Setting Up Google Drive for Unreal Turnkey

How to set up the Google Drive API and host SDKs for use with Unreal Turnkey

Setting up **Google Drive** for **Unreal Turnkey** is more complex than using Perforce or a local repository, but it provides a common place to store and maintain SDK files once the initial setup is complete. To do this, you will need to set up the following resources for your organization:

- A Google account for your organization with Google Drive enabled.
- A **Google Cloud Platform** app with **Google Drive API** enabled, set up with **OAuth 2.0 credentials**.
- Turnkey manifest files pointing to the Google Drive folder and the required credentials.

This document will guide you through the process of setting up these resources for your organization.

1. Required Setup

This guide assumes that you have a Google Drive account set up for your organization. For information on how to set one up, refer to Google's documentation.

This guide also assumes that you have generated the required SDK packages for your supported platform, and that they are contained in a .zip file. Each platform has different methods for creating a full SDK or creating a flash SDK. Some packages will already be zipped when you receive them, in which case you can upload them directly to Google Drive without any changes. However, some consoles' SDKs require extra steps.

For information on how to generate the required files for a full or flash SDK from files received from your SDK provider, refer to the **Help** command in the Turnkey commandline. This command will output instructions on how to configure your SDKs as well as the proper version numbering conventions.

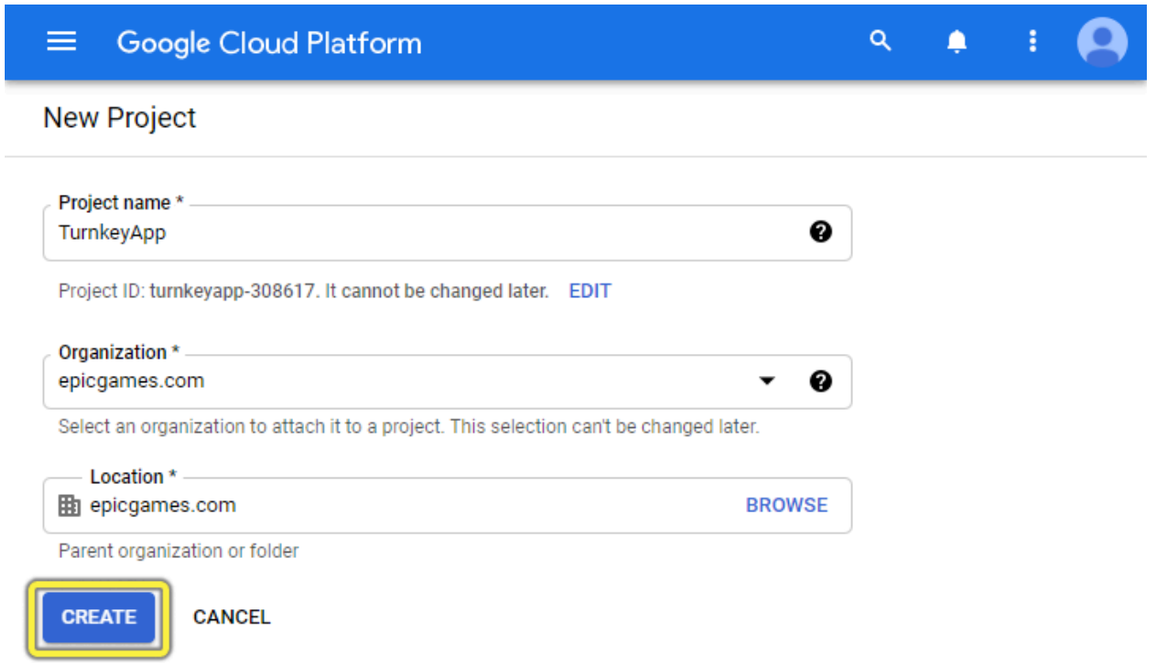
While there is no required naming convention for these .zip files, we recommend you choose a consistent compression format and naming convention to make them easy to reference. We also highly recommend zipping your SDK files before uploading them to Google Drive, as download requests through Google Drive perform significantly faster for single, compressed files than they do for large numbers of files.

2. Setting Up Google Drive API

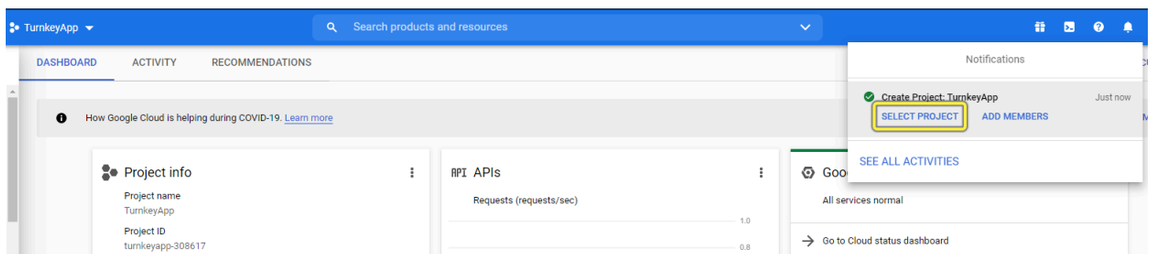
You need to set up an app with Google Drive API enabled to permit users to access your Google Drive folder and its files. You also need to set up secure credentials so that only authorized users can access your Drive folder.

This section will walk you through the process of enabling the API and OAuth 2.0 credentials:

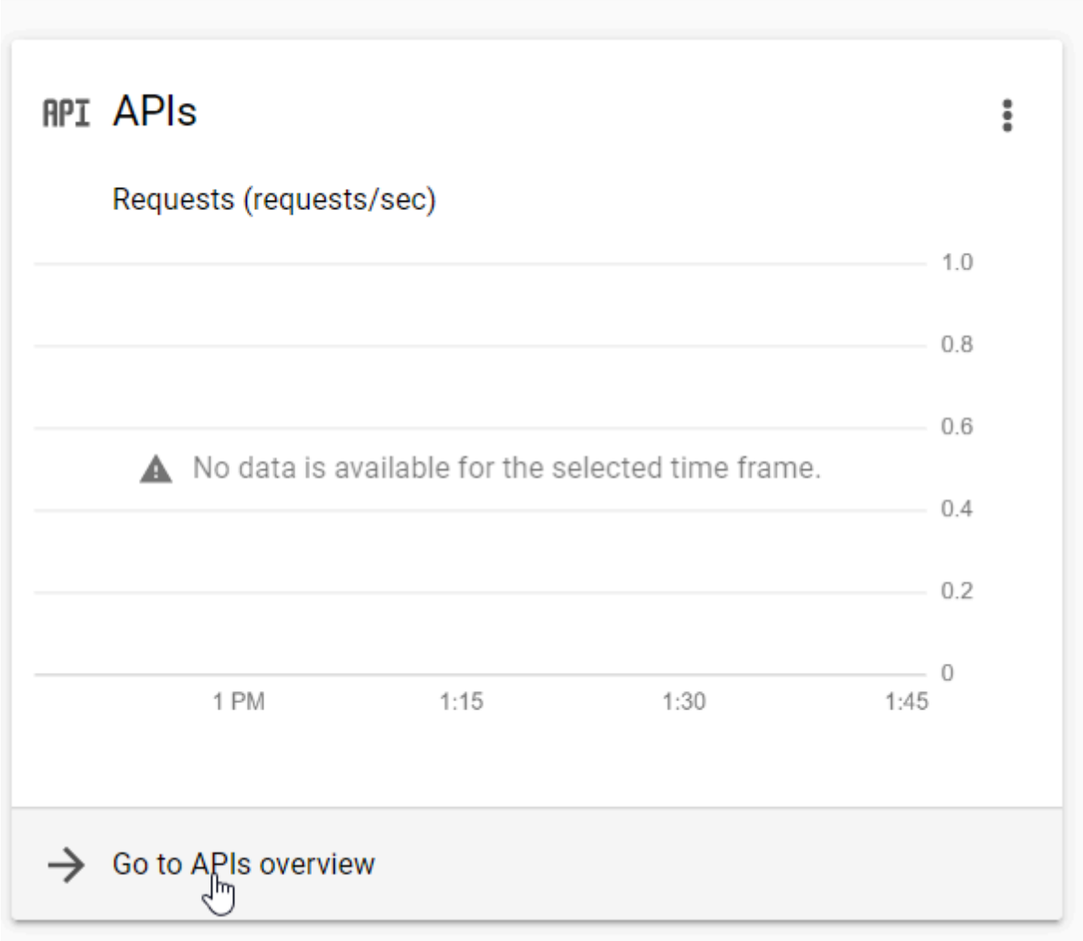
1. Visit the Google Developer Console at https://console.developers.google.com/projectcreate and **agree** to the Terms of Service.
2. Fill out the **project name**, **organization**, and **location** fields, then click **Create**.



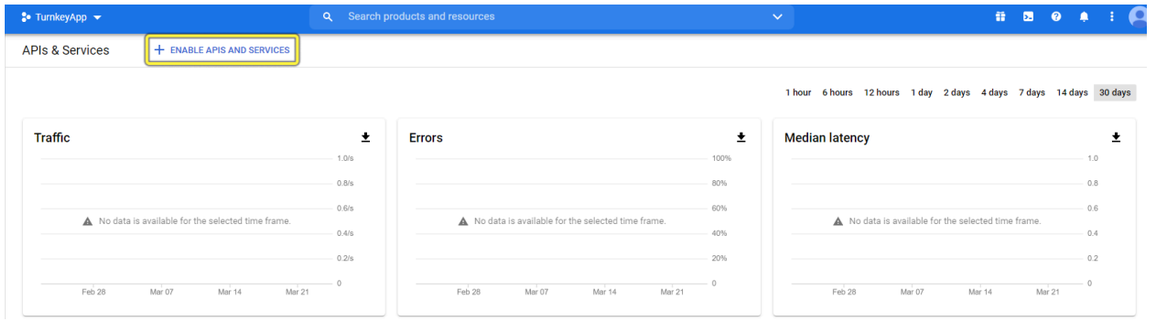
3. You will be navigated to the **Google Cloud dashboard**, and a **notification** will pop up confirming that your project is set up. Click **Select Project** in this notification.



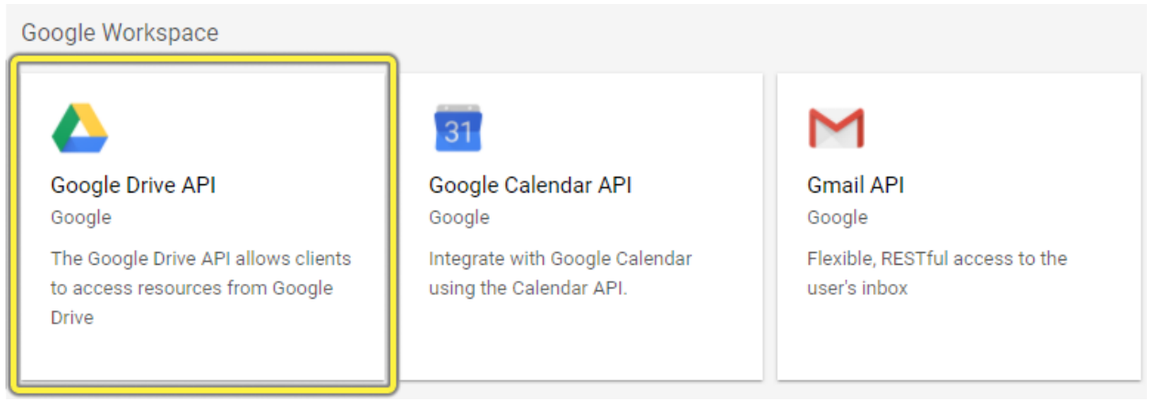
4. In the **APIs** panel, click **Go to APIs overview**. This will open the **APIs & Services** page.



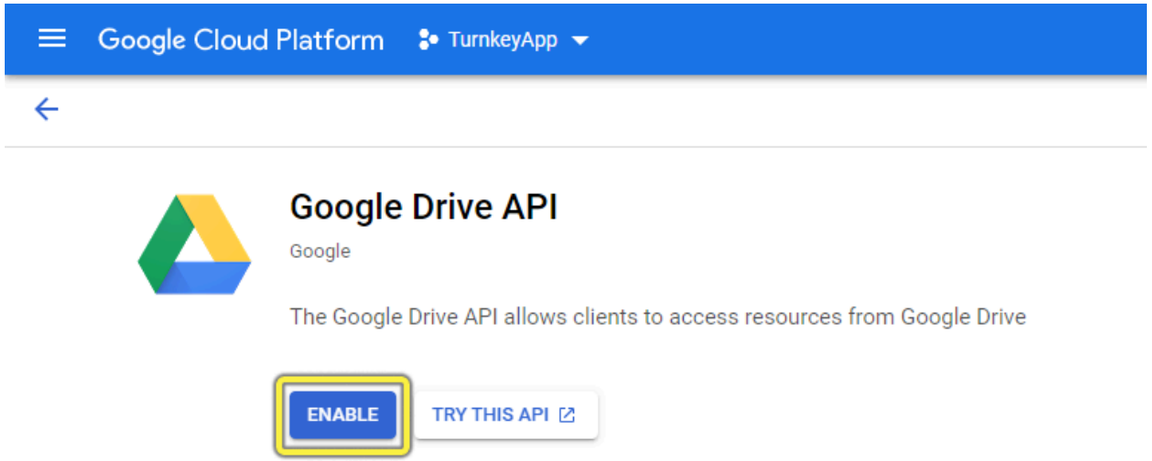
5. At the top of the APIs & Services page, click **Enable APIs and Services**. This will open the **API Library**.



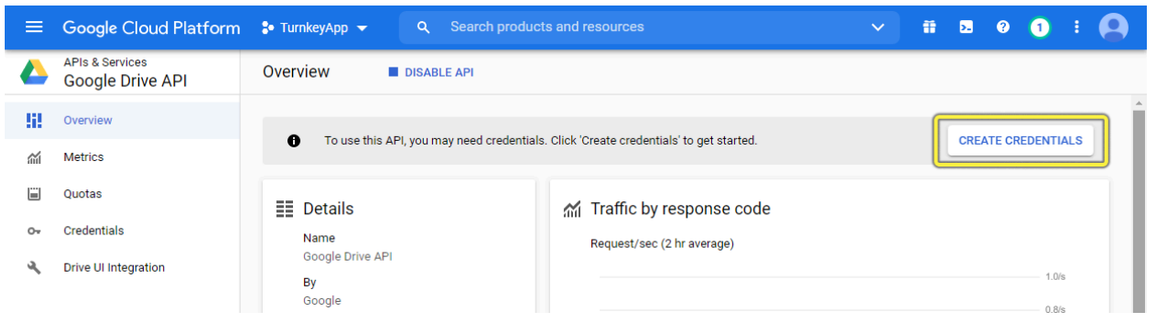
6. Choose the **Google Drive API**. You can find it under the **Google Workspace** section.



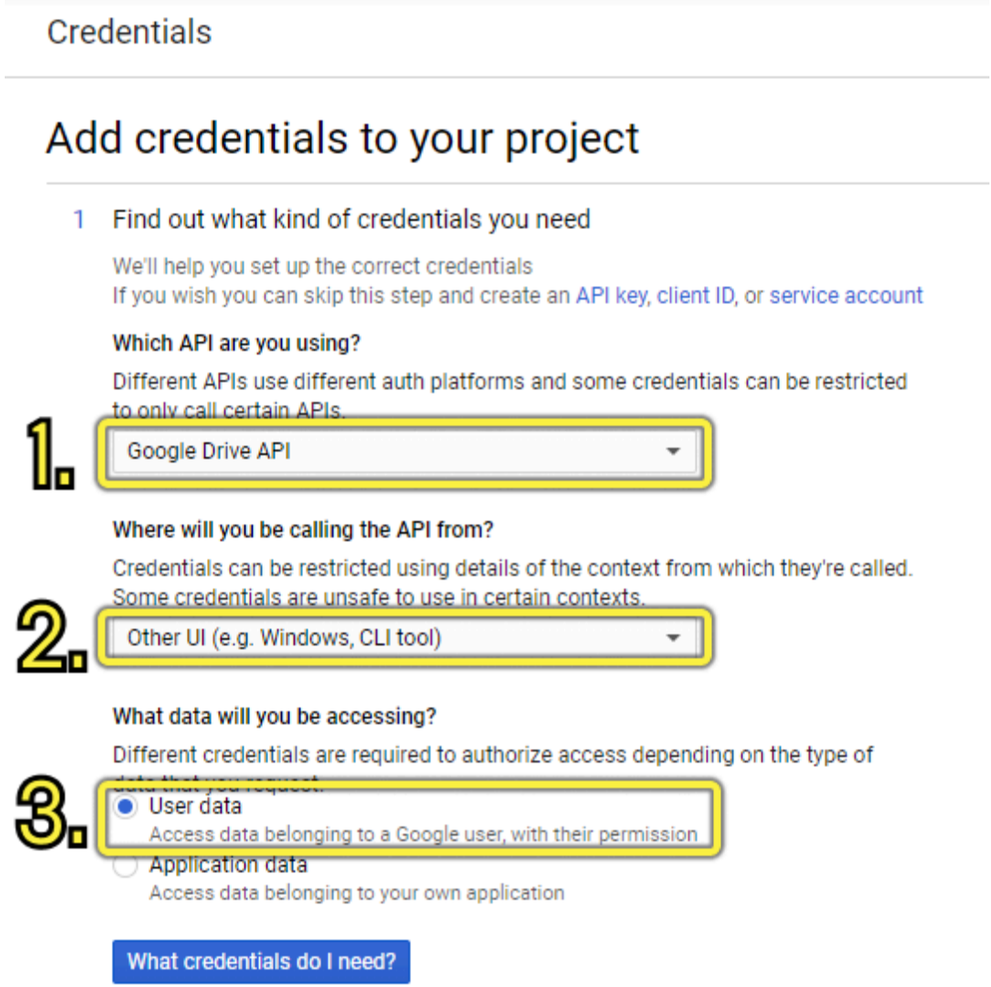
7. Click the **Enable** button for this API.



8. The APIs & Services page for the Google Drive API will open. Click the **Create Credentials** button. This will open a form to add credentials to your project.



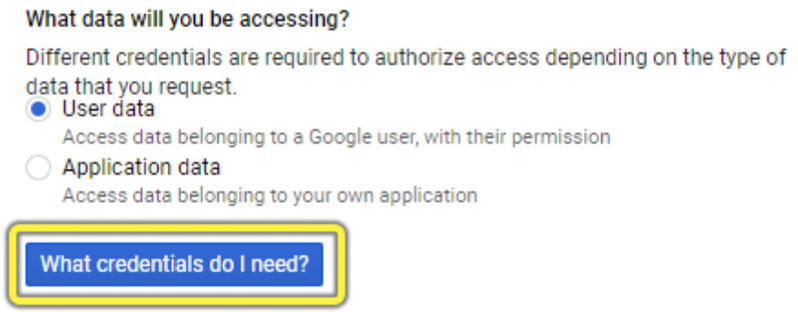
9. Fill out the form with the following settings:



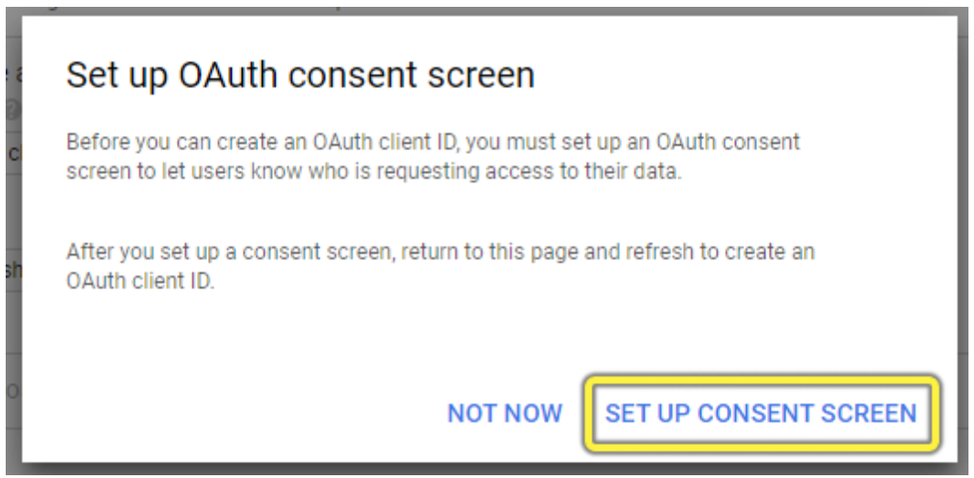
Number	Setting	Value
1	Which API are you using?	Google Drive API
2	Where will you be calling the API from?	Other UI

Number	Setting	Value
3	What data will you be accessing?	User Data

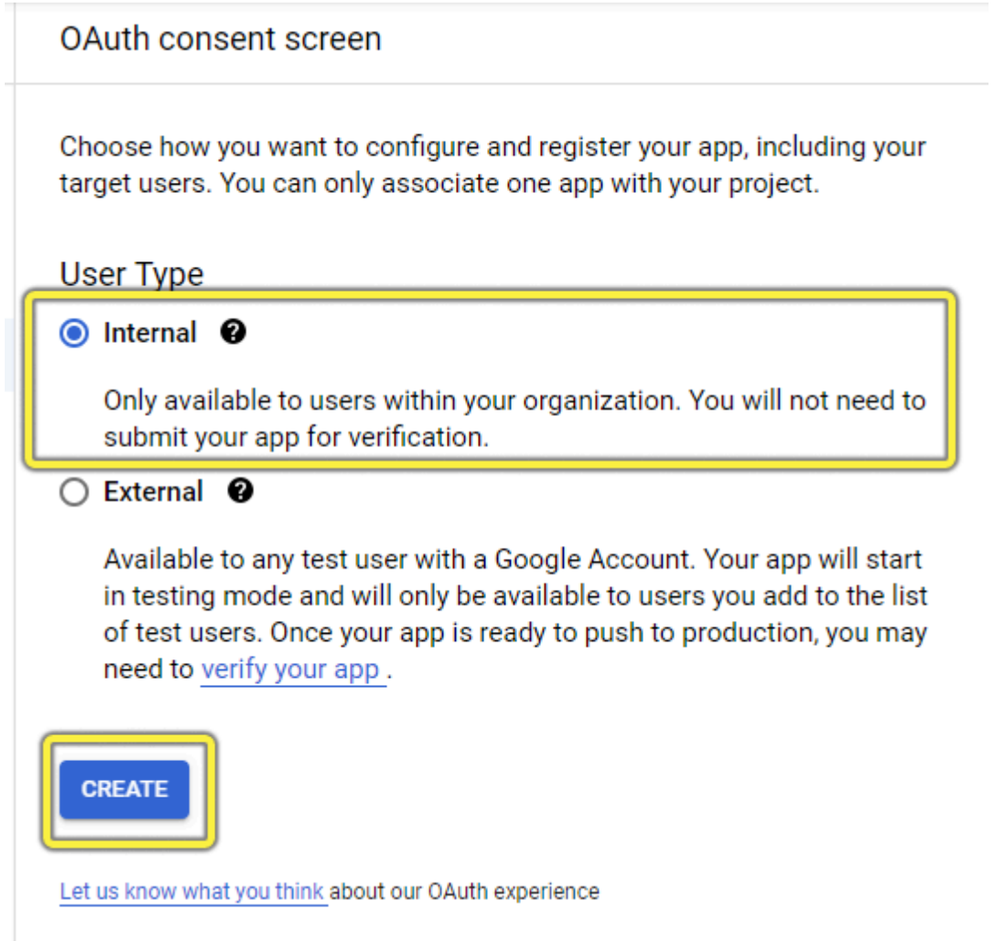
10. Click the button labeled **What credentials do I need?**



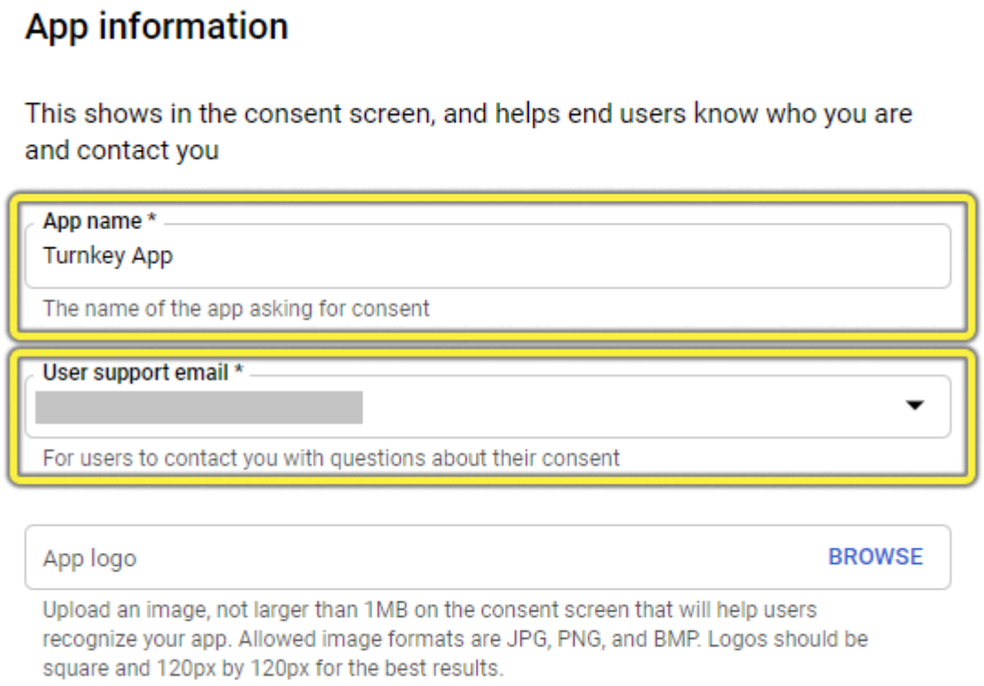
11. A popup will appear prompting you to set up an OAuth Consent Screen. Click the option to **Set Up Consent Screen**.



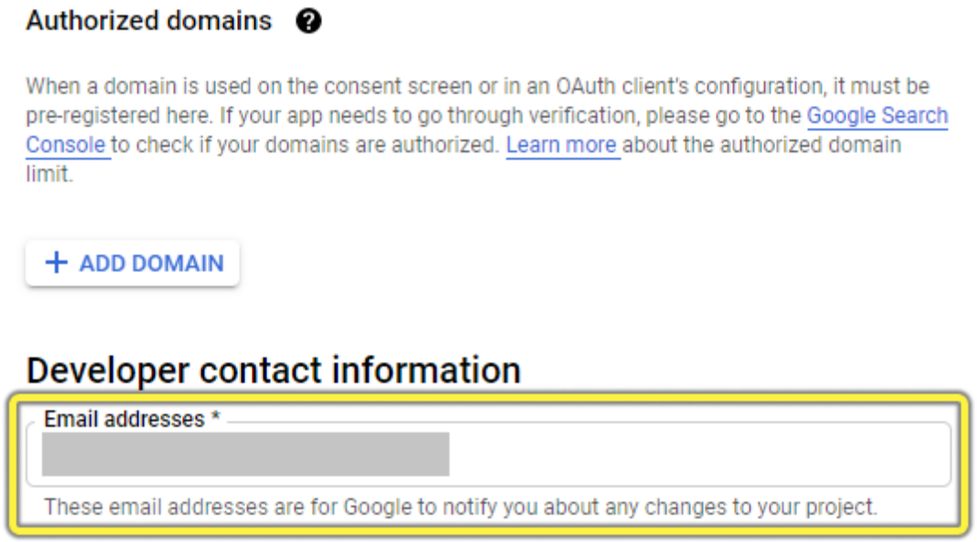
12. This will open a new tab for configuring OAuth credentials. Under **User Type**, select **Internal**, then click **Create**.



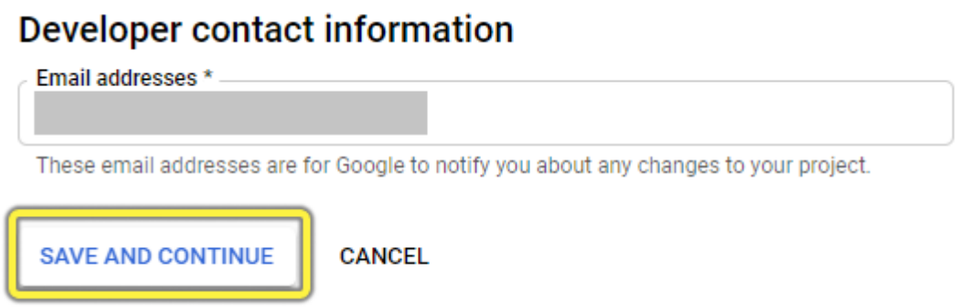
13. Another form will open for editing your app registration. Under **App Information**, enter the **Application Name** and **User support email** for your application. The user support Email should be a point of contact members of your organization can use to troubleshoot issues.



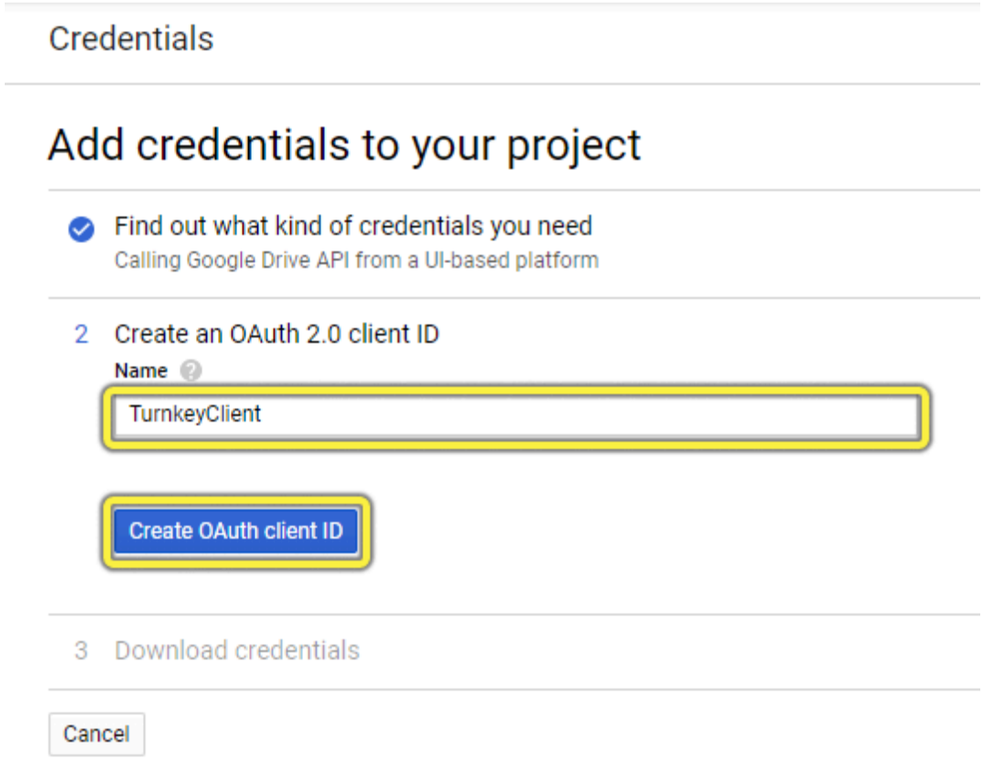
14. Scroll down to **Developer Contact Information** and enter the Email addresses for whomever is managing your organization's Turnkey Drive folder.



15. Click the **Save and Continue** button.



16. Close the current tab and return to the **Credentials** page, then click the **Refresh** button. Give your credentials a **name**, then click **Create OAuth Client ID**.



17. Once this has been created, the **Download credentials** section will become available. Click the **Download** button to download a file named `client_id.json`, which will be needed to provide your users with access to your organization's Google Drive.

The credentials you have set up will allow an application to access your shared drive and download files from it.

3. Providing Google Drive Credentials

To discover files on your Google Drive, you will need to provide the security credentials to access it. This information is contained in a `TurnkeyManifest.xml` file under its `<Studio_GoogleDriveCredentials>` setting.

1. Move the `client_id.json` file you generated in the previous section into your engine installation's `Engine/Build/Turnkey` folder. Be sure to add the file to your organization's version control system to make it available for your users.
2. Open the `TurnkeyManifest.xml` file located inside this folder and add the following line to the top of the file:

```
<Studio_GoogleDriveCredentials>$(ThisManifestDir)/client_id.json</Studio_GoogleDriveCredentials>

```

Copy full snippet

3. Save the manifest and update it on your organization's version control system.

Your application can now connect to your Google Drive using the provided credentials.

> The first time a user connects to your Google Drive, it will open a web page to authenticate the user. During this time, Turnkey will pause and display a message indicating that it is waiting for authentication, but you may need to check for the new browser window manually. After completing this step once, Google Drive will cache the user's login information for future use.

4. Uploading Your SDKs to Google Drive

Now that you have generated and assigned the credentials needed to access your Drive, you need to set up the Google Drive folder structure to store your files.

1. Create a shared drive for your organization named **SdkInstallers**.
2. Inside this drive, create folders for each platform you will support. Make sure the names of these folders match the names listed in your `TurnkeyManifest.xml` manifest file.
3. Create a subfolder for each SDK version you want to host within each platform's folder.

4. For each SDK version, create a .zip file named `Install.zip`. Place this file into its corresponding platform and version folder on your Google Drive.

Once these steps are completed, your SDKs are ready for distribution within your organization.

> If you are using automatic file discovery with the `$fileexpansion:` prefix and the file expansion variable `$[ExpVersion]`, make sure your version folder names match the version numbering convention used by the corresponding platform. You can check this information by using the Help command in the Turnkey commandline.

5. Creating Manifest Files

Finally, you need to edit your Turnkey manifest to point to the files on your Google Drive.

1. Open the `TurnkeyManifest.xml` located in your engine installation's `Engine/Build/Turnkey` folder.
2. Add the relevant `<FileSource>` information for the SDKs you have available. For example, the following would be a valid `<FileSource>` for `TurnkeyManifest_Win64.xml`:

```
<FileSource>
<Platform>Win64</Platform>
<Type>Full</Type>
<Version>$(ExpVersion)</Version>
<Name>Win64 v.($ExpVersion)</Name>
<Source>fileexpansion:googledrive:/SdkInstallers/Win64/$[ExpVersion]/Install.zip</Source>
</FileSource>

```

Copy full snippet

Because the Source path includes `fileexpansion:` and the `$[ExpVersion]` capture variable, Turnkey will look through all available version folders within the Win64 folder and create a FileSource object for each valid version number. Each of these FileSource objects will substitute `$(ExpVersion)` with the version number found.

Refer to Setting Up Turnkey for Your Organization for more information on how to format manifests.

3. Upload each manifest to its corresponding platform folder on your SdkInstallers drive.
4. Make sure the base `TurnkeyManifest.xml` file in `Engine/Build/Turnkey` is saved and updated on your organization's version control system so that every user's engine installation has access to it.

   Once your manifests have been created, your SDK files will be discoverable on Google Drive when Turnkey starts.

6. Final Result

After following this guide, your organization is now equipped with a shared drive where you can upload and access platform SDKs. Your organization's administrators can add or remove SDKs and edit manifests, while Unreal Engine users will be able to access them by making a request to Turnkey to install or update SDKs.