**Google Cloud Storage Setup Procedure**

**for Laravel Platform**

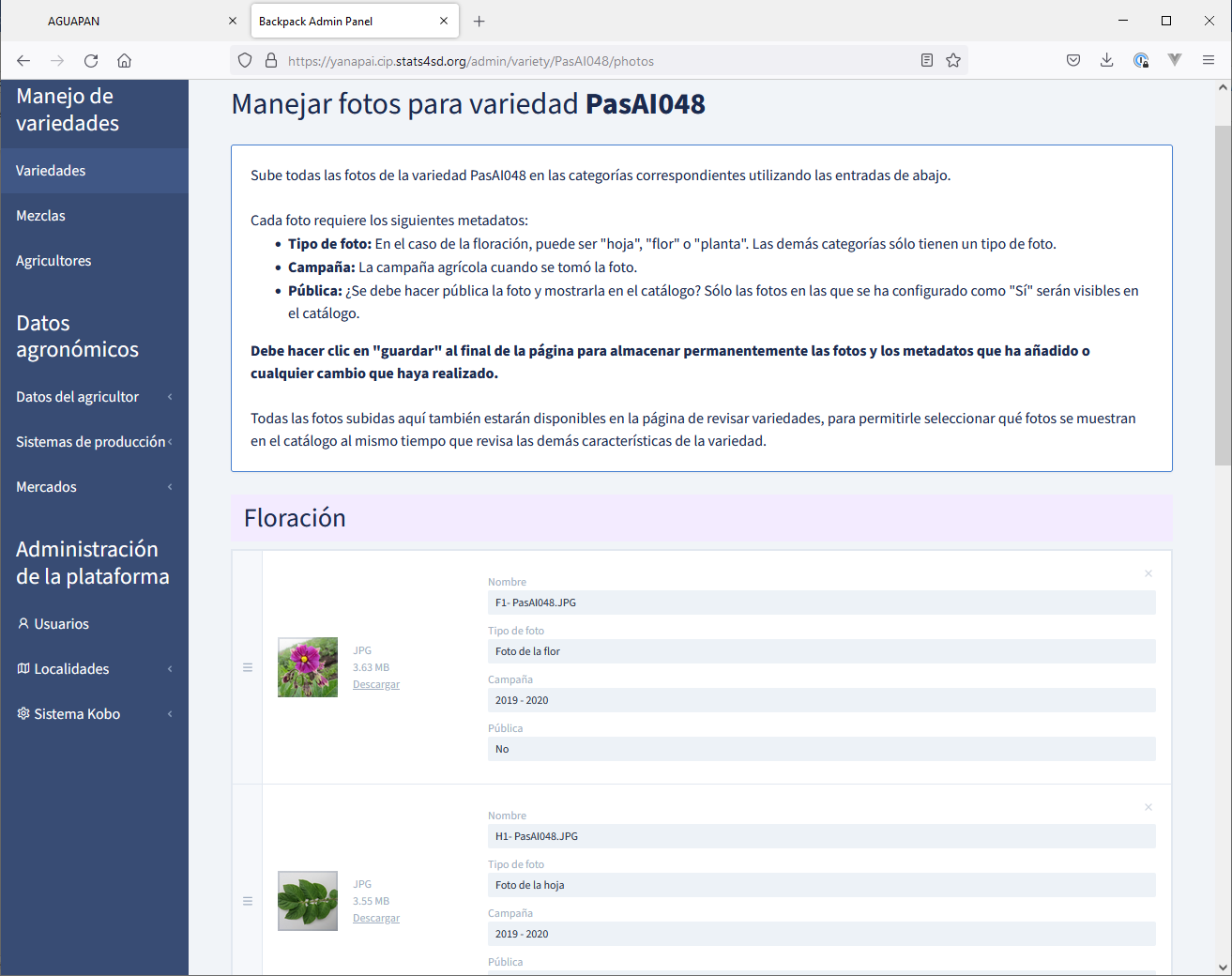
**Introduction**

Stats4SD has utilized Google Cloud Storage and Spatie Media Library for file upload functionality. We facilitate variety photos upload and farmer photos upload for Yanapai potato platform. And we allow user to upload any file for further research activities in TPP case studies platform.

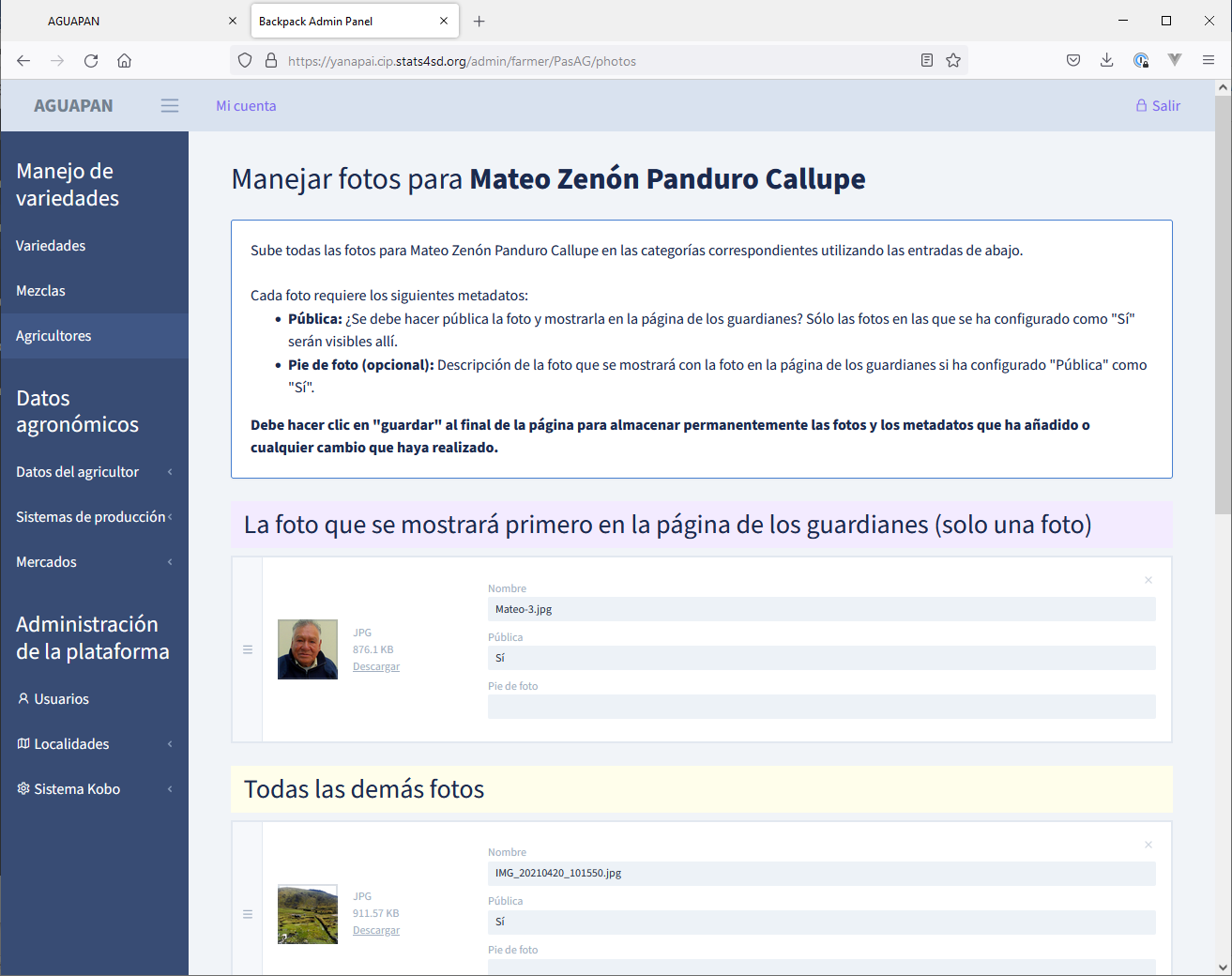
Google Cloud Storage provides a cloud-based storage space with low cost and high availability.

Spatie Media Library (<https://spatie.be/docs/laravel-medialibrary>) provides a streamlined front end user interface for uploading files. It provides a wide range of features including the configuration for accepted file format, maximum file size, single file upload or multiple file upload, generate image thumbnails, adding tags to uploaded file, etc.

Yanapai Potato Platform – Variety > Manage Photo Page



Yanapai Potato Platform – Farmer > Manage Photo Page



TPP Case Studies Platform – Team > Step 8 Further Research > Research Activity > View Files

Graphical user interface, application

Description automatically generated

TPP Case Studies Platform – Team > Step 8 Further Research > Research Activity > Manage Files PageGraphical user interface, text, application, email

Description automatically generated

**Google Cloud Storage Setup Procedure**

This document illustrates the setup procedure for Google Cloud Storage. For Spatie Media Library, please visit their official web site (<https://spatie.be/docs/laravel-medialibrary>) for installation and setup procedure.

In Google Cloud Storage, file storage bucket is created with below hierarchy:

Google login account > Project > Bucket

**Login Google Cloud Console**

1. Visit Google Cloud Console (<https://accounts.google.com/ServiceLogin/signinchooser?service=cloudconsole&passive=1209600&osid=1&continue=https%3A%2F%2Fconsole.cloud.google.com%2Fstorage%2Fbrowser%3Fproject%3Dtpp-case-studies%26prefix&followup=https%3A%2F%2Fconsole.cloud.google.com%2Fstorage%2Fbrowser%3Fproject%3Dtpp-case-studies%26prefix&flowName=GlifWebSignIn&flowEntry=ServiceLogin>)
2. Login with account “support@stats4sd.org”, the login password can be found in 1Password item “Google – For Gcloud stuff”
3. After login, it shows 3 buckets created for project “TPP Case Studies”

Graphical user interface, application

Description automatically generated

TPP Case Studies project

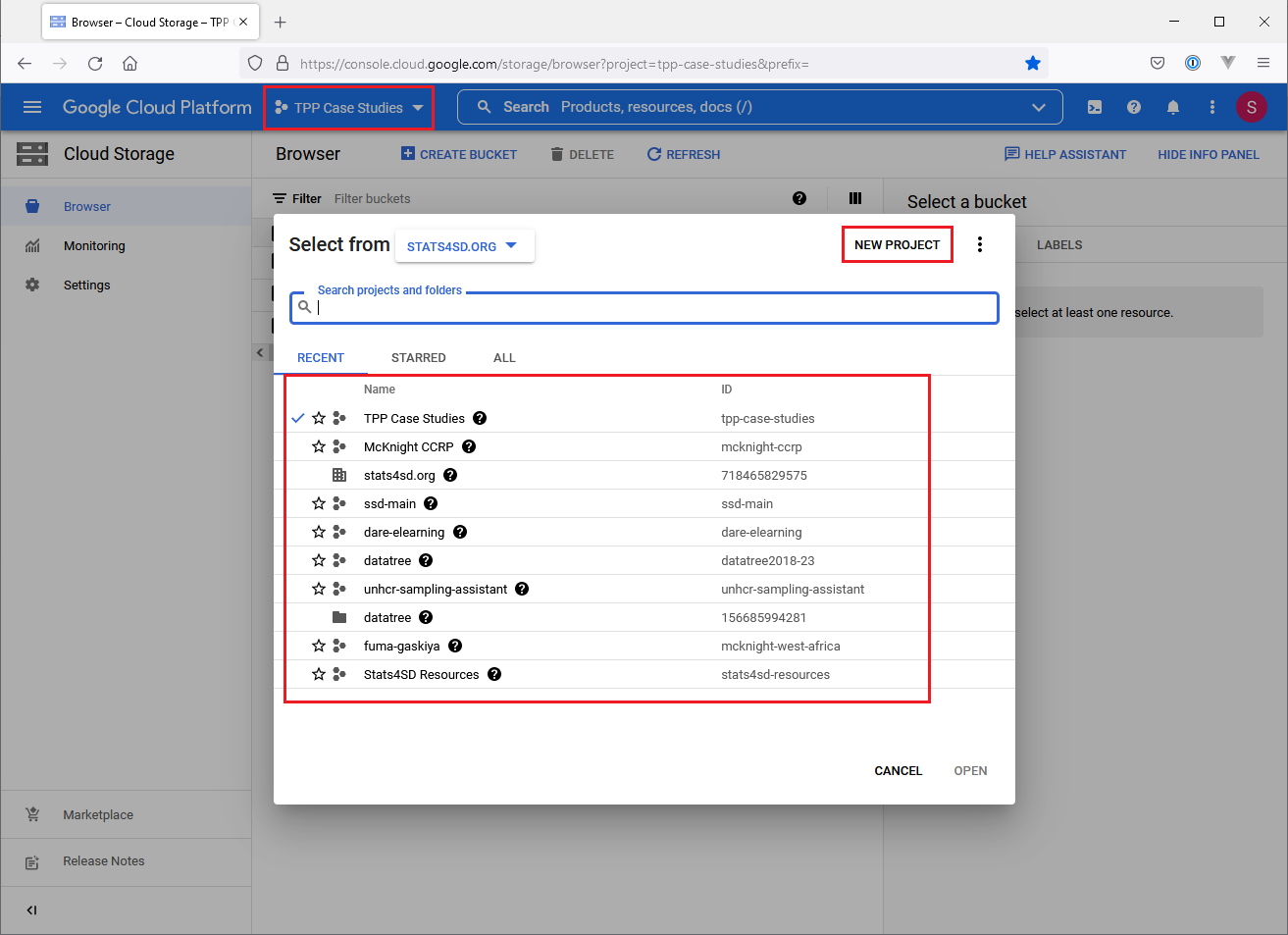
Graphical user interface, application

Description automatically generated

**Show Existing Projects**

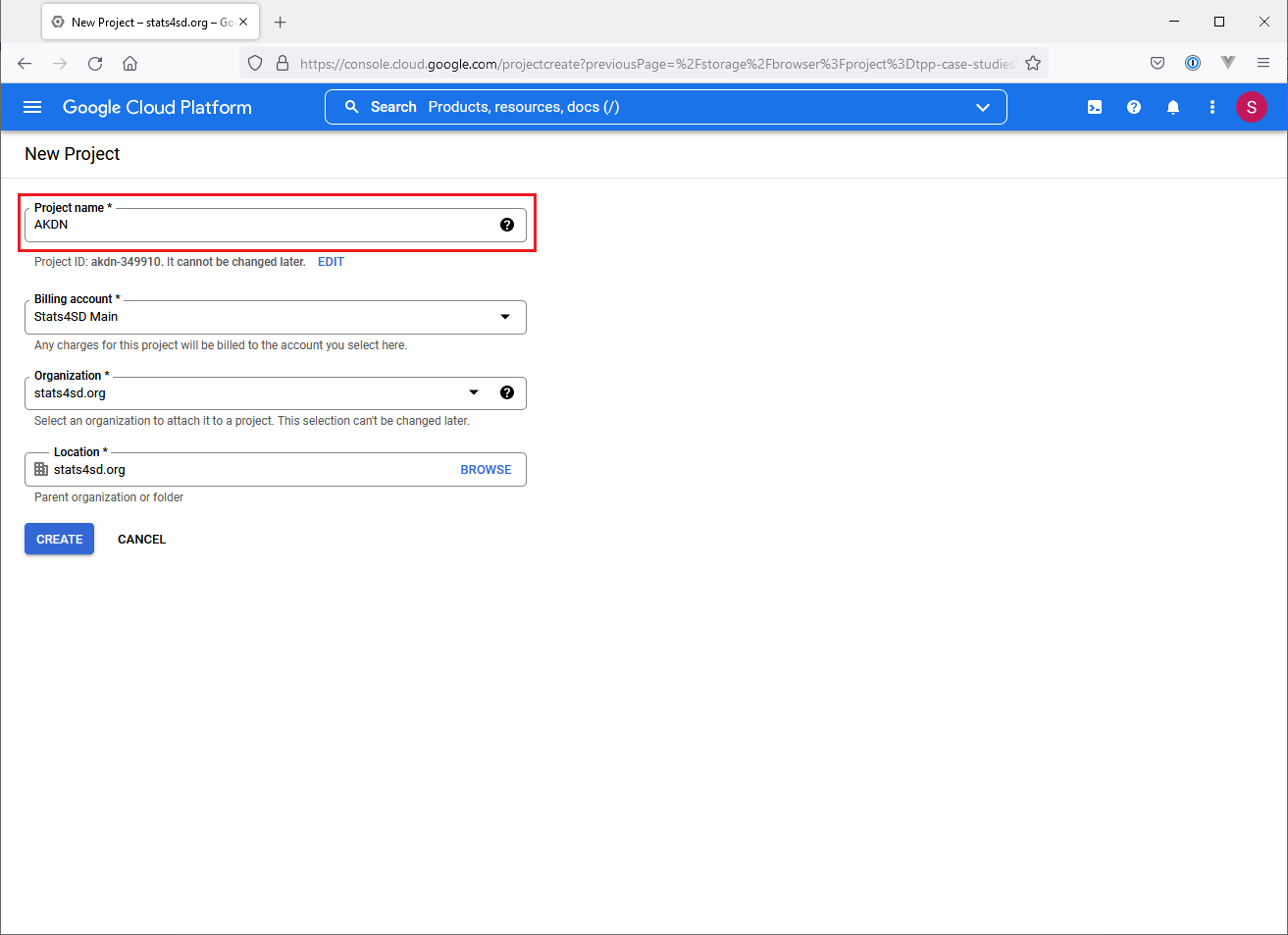
1. Click “TPP Case Studies” button at top bar to show all projects
2. In the popup panel, select an existing project or click “NEW PROJECT” button to create a new project
3. For our convention, we can create new buckets for CCRP related application in “McKnight CCRP” project. As TPP Case Studies platform does not belong to McKnight CCRP, we created a new project “TPP Case Studies” for new buckets.
4. As an example, we will create a new project for “AKDN”

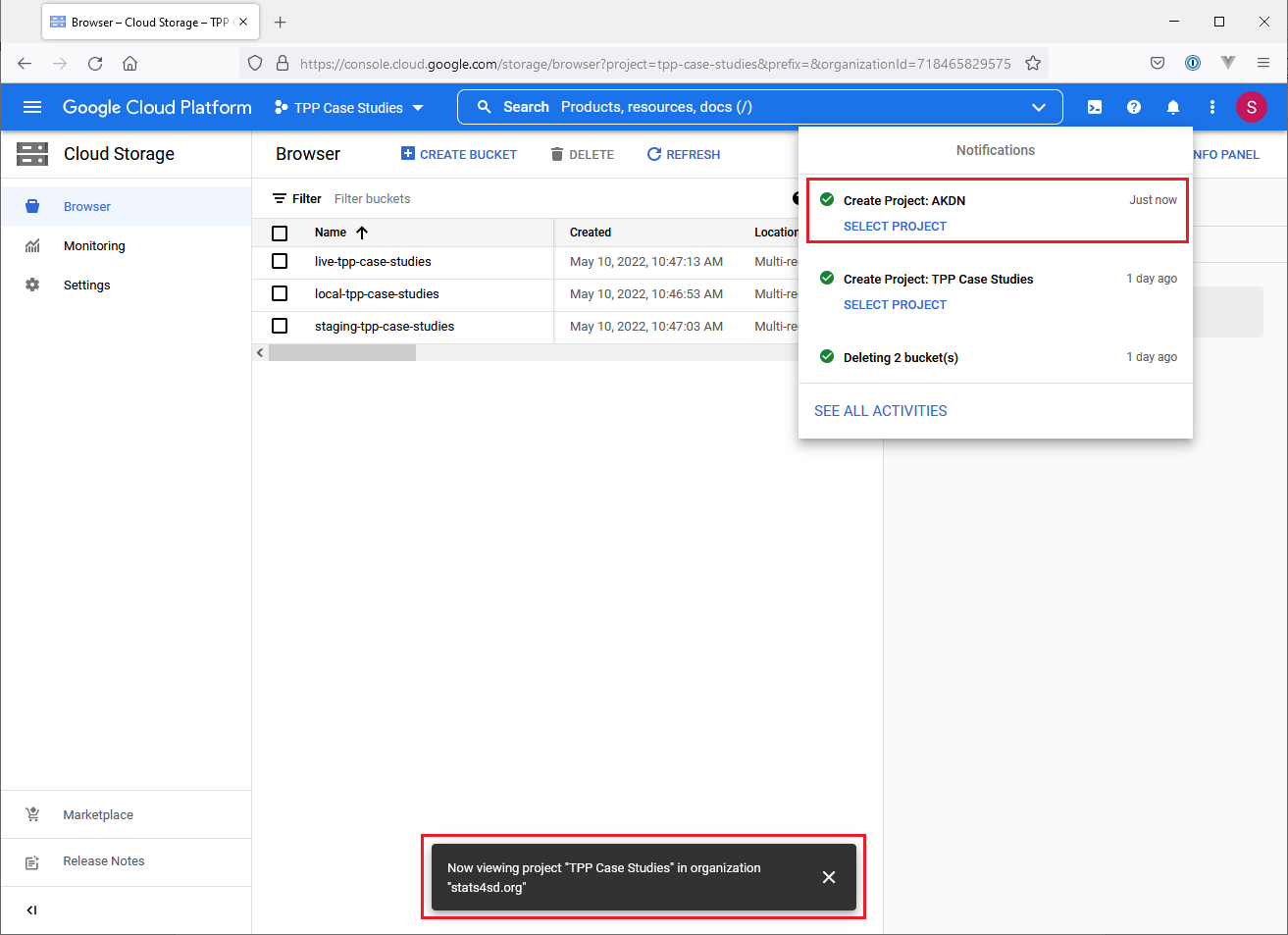
Shows all existing project



**Create a New Project**

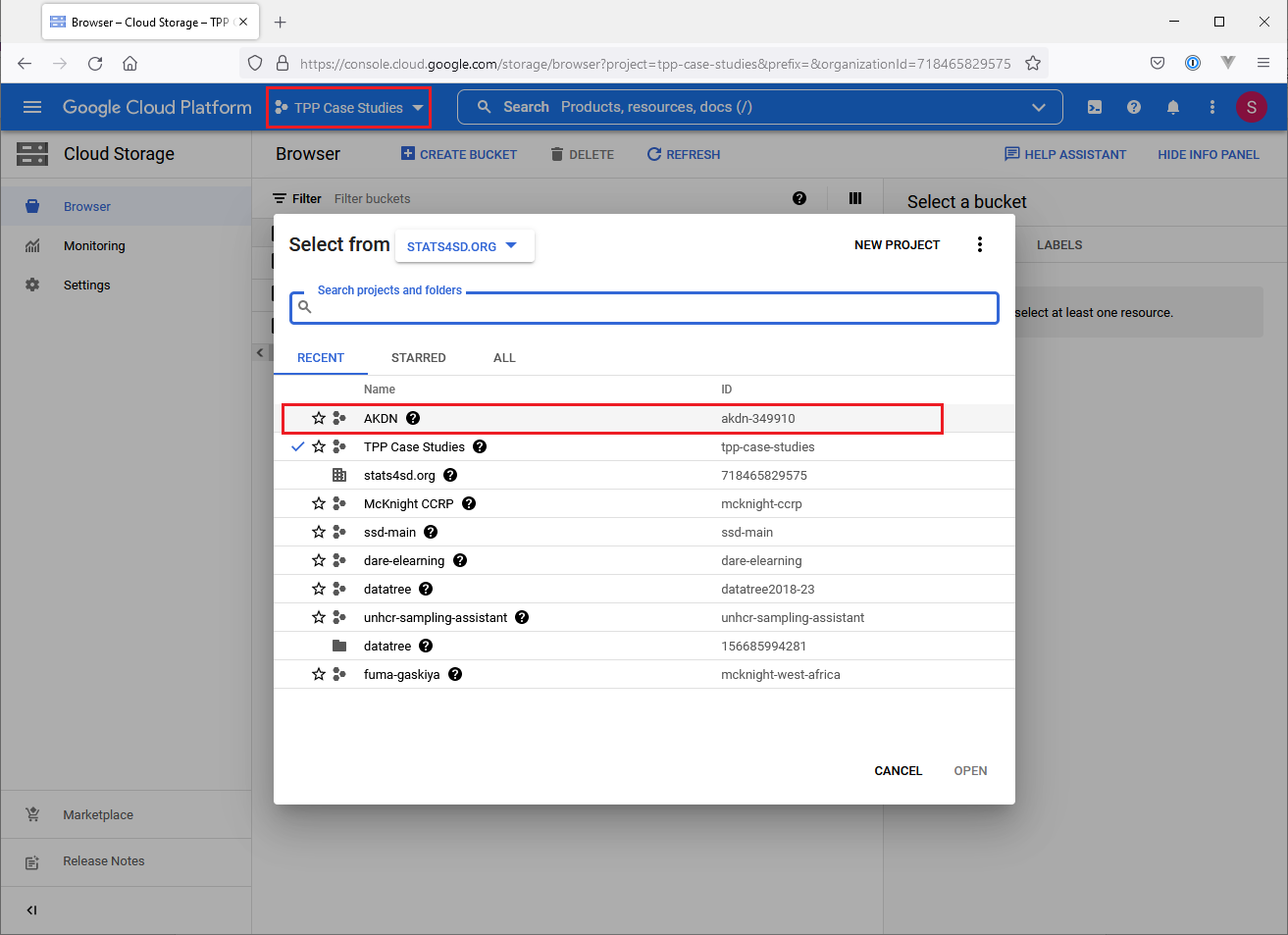
1. Click “NEW PROJECT” button in previous popup panel
2. Enter project name
3. Click “CREATE” button





**Switch to Another Project**

1. New project “AKDN” has been created
2. Click “TPP Case Studies” button in top bar, popup panel shows existing project
3. Click “AKDN” to open “AKDN” project



Graphical user interface, application

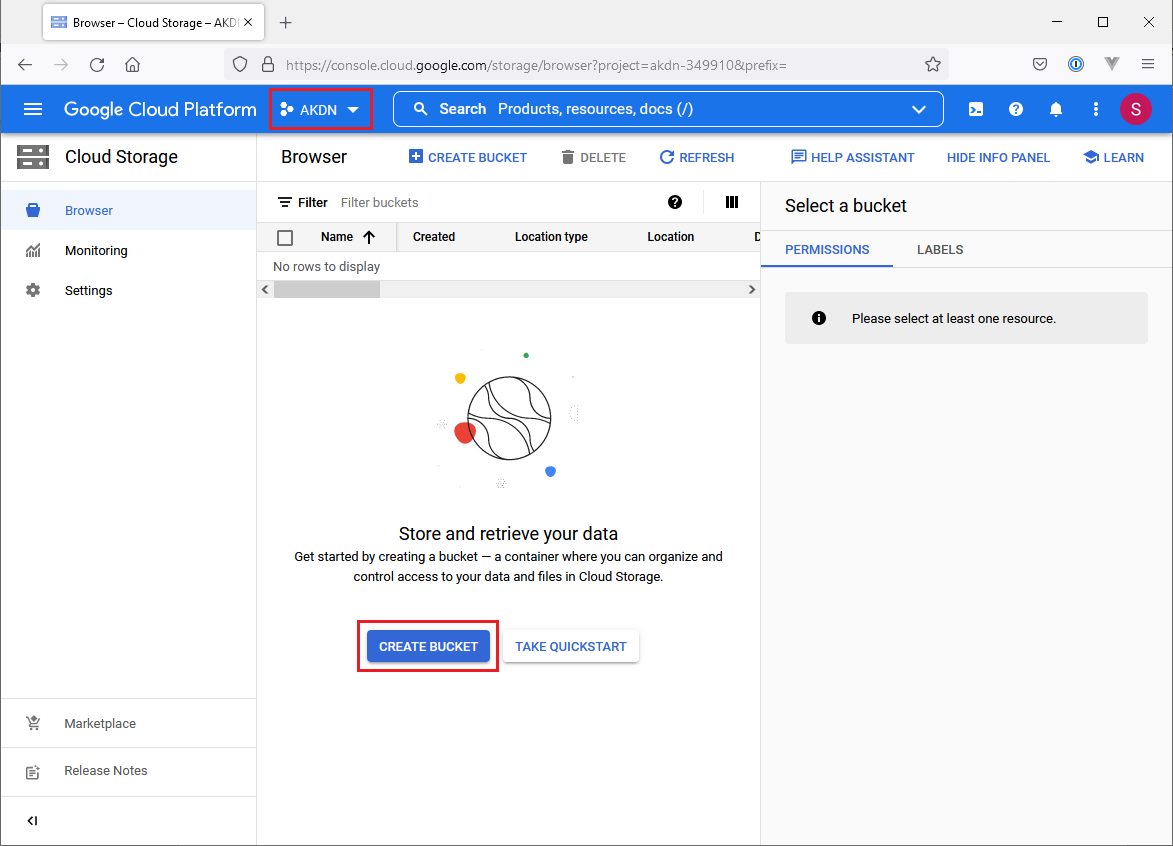
Description automatically generated

**Create New Buckets**

1. We should create individual bucket for different environment. E.g. we need three individual buckets for local env, staging env, live env respectively
2. Click “CREATE BUCKET” button
3. Fill in below details:

* Bucket name: local-xxx / staging-xxx / live-xxx
* Location type: Multi-region, select a region that nearer to most users
* Default storage class: Standard
* Access control: Fine-grained
* Protection tools: None
* Data encryption: Leave it empty

1. Click “Create” button
2. Repeat step 2 to 4 until all three buckets are created



Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

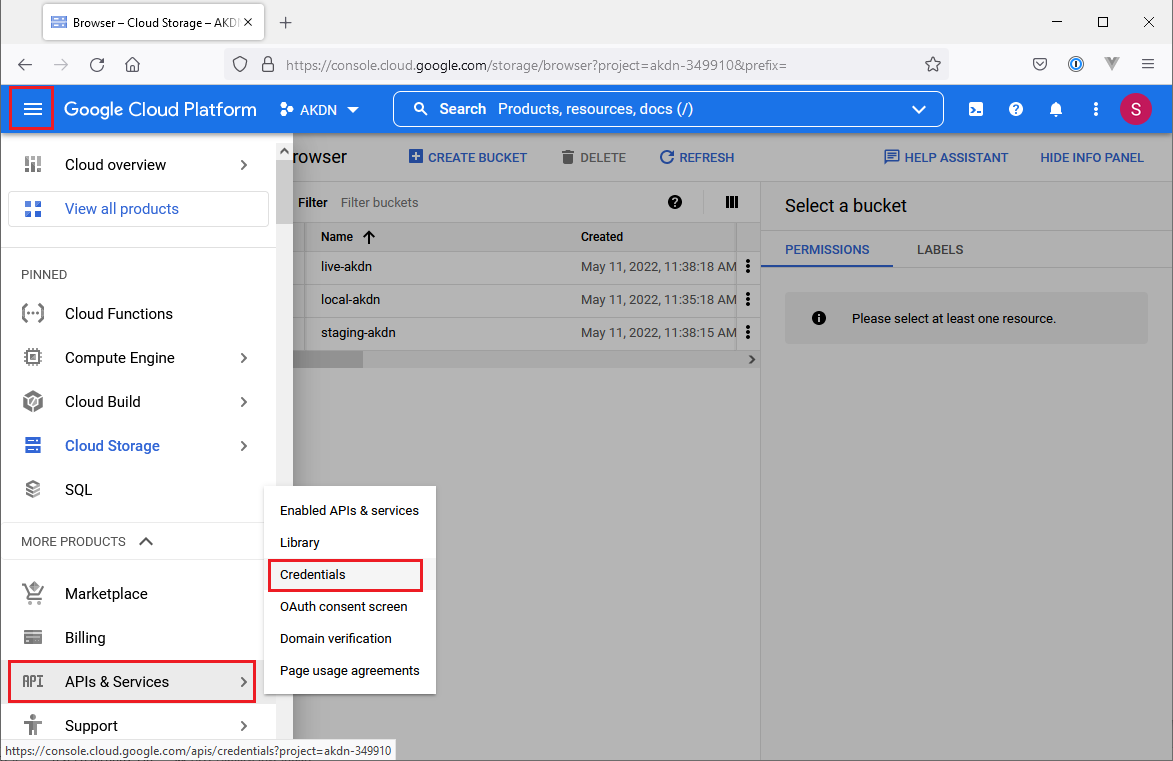
Three new buckets created for local env, staging env, live env respectively

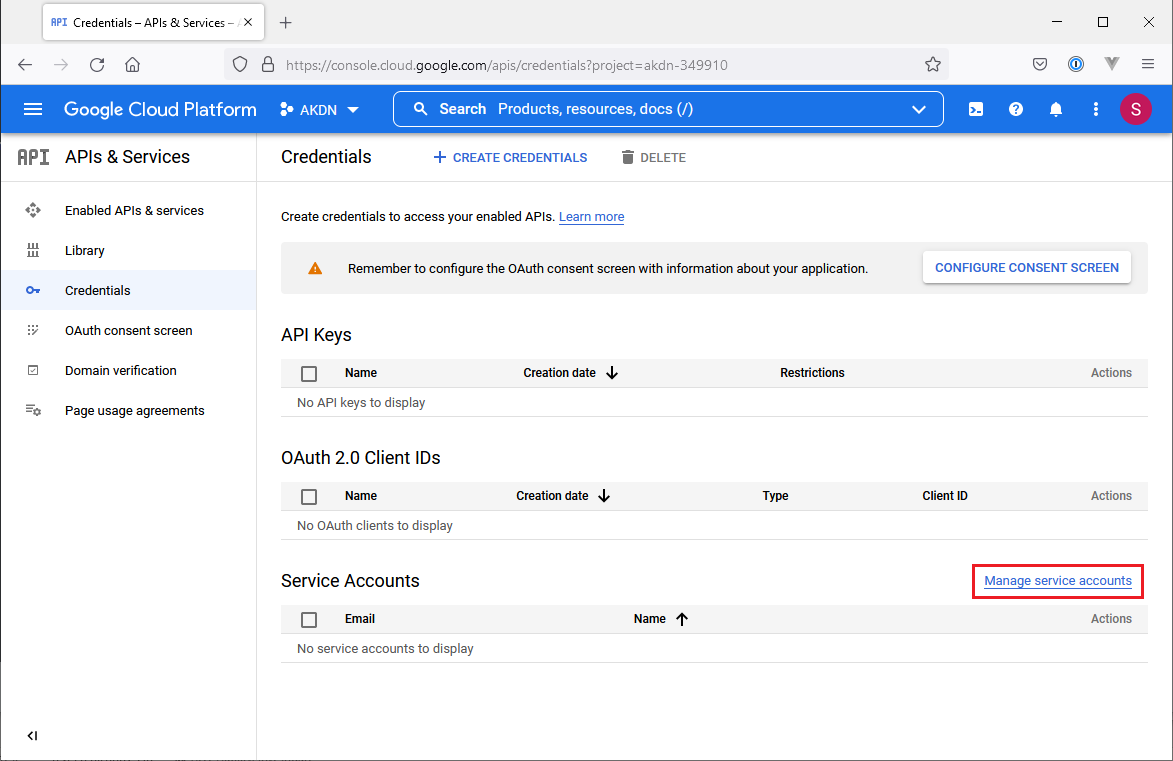
Graphical user interface, text, application

Description automatically generated

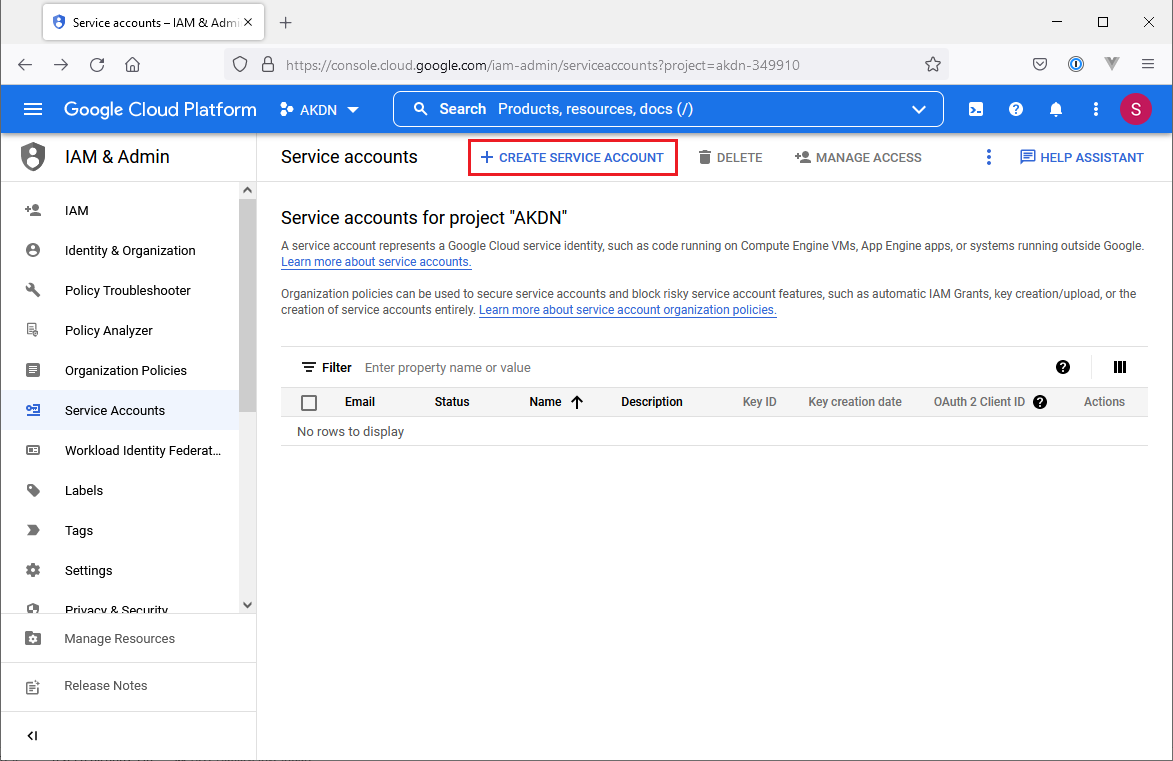
**Add a Service Account**

1. Click the button at upper left corner to show main menu
2. Click menu item APIs & Services > Credentials
3. In Credentials page, click “Manage service accounts”

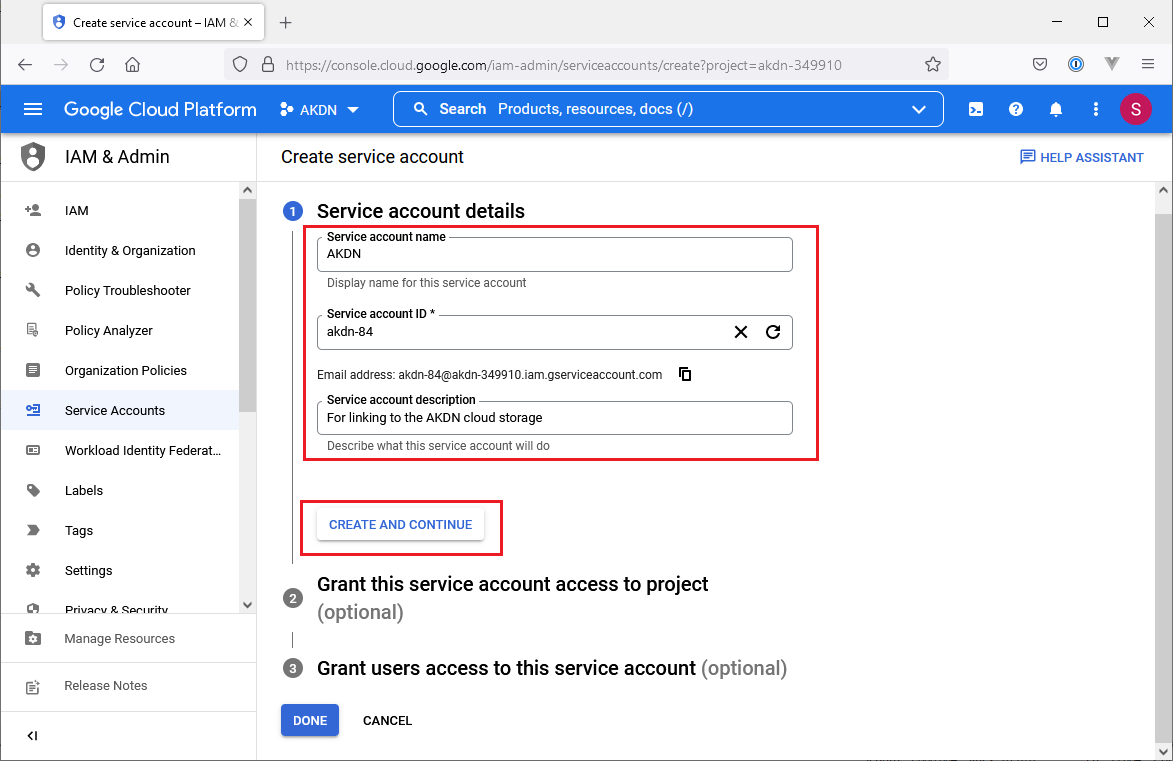




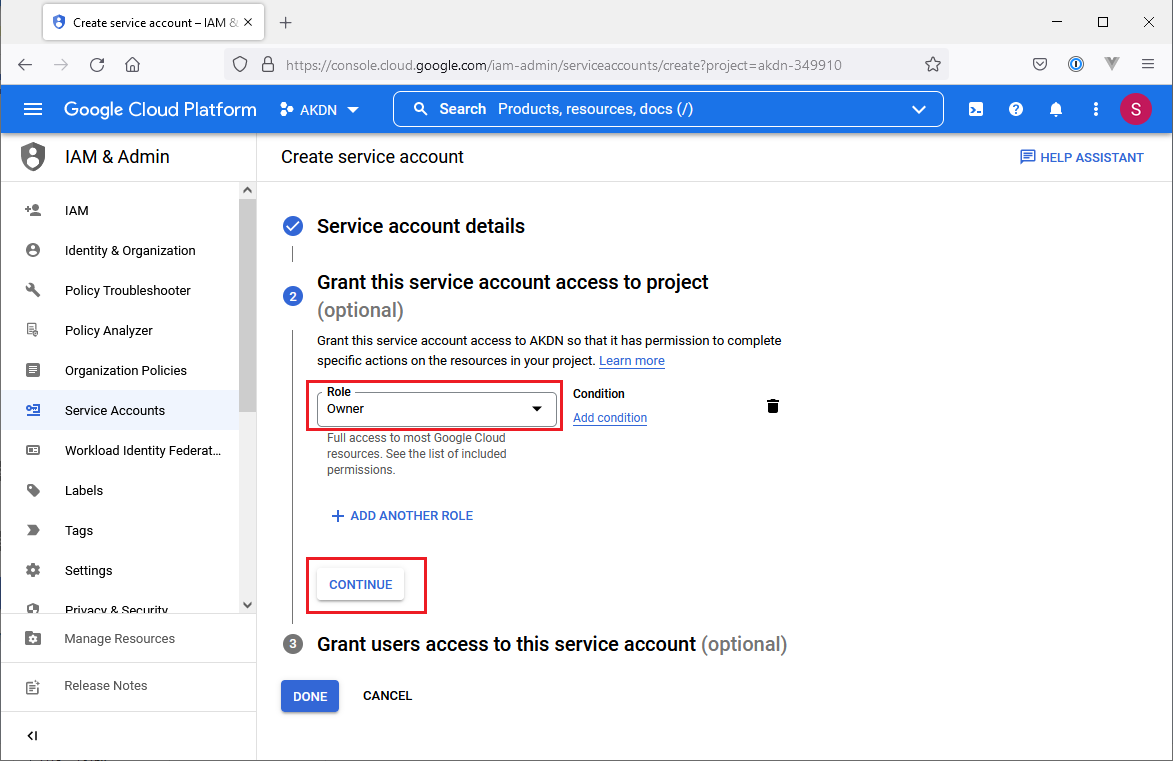
1. In Service accounts page, click “CREATE SERVICE ACCOUNT” button



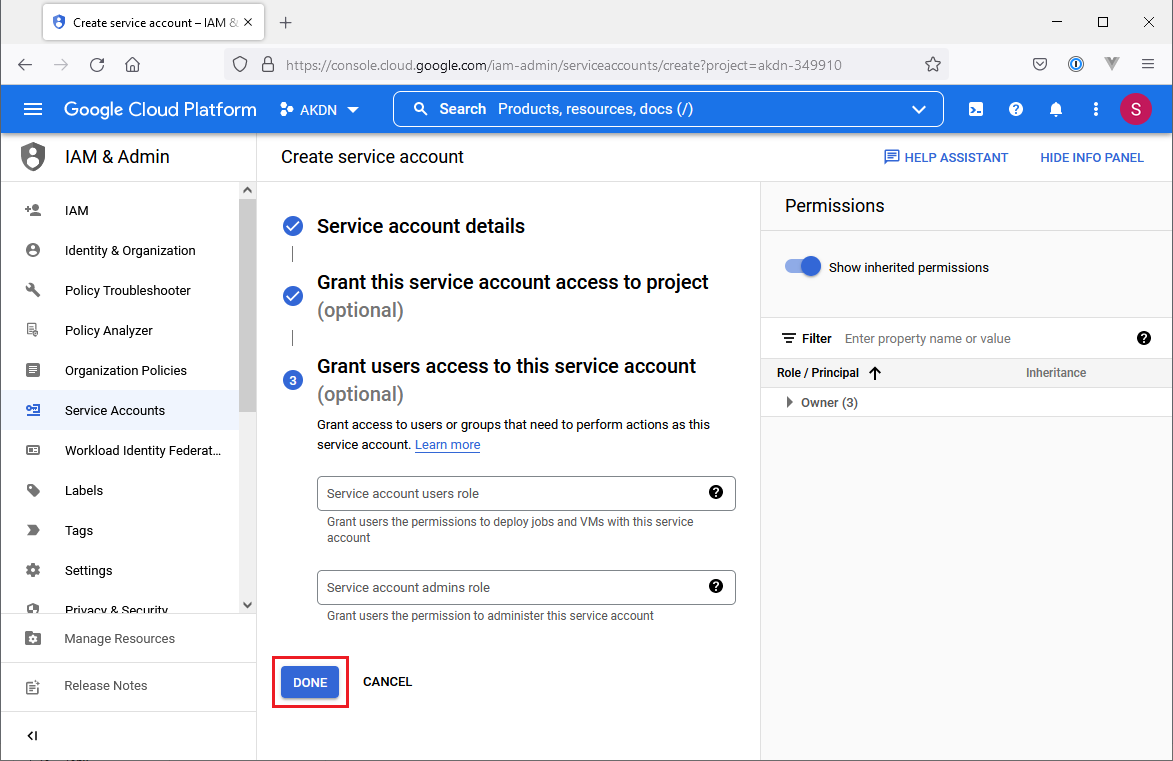
1. Fill in service account name and service account description, keep the auto generated Service account ID if you wish
2. Click “CREATE AND CONTINUE” button



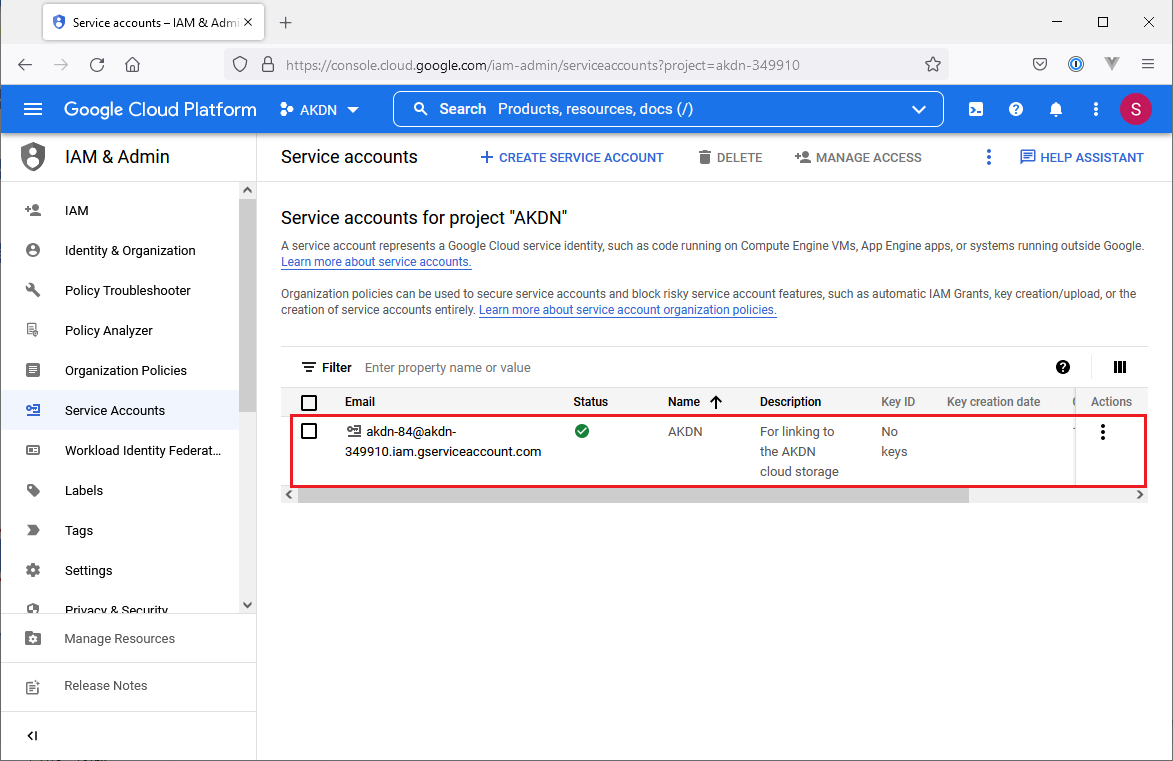
1. Select “Owner” role, click “CONTINUE” button



1. No need to fill in anything, click “DONE” button to complete

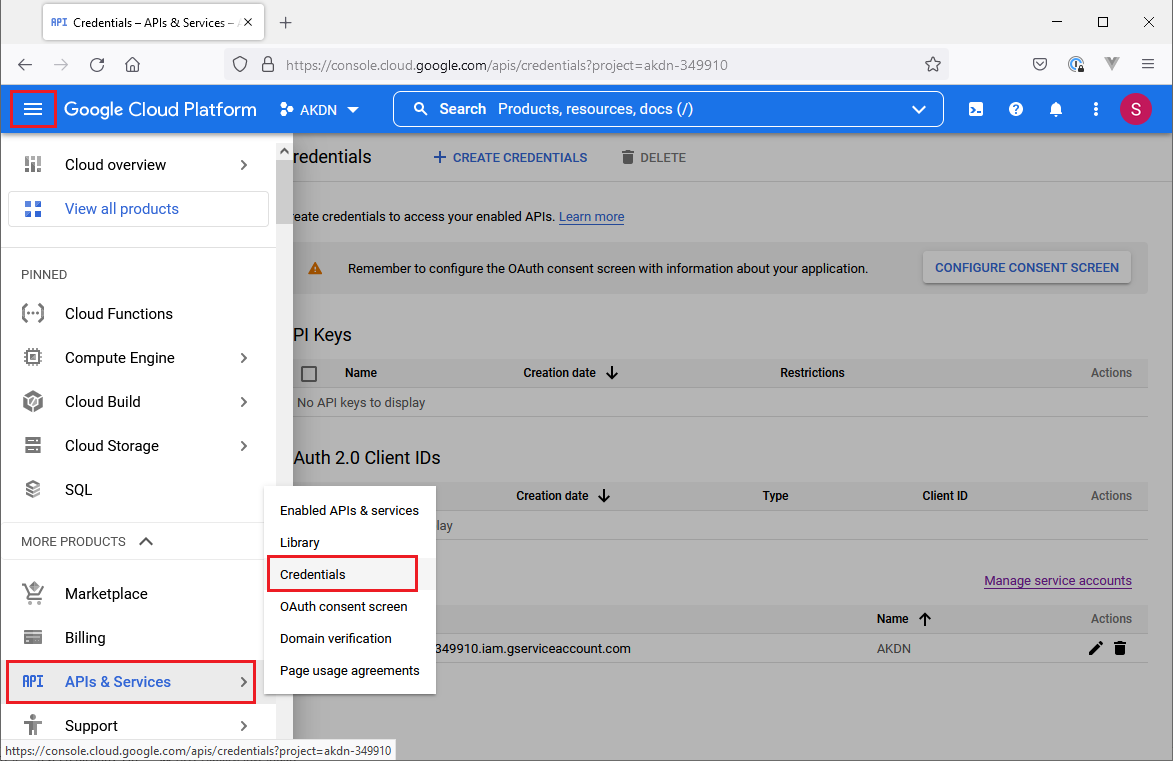


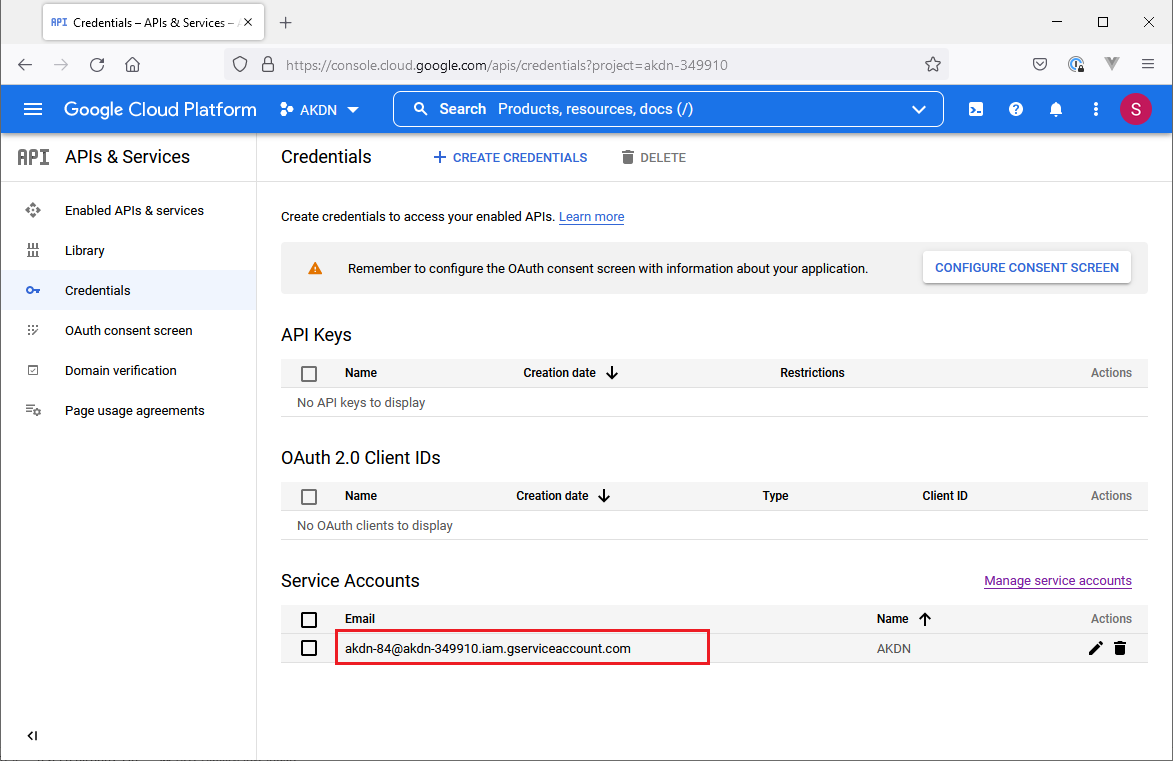
1. A new service account has been created



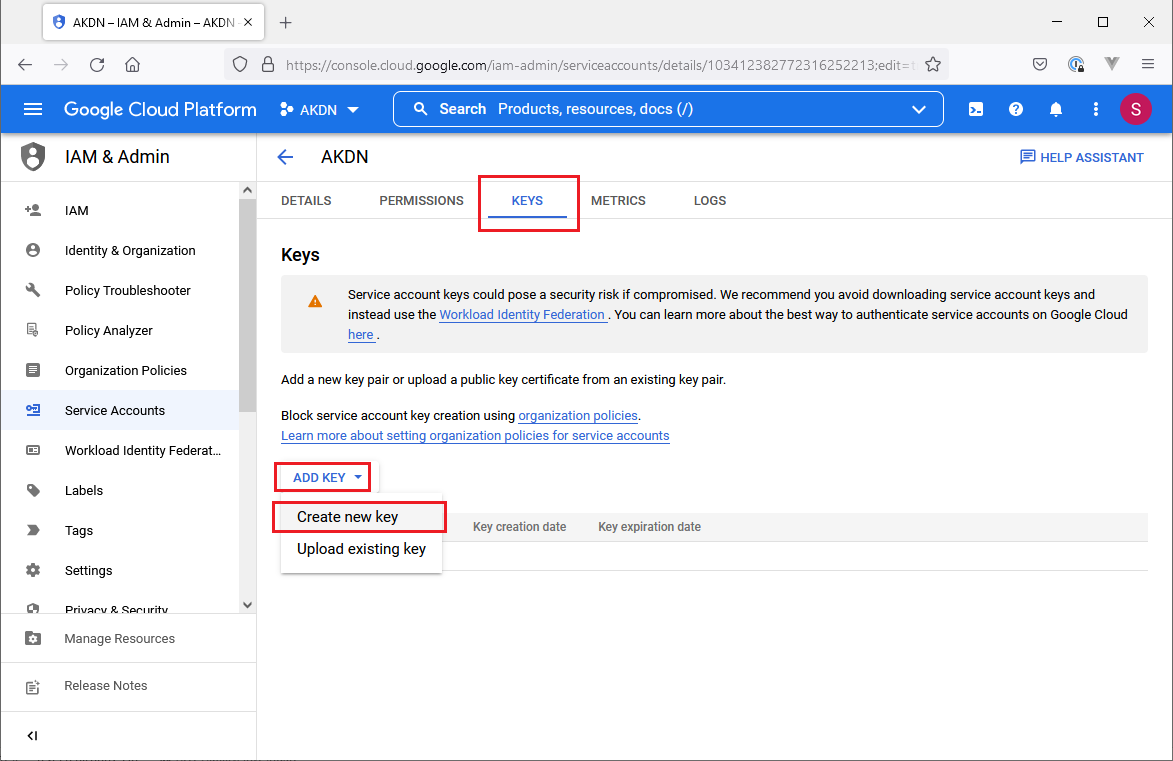
**Generate JSON Authentication File**

1. Click the button at upper left corner to show main menu
2. Click menu item APIs & Services > Credentials
3. In Credentials page, click the newly created service account





1. Click KEYS tab
2. Click “ADD KEY” button
3. Click “Create new key” button



1. Select “JSON”, click “CREATE” button

Graphical user interface, text, application

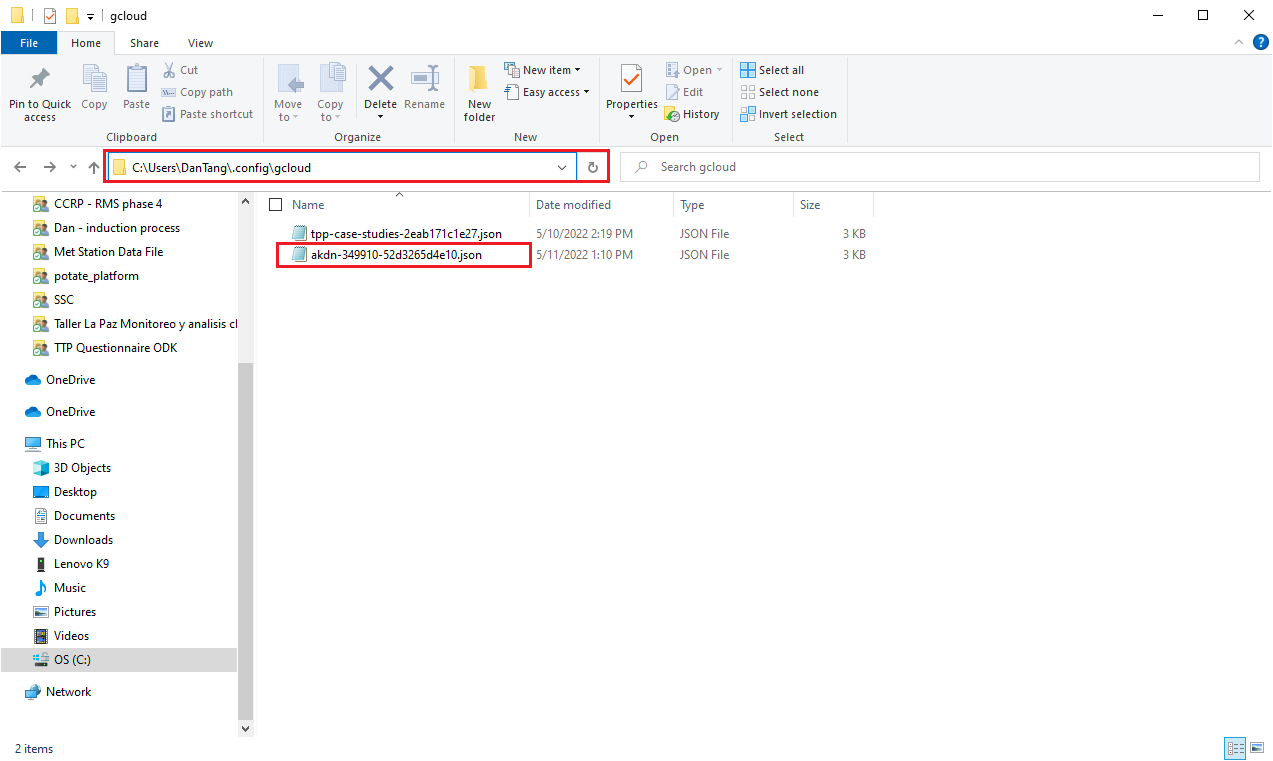
Description automatically generated

1. JSON file will be downloaded to local computer automatically
2. P.S. This JSON file can be used for different environments (local env, staging env, live env) for the same application. Because it provides full read/write access to the file storage, this file **MUST NOT** be included in Git repository or any publicly accessible folder
3. For local environment, you can move this file to folder, e.g. “C:\Users\DanTang\.config\gcloud”
4. For staging environment and live environment, you can upload this file to directory “/home/forge/.config/gcloud/”. As staging env and live environment are normally in the same server, the uploaded file will be accessible for both environments

* scp \*.json forge@xxx.xxx.xxx.xxx:/home/forge/.config/gcloud

Graphical user interface, text, application

Description automatically generated



**Install Google Cloud Storage Driver in Laravel Platform**

1. Install Laravel package “superbalist/laravel-google-cloud-storage” in local env by executing command in your Laravel project root folder
   * composer require superbalist/laravel-google-cloud-storage
2. composer.lock will be updated after installation, you can push composer.lock file to Git repo and merge into “dev” branch / “main” branch for deployment in staging env / live env (command “composer install” will be executed to install required Laravel package after deployment in staging env / live env)

**Add New Disk Configuration Items in config\filesystems.php**

1. Add below highlighted “gcs” segment in “disks” section in config\filesystems.php file

'disks' => [

'local' => [

'driver' => 'local',

'root' => storage\_path('app'),

],

'public' => [

'driver' => 'local',

'root' => storage\_path('app/public'),

'url' => env('APP\_URL').'/storage',

'visibility' => 'public',

],

'gcs' => [

'driver' => 'gcs',

'project\_id' => env('GOOGLE\_CLOUD\_PROJECT\_ID', 'your-project-id'),

'key\_file' => env('GOOGLE\_CLOUD\_KEY\_FILE', null), // optional: /path/to/service-account.json

'bucket' => env('GOOGLE\_CLOUD\_STORAGE\_BUCKET', 'your-bucket'),

'path\_prefix' => env('GOOGLE\_CLOUD\_STORAGE\_PATH\_PREFIX', null), // optional: /default/path/to/apply/in/bucket

'storage\_api\_uri' => env('GOOGLE\_CLOUD\_STORAGE\_API\_URI', null), // see: Public URLs below

'visibility' => 'public', // optional: public|private

],

],

**Add Google Cloud Storage Configuration Items in .env File**

1. Add new configuration items in .env file (only keep relevant items for corresponding env)
2. P.S. It is recommended to update both .env file in staging env and live env when arranging deployment for staging env. This can prevent forgetting to update .env file for live env deployment in later stage. It is no harm to add new configuration items in .env when they are not being used

# Google Cloud Storage

GOOGLE\_CLOUD\_PROJECT\_ID=akdn

# For local env

GOOGLE\_CLOUD\_KEY\_FILE=C:\Users\DanTang\.config\gcloud\akdn-349910-52d3265d4e10.json

GOOGLE\_CLOUD\_STORAGE\_BUCKET=local-akdn

# For staging env

# GOOGLE\_CLOUD\_KEY\_FILE=/home/forge/.config/gcloud/akdn-349910-52d3265d4e10.json

# GOOGLE\_CLOUD\_STORAGE\_BUCKET=staging-akdn

# For live env

# GOOGLE\_CLOUD\_KEY\_FILE=/home/forge/.config/gcloud/akdn-349910-52d3265d4e10.json

# GOOGLE\_CLOUD\_STORAGE\_BUCKET=live-akdn

GOOGLE\_CLOUD\_STORAGE\_PATH\_PREFIX=

GOOGLE\_CLOUD\_STORAGE\_API\_URI=

# Set media storage disk

MEDIA\_DISK='gcs'

**Perform Testing in Local Env**

1. Make sure uploaded files can be stored in local storage before switching to Google Cloud Storage
2. To avoid possible confusion, remove all uploaded files in application
3. Update .env file
4. Apply changes to switch from local storage to Google Cloud Storage. i.e. Install Google Cloud Storage driver, update config\filesystems.php
5. Perform testing, upload files in application
6. Check whether newly uploaded files are uploaded in corresponding bucket for local env in Google Cloud Storage

**Perform Testing in Staging Env**

1. Make sure uploaded files can be stored in server local storage before switching to Google Cloud Storage
2. To avoid possible confusion, remove all uploaded files in application
3. Update .env file in Laravel Forge
4. Submit a pull request (PR) to apply changes to switch from local storage to Google Cloud Storage (including composer.lock, config\filesystems.php)
5. Get PR approval, then merge code into branch “dev”
6. Deployment will be proceeded automatically if “Quick Deploy” is enabled. You can click “Deploy Now” button to trigger deployment
7. Perform testing, upload files in application
8. Check whether newly uploaded files are uploaded in corresponding bucket for staging env in Google Cloud Storage

**Perform Testing in Live Env**

1. Make sure uploaded files can be stored in server local storage before switching to Google Cloud Storage
2. To avoid possible confusion, remove all uploaded files in application
3. Update .env file in Laravel Forge (P.S. It is recommended to update .env in live env when updating staging env)
4. Submit a pull request (PR) to apply changes to switch from local storage to Google Cloud Storage (including composer.lock, config\filesystems.php)
5. Get PR approval, then merge code into branch “main”
6. Deployment will be proceeded automatically if “Quick Deploy” is enabled. You can click “Deploy Now” button to trigger deployment
7. Perform testing, upload files in application
8. Check whether newly uploaded files are uploaded in corresponding bucket for live env in Google Cloud Storage