

Creating the API account in the Admin Console

To make API calls into the admin console, an API user account needs to be created. This account requires all of the specific API privileges in order to push and pull data from the Admin console.

1. Go to the admin console and sign in with an admin account with the necessary privileges to enable API access and create new user accounts (usually this is a super admin account).
2. Go to Admin Roles -> Create Role
3. Create a name for your API role and select desired privileges under “Organizational Units”, "Chrome Management" and then hit the Create Role button. See the table below for the specific rights that are needed:

Under **Organizational units** add (Read,Create,Update, Delete)
Under **Security Center** add (Investigation Tool>Chrome>View Metadata and Attributes, Investigation Tool>Admin>View Metadata and Attributes)
Under **Chrome Management** add (Managed Browsers, View Reports)

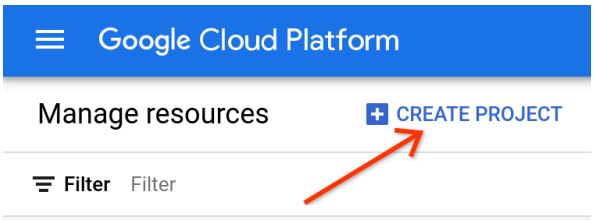
4. Create a user account that you will assign the API role to under Directory>Users>add new user
5. Give the API account a name and create an email address and click the add new user button.
6. Go to the user account that you created in the previous step and go to -> Admin roles and Privileges -> and select the role that you created in Step 3.
    a. Please note that propagation of this new permission may take a few minutes

Creating your project in console.cloud.google.com

1. **Open** console.cloud.google.com

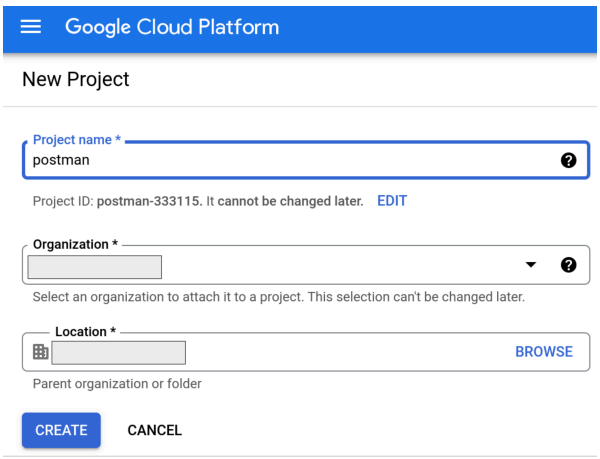
   Make sure that you are signed into the cloud console with the new API role account that you created in the previous section.

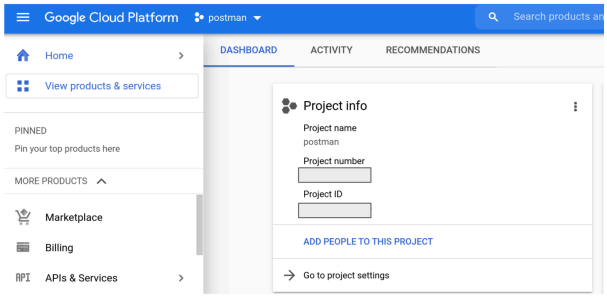
2. **Press the create project button.**



3. **Enter in the Project name**

   (can be whatever you choose) and the location should match the domain of your admin console. Hit create.

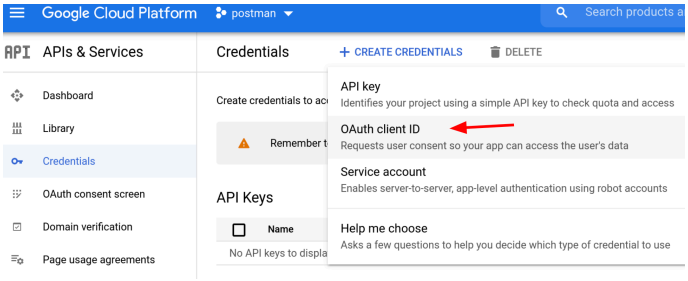




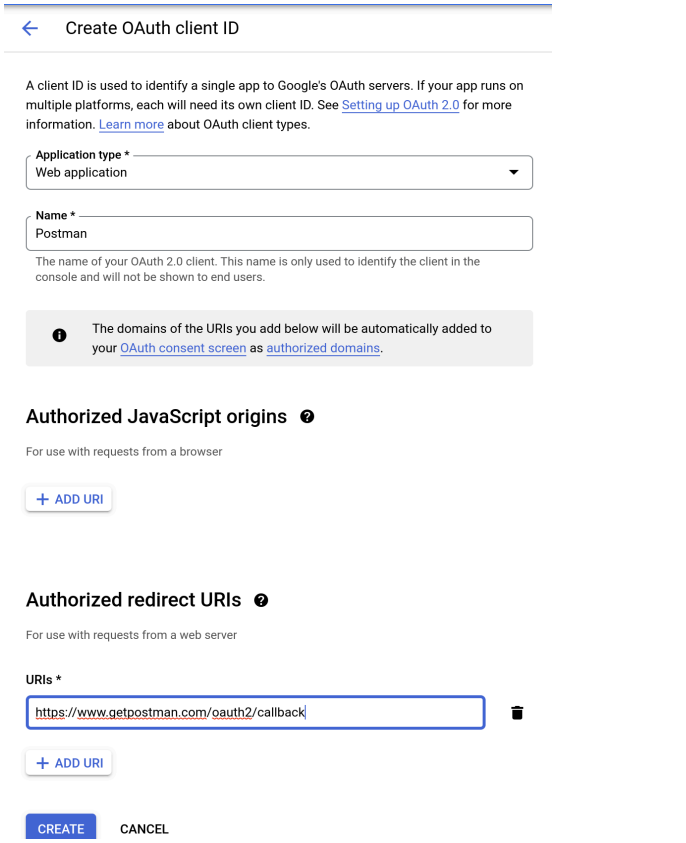
4. **Once created make sure that you have that project selected.**

Setting up Authorize with Consent .

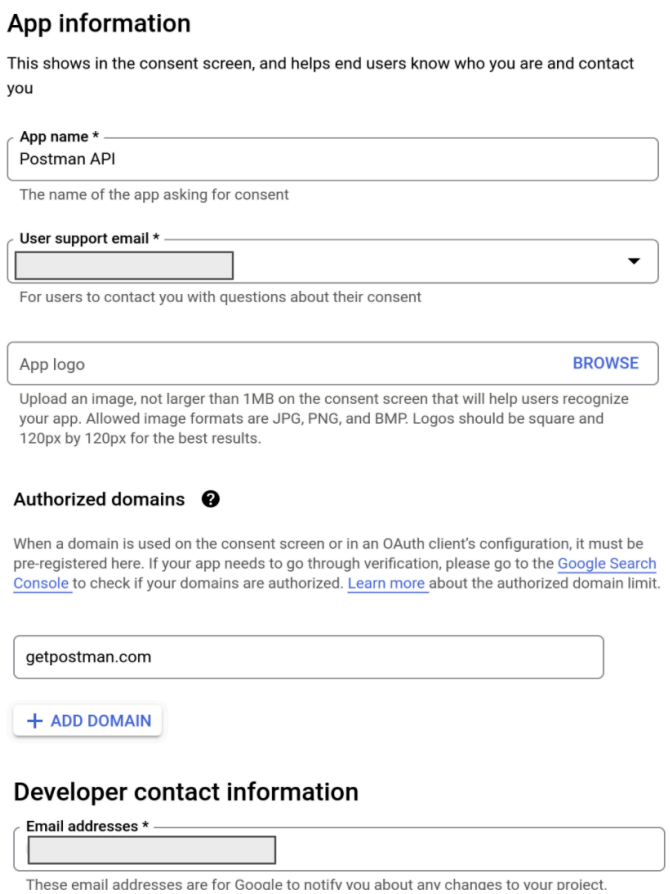
1. **Open console.cloud.google.com,** select the project that you created in the previous steps and browse to APIs and Services> Credentials>and hit the Create Credentials button and select OAuth client ID



2. **Select web application** for Application type web application and provide a name.
3. **For authorized redirect URI enter in the following:**
   a. https://www.getpostman.com/oauth2/callback
   b. https://oauth.pstmn.io/v1/browser-callback
4. **Hit the create button.**

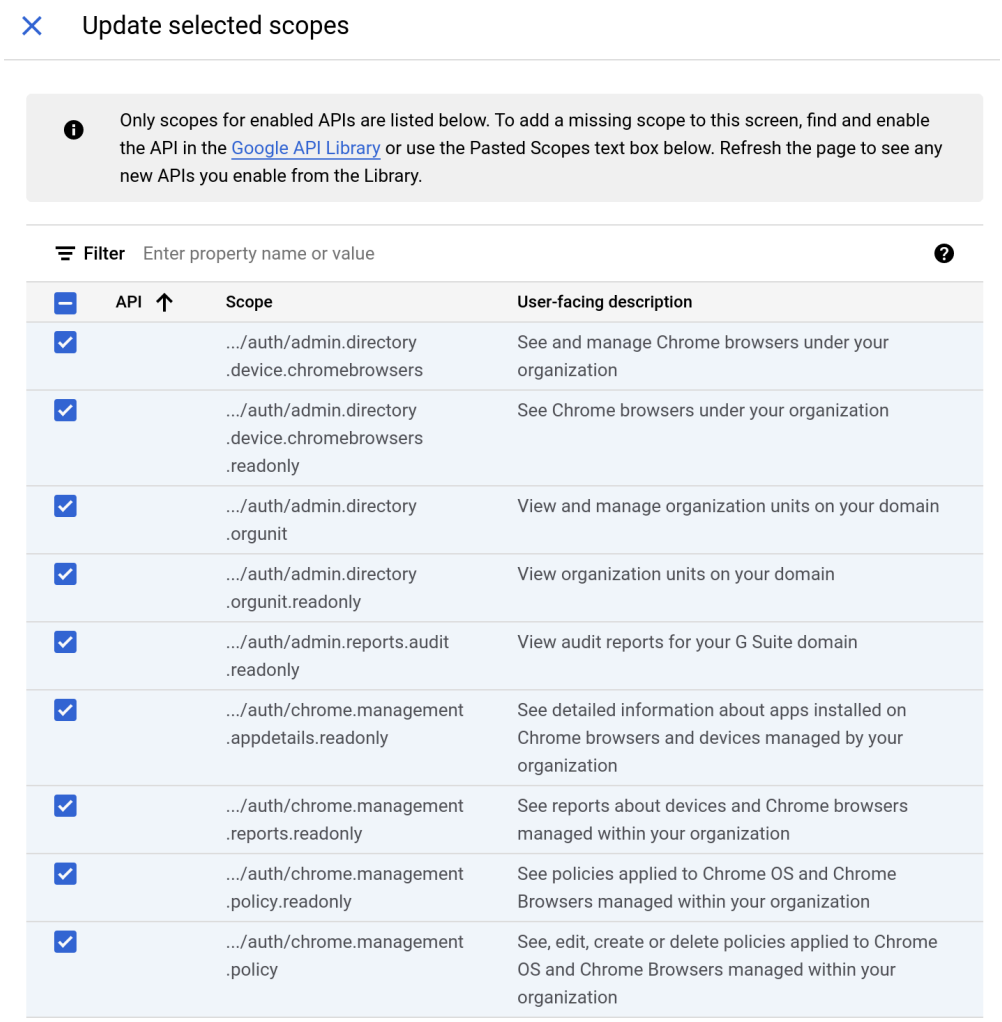


5. **On the OAuth client created window, click the download JSON button.**
6. **Click on the OAuth consent screenConfigure consent screen**
   When you use OAuth 2.0 for authorization, your app requests authorizations for one or more scopes of access from a Google Account. Google displays a consent screen to the user including a summary of your project and its policies and the requested scopes of access.
   There are two types:
   a. **Internal** is just for users within your Google Workspace users. Requires Google Workspace or Google Identity to function.
   b. **External** is available to any test user with a Google Account. Your app will start in testing mode and will only be available to users you add to the list of test users. Once your app is ready to push to production, you may need to verify your app.
7. **Enter in the App name,** the user support email (this is your admin account email in the admin console) and you can also choose to display a custom logo.
8. Under Authorized domains enter in **getpostman.com**
9. **Enter in an email address** that you want Google to notify you about changes in your project and hit save and continue.



10. **Click the Add or Remove Scopes button** to add the specific scopes manually. They are all listed in this [list of scopes in the Chrome Browser Enterprise Github](). Hit save and continue.

    Note that the list contains both read only and full access. Pick and choose which scopes you need for your specific use case.

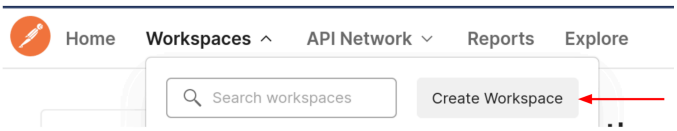


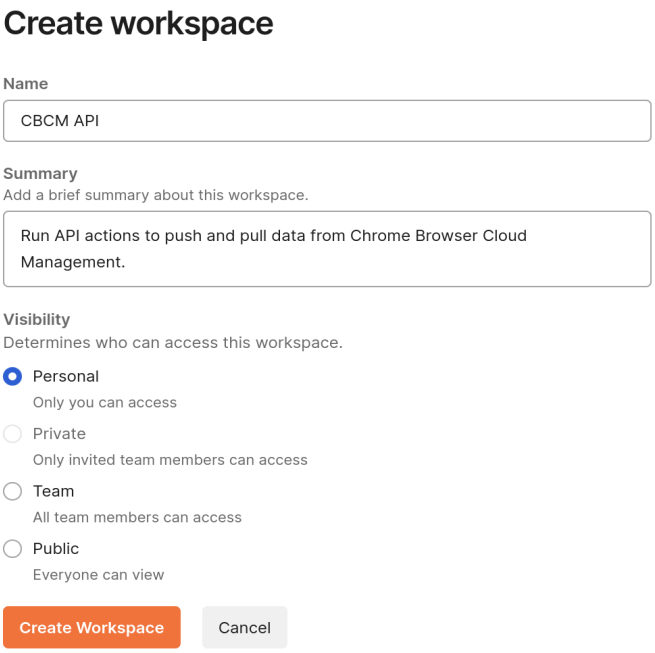
11. **Optional** - Add a user(s) that can access while the publishing status is listed as “Testing”. Enter in the account that you used to create the project and any others that you want to provide access to by hitting the add users button.
12. **Hit save and continue**, review the summary screen and hit the back to dashboard button.
13. If you like you can **publish your App.**
    a. Once you set your app status as "**In production,**" your app will be available to anyone with a Google Account. Depending on how you configure your OAuth screen, you may have to submit your app for verification

Setting up Postman integration

Setting up an account

1. **Go to** Postman.com and set up a new account or use an existing one.
2. Once your account is setup, **go to Workspaces>Create Workspace** and give it a name and select the level of visibility you want and hit the Create Workspace button



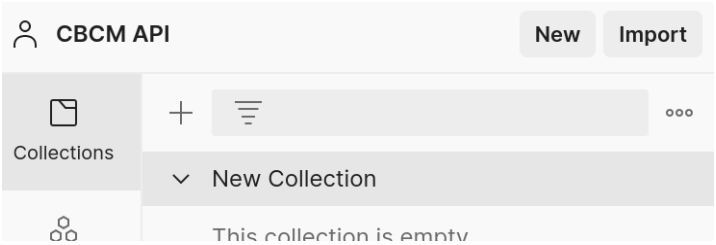


Importing a collection from the Chrome Enterprise Github

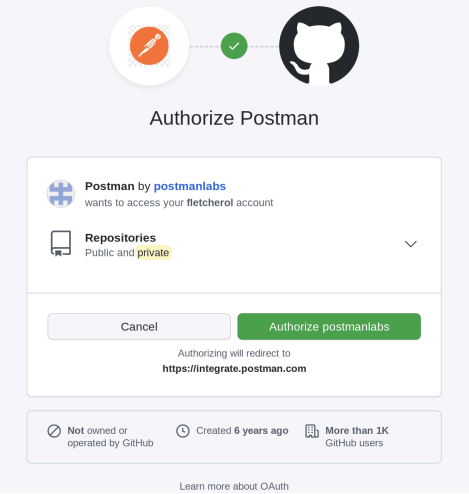
It is recommended to import the collections that are provided in the Chrome Enterprise Github. The provided collections provide sample scripting for common use cases for calls to Chrome Enterprise Core. You can import them via the file method or link to your Github repository.

Importing via Code repository

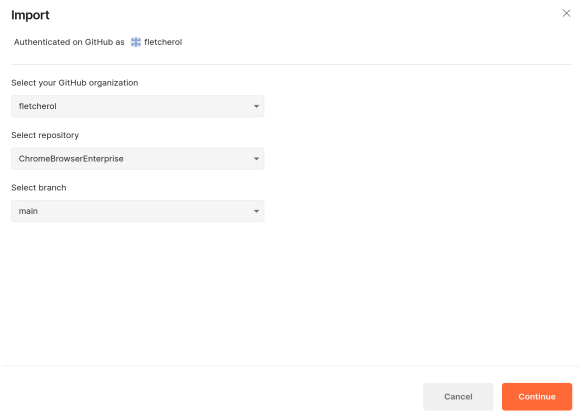
1. **Log into your Github account.**
2. **Go to** https://github.com/google/Chrome BrowserEnterprise and hit the fork button to add it to your repository.
3. In Postman with your Workspace selected, **click on the import button.**



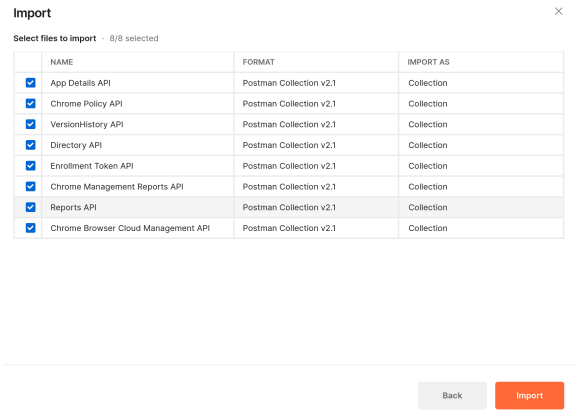
4. **Click on Code repository> Github**
5. **Log into your Github account** and click on the authorize postmanlabs button and confirm your password.



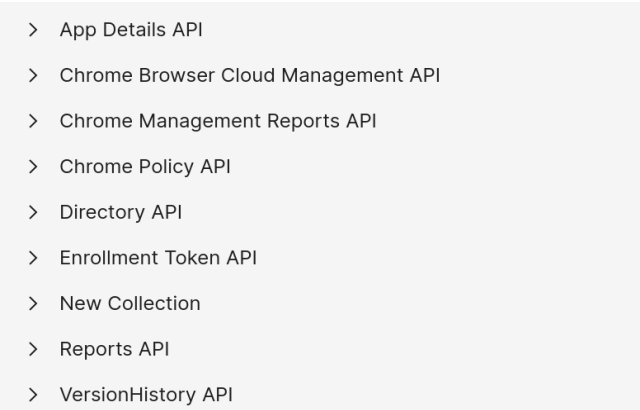
6. Once this is complete in the import section you should see the ChromeBrowserEnterprise selectable in the dropdown menu under Select repository. **Hit the continue button.**



7. **Select which collections you want to import and hit the import button.**

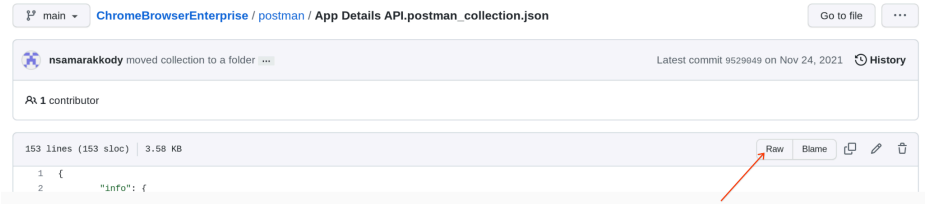


8. Once complete you should see all of the collections included in your Postman workspace

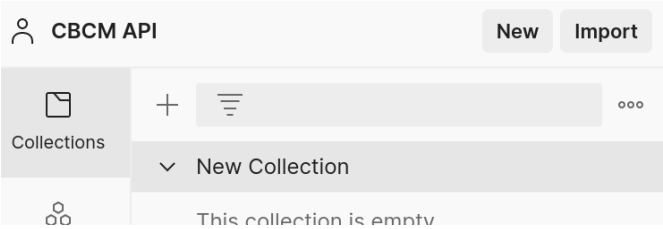


Importing via Upload files

1. Browse to the Chrome Enterprise Github and **select the collection that you want to import.**
2. **Click on the raw button**

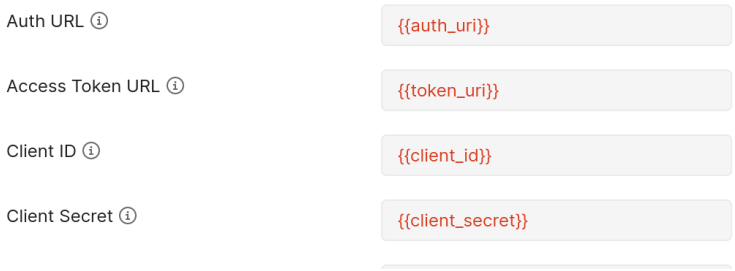


3. **Right click and save as .json.**
4. In Postman with your Workspace selected, **click on the import button.**

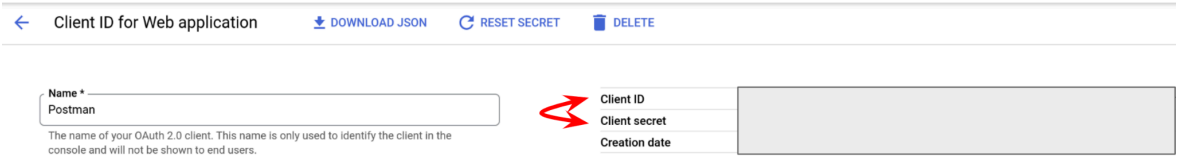


5. **Hit the Upload Files button** and select the JSON that you downloaded in the previous step and hit the import button

6. Once the Collection shows up, you will need to fill in the sections highlighted in red.



7. **For Auth URL enter in the following:** https://accounts.google.com/o/oauth2/auth
8. **For Access Token URL enter in the following:** https://oauth2.googleapis.com/token
9. Get the client id and client secret from the API and services section of https://console.cloud.google.com/(direct link) and **enter them into Postman.**



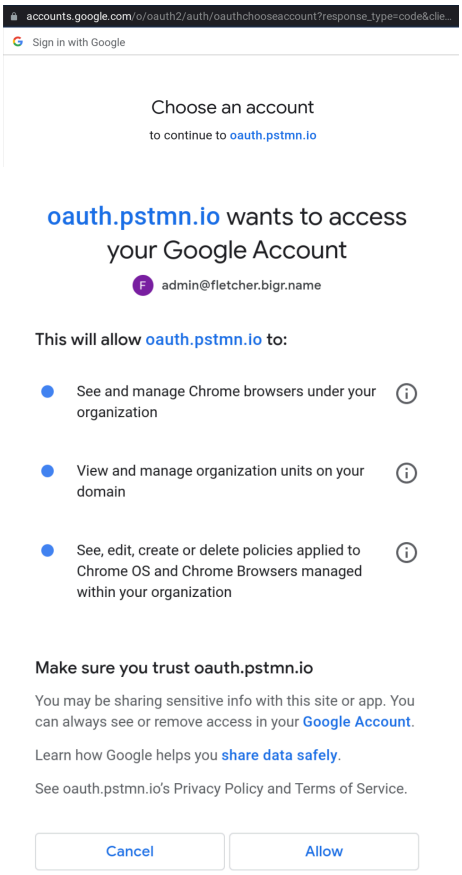
   a. Note that it is recommended to add in these values as variables to keep sensitive data secure and make it easier to add the data to multiple collections as you import them.

   b. If you want to use the same variables in multiple collections you need to declare them as global variables.

   More information about Postman variables is located here.

10. **Get the client id and client secret** from the API and services section of https://console.cloud.google.com/(direct link) and enter them into Postman.
11. If you are using the browser version of Postman, the callback URL this is prefilled in, on the desktop version you might need to **input to put the following:** https://www.getpostman.com/oauth2/callback
12. **Hit the Get New Access Token button.**
13. When you hit the get new access token button, you will be prompted to authenticate using your admin credentials for Chrome Enterprise Core.

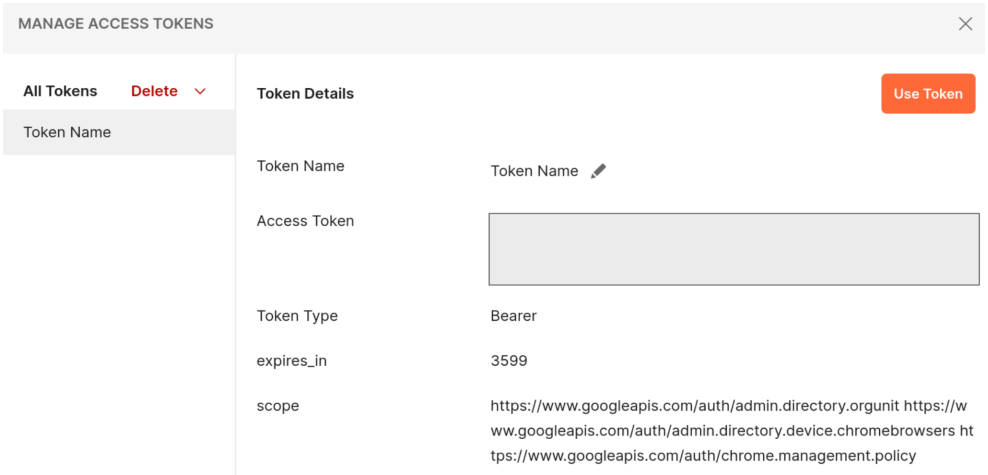
    Make sure that this account has API privileges granted for the scopes that you have added.



14. Once completed you will be presented with your new token that you will be using to authenticate requests to the APIs.

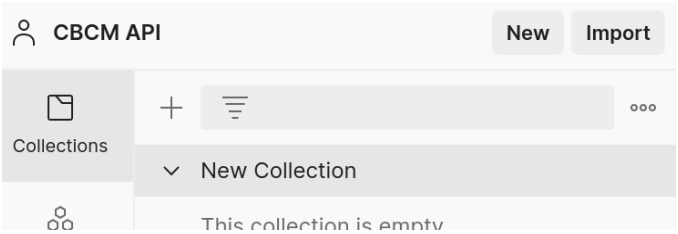
    Hit the use token button and the token information will be prefilled into the collection for you.

    Tokens are valid for one hour by default.



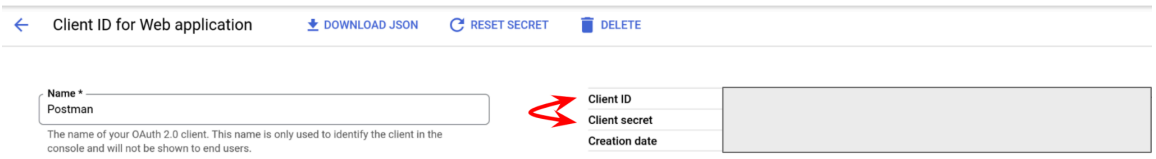
Creating a new collection

1. **Under the Collections tab hit the plus mark to create a new collection**



The following steps are only needed for the Authorize with Consent method only:

2. **Select Oauth2 for the type of Authorization**
3. **For the callback URL enter in the following:**
    a. If using the browser version of Postman this is prefilled in, on the desktop version you need to **put the following:** https://www.getpostman.com/oauth2/callback
4. **For Auth URL enter in the following:** https://accounts.google.com/o/oauth2/auth
5. **For Access Token URL enter in the following:** https://oauth2.googleapis.com/token
6. Get the client id and client secret from the API and services section of https://console.cloud.google.com/(direct link) and **enter them into Postman.**



Note that it is recommended to add in these values as variables to keep sensitive data secure and make it easier to add the data to multiple collections as you import them.

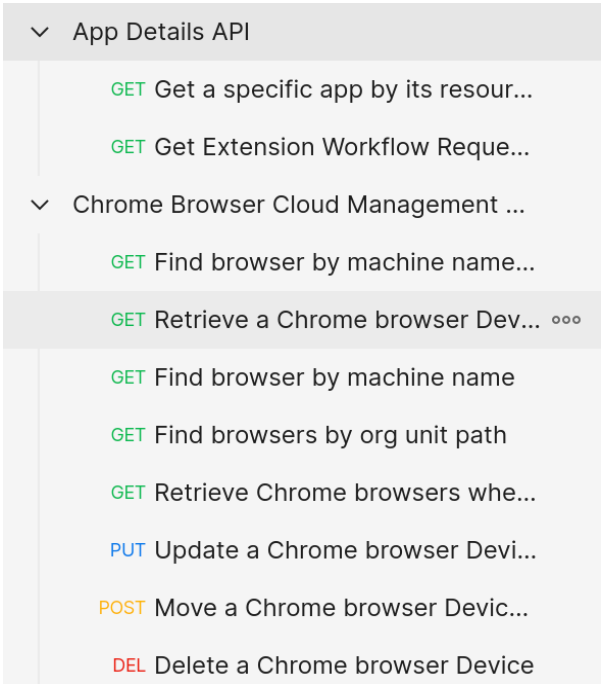
More information about Postman variables is located here.

7. **Add in the Cloud management API specific scope(s) that you want** (reference this section for more information), each scope needs to be separated by a space.
8. **Hit the Get New Access Token button.**

Verifying the connection to Chrome Enterprise Core

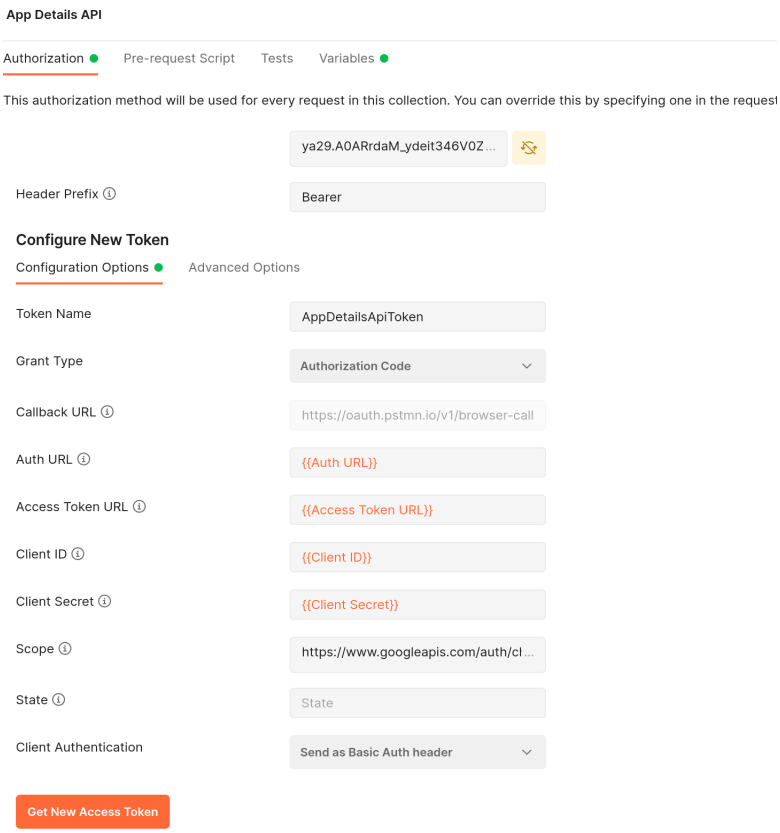
Each collection that is present in the Chrome browser Enterprise Github have some example scripts to get you started, with examples for GET, PUT, POST and DEL requests. Once you have imported the collection, you can run one of these to make sure that the Postman is able to make requests to Cloud management.

1. **Click on the collection that you imported.**
   The following steps (2-6) are only needed for the Authorize with Consent method only:

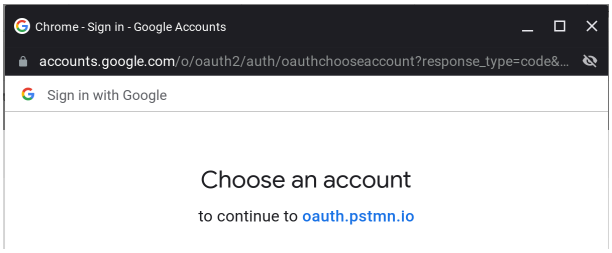


2. **Confirm** that the Auth URL, Access Token URL, Client ID and Client Secret and scope sections are filled in either with the values or with a variable.

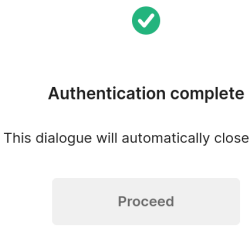
3. Click on the Get New Access Token button



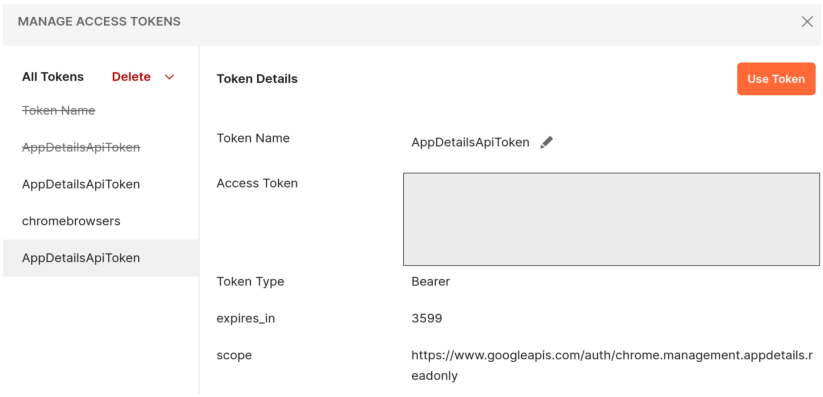
4. Choose the account that you want to auth to



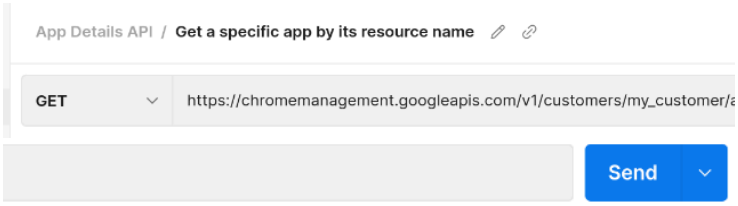
5. You will receive the authorization complete dialog if all is correct



6. **Click on the Use Token button.**



7. **Select the request that you want to make and click on the send button**



8. The response will show down below and you can change from JSON to other formats if you wish via the dropdown menu.