Before getting to the stage where I could build a Generative AI application in Google Cloud that everyone is excited about, I worked through some instructions to set up a local VS Code setup of the GenAI application which authenticates using a Service account into Google Cloud. One service account, three projects. I had three projects in GCP. At set up time, I was running IaC with a given Terraform code base and I ran into some issues where the IaC step failed for two of the three projects.

The original recommendation was to do the set up with default credentials. This would have meant, running these 4 commands below (as instructed here - <https://github.com/UtpalMattoo/e2e-git-local-git>)

*export PROJECT\_ID="YOUR\_PROJECT\_ID"*

*gcloud config set project $PROJECT\_ID*

*gcloud auth application-default login*

*gcloud auth application-default set-quota-project $PROJECT\_ID*

However, since I decide to use a service account I ran into some interesting problems. (I needed to run only the first two and not the last two lines in the 4 commands above).

As a beginner into GCP (and the cloud space in general), I thought I would capture the error trail and resolution:

**✅ Root Cause Analysis: Why Terraform Only Created 69 of 71 Resources**

An issue where **Terraform only created 69 out of 71 resources**.

The missing two resources were:

* **Staging environment IAM setup**
* **Production environment IAM setup**

These failures occurred **silently or with partial errors**, depending on whether the associated **service accounts** lacked permissions.

**🔍 Immediate Symptoms**

* You had **69/71 resources created** during terraform apply.
* Terraform **did not create the CICD Runner Service Account**
* You didn’t see the expected **OAuth 2 Client ID** until later (after the fix)
* The **staging and production IAM roles** for the CICD service account were **not created**. Terraform did not have sufficient permissions to modify IAM bindings in the staging and production projects because (as explained below) Terraform used the wrong identity as my Application Default Credentials (ADC) were pointing to an email address (which was not the service account I was expecting)
* No errors pointed clearly to the root cause.

**⚠️ Misleading Clues: gcloud auth list Looked Fine**

You had the correct service account set as active in your CLI (my GitBash terminal in VS Code showed the Service Account as ACTIVE and the personal gmail as inactive)

$ gcloud auth list

ACTIVE: \*

ACCOUNT: *service account*

But I wasn’t aware that Terraform *does not use this setting*. Instead, it reads credentials from **Application Default Credentials (ADC)** at:

~/.config/gcloud/application\_default\_credentials.json

At that time, the ADC file still contained:

{

"client\_id": "...",

"client\_secret": "...",

"refresh\_token": "...",

"type": "authorized\_user",

"email": "the email that caused permission failure for staging and production”

}

As a result, Terraform authenticated as a **personal Gmail account**, which:

* Had **limited access to IAM resources**, especially in **staging** and **production** projects.
* **Lacked permission to create service accounts** or bind IAM roles across environments.

**🔑 The Breakthrough: Overwriting the ADC File**

You fixed this by explicitly pointing ADC to the correct **Service Account JSON key**:

gcloud auth application-default login --cred-file=/path/to/your-service-account.json

This replaced the ADC file with something like:

{

"type": "service\_account",

"project\_id": "dev project id that triggers the CI CD pipeline",

"private\_key\_id": "long alphanumeric string",

"client\_email": "the actual service account email that should have been used instead of the personal email",

...

}

✅ Now, Terraform used the **correct service account** with sufficient permissions.

**✅ Additional Actions That Were Crucial**

| **Step** | **Details** |
| --- | --- |
| ✅ Enabled required APIs | On staging and prod: IAM, Cloud Resource Manager, Cloud Build |
| ✅ Assigned IAM roles | Editor, Service Account User, Project IAM Admin to the service account |
| ✅ Cleaned up old .tfstate references | Removed irrelevant state files pointing to a deprecated project |
| ✅ Used --var-file=vars/env.tfvars | Ensured variables were passed properly to differentiate environments |

**📌 Final Verification and Outcome**

After correcting the ADC and permissions:

* terraform apply ran successfully with **no skipped resources**
* ✅ **All 71 resources** were created
* ✅ The **CICD service account** was finally created
* ✅ You finally saw the **OAuth 2 Client ID** for that service account populated in the Console:
* OAuth 2 Client ID: 22 digit long number

This confirmed successful **key issuance**, and the service account was now **fully provisioned** with the proper credentials.

**💡 Why This Happened (Technically Speaking)**

| **Component** | **What Happened** | **Why It Mattered** |
| --- | --- | --- |
| gcloud auth list | Showed active service account | ❌ Misleading for Terraform/SDKs |
| application\_default\_credentials.json | Still pointed to personal Gmail | ❌ Had insufficient permissions |
| ADC overwritten via --cred-file | Used correct SA JSON | ✅ Let Terraform authenticate correctly |
| Terraform state logic | Could not create staging/prod IAM roles | ✅ Fixed once permissions were correct |

**✅ Final Commands That Fixed the Issue**

**# 1. Authenticate ADC using service account credentials**

gcloud auth application-default login --cred-file=/path/to/service-account.json

**# 2. (Optional because local set up always showed this to be active) Set CLI to same account, for consistency**

gcloud config set account *service\_account\_use\_to\_authenticate\_from\_local\_to\_GCP*

**# 3. Apply Terraform using correct variable set**

terraform apply --var-file=vars/env.tfvars

**# Set ADC to correct service account**

gcloud auth application-default login --cred-file=/path/to/your-service-account.json

**# Check which identity is currently active for ADC**

gcloud auth application-default print-access-token | \

xargs -I {} curl -s https://oauth2.googleapis.com/tokeninfo?access\_token={}

This command above showed my personal email instead of the service account (although it seemed that my service account was active and it was also assigned to all the projects with the right permissions in google cloud console) in use by terraform

**# Clean up outdated Terraform state**

rm terraform.tfstate terraform.tfstate.backup

**# Redeploy**

terraform apply --var-file=vars/env.tfvars