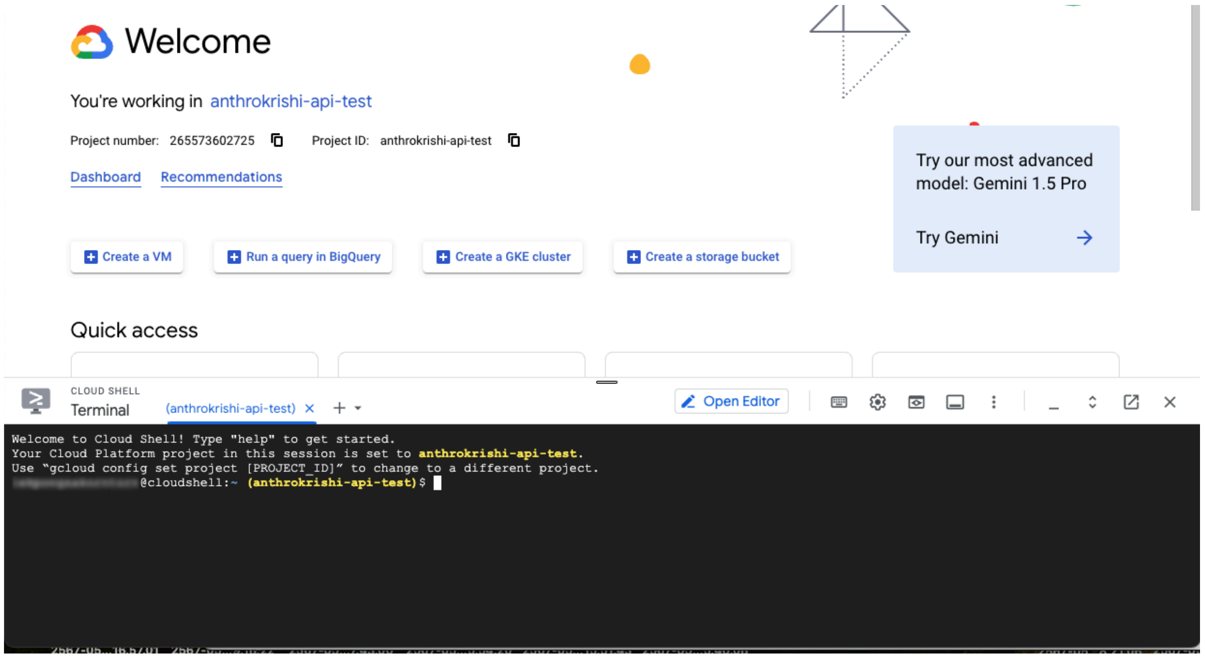
ALU API | Customer Onboarding

Prerequisites

Make sure you have GCP billing and project setup. Please use this checklist to enable and use all the necessary services.

Deploy cloud services

1. Open a Cloud Shell in the GCP console.



```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to anthrokrishi-api-test.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
[redacted]@cloudshell:~ (anthrokrishi-api-test)$
```

2. Set environment variables to store your GCP project ID and enable needed service:

```
Unset
PROJECT_ID="YOUR_GCP_PROJECT_ID"

gcloud services enable cloudresourcemanager.googleapis.com --project
$PROJECT_ID
```

3. Clone the alu-api repository from Github and make sure you are in the top-level directory of the Git repository:

```
Unset
git clone https://github.com/google/alu-api && cd alu-api
```

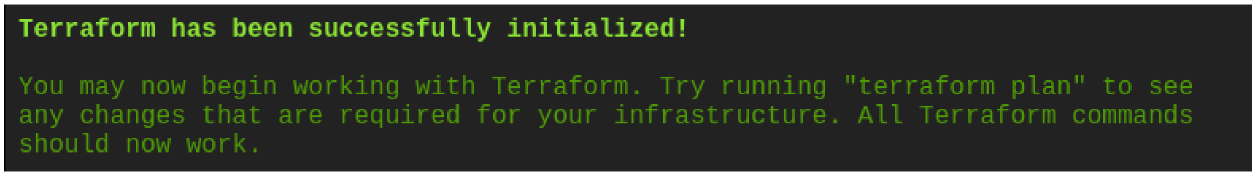
4. Optionally, create a storage bucket and configure Terraform to store its state remotely for improved reliability and collaboration:

```
Unset
gsutil mb "gs://$PROJECT_ID-tf"

cat <<EOF >terraform/backend.tf
terraform {
  backend "gcs" {
    bucket  = "$PROJECT_ID-tf"
    prefix  = "terraform/state"
  }
}
EOF
```

5. Initialize Terraform:

```
Unset
terraform -chdir=terraform/ init
```



```
Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
```

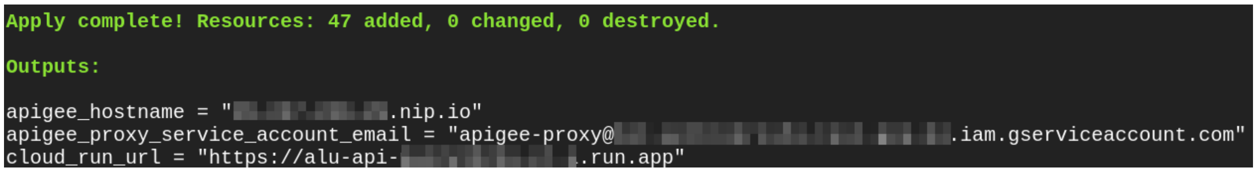
Example successful response

6. Optionally, preview the planned infrastructure changes:

```
Unset
terraform -chdir=terraform/ plan -var="project_id=$PROJECT_ID"
```

7. Applies the planned infrastructure changes (It takes approximately 40 minutes):

Unset
```
terraform -chdir=terraform/ apply -var="project_id=$PROJECT_ID" -auto-approve
```



```
Apply complete! Resources: 47 added, 0 changed, 0 destroyed.

Outputs:

apigee_hostname = "[redacted].nip.io"
apigee_proxy_service_account_email = "apigee-proxy@[redacted].iam.gserviceaccount.com"
cloud_run_url = "https://alu-api-[redacted].run.app"
```

Example successful response

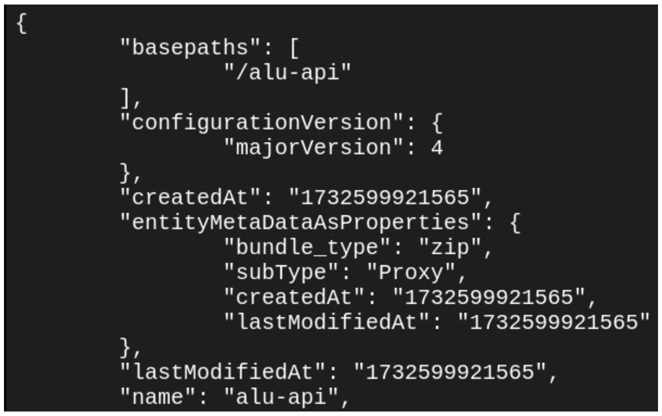
8. Install and configure [apigeecli](https://github.com/apigee/apigeecli) tool:

Unset
```
curl -L
https://raw.githubusercontent.com/apigee/apigeecli/main/downloadLatest.sh | sh
-
export PATH=$PATH:$HOME/.apigeecli/bin
apigeecli prefs set -o $PROJECT_ID
```

9. Update config files and create Apigee API proxy:

Unset
```
CLOUD_RUN_URL=$(terraform -chdir=terraform/ output cloud_run_url | tr -d '"')
sed -i "s|@CLOUD_RUN_URL@|$CLOUD_RUN_URL|" ./apiproxy/targets/default.xml

apigeecli apis create bundle \
--name="alu-api" \
--proxy-folder="./apiproxy" \
--metadata-token
```



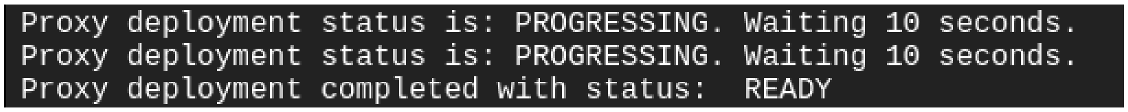
```
{
        "basepaths": [
                "/alu-api"
        ],
        "configurationVersion": {
                "majorVersion": 4
        },
        "createdAt": "1732599921565",
        "entityMetaDataAsProperties": {
                "bundle_type": "zip",
                "subType": "Proxy",
                "createdAt": "1732599921565",
                "lastModifiedAt": "1732599921565"
        },
        "lastModifiedAt": "1732599921565",
        "name": "alu-api",
```

Example successful response

10. Deploy Apigee API proxy (It takes approximately 3 minutes):

```
Unset
APIGEE_PROXY_EMAIL=$(terraform -chdir=terraform/ output
apigee_proxy_service_account_email | tr -d '"')

apigeecli apis deploy \
--name="alu-api" \
--env="prod" \
--sa="$APIGEE_PROXY_EMAIL" \
--safedeploy \
--sequencedrollout \
--wait \
--metadata-token
```



```
Proxy deployment status is: PROGRESSING. Waiting 10 seconds.
Proxy deployment status is: PROGRESSING. Waiting 10 seconds.
Proxy deployment completed with status:  READY
```

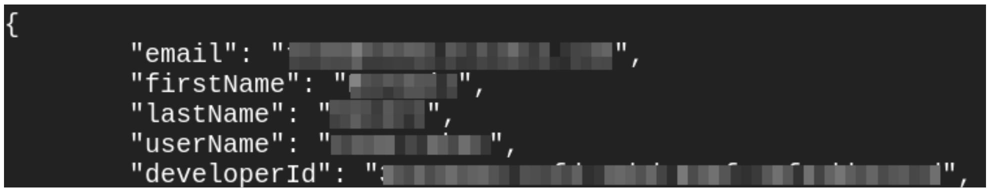
Example successful response

11. Set the following environment variables with your values:

```
Unset
DEVELOPER_EMAIL="YOUR_EMAIL"
DEVELOPER_FIRST_NAME="YOUR_FIRST_NAME"
DEVELOPER_LAST_NAME="YOUR_LAST_NAME"
DEVELOPER_USERNAME="YOUR_USERNAME"
```

12. Create an Apigee developer account:

```
Unset
apigeecli developers create \
--email=$DEVELOPER_EMAIL \
--first=$DEVELOPER_FIRST_NAME \
--last=$DEVELOPER_LAST_NAME \
--user=$DEVELOPER_USERNAME \
--metadata-token
```



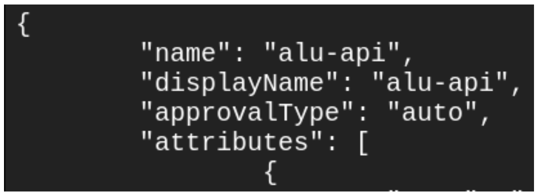
```
{
        "email": "[redacted]",
        "firstName": "[redacted]",
        "lastName": "[redacted]",
        "userName": "[redacted]",
        "developerId": "[redacted]",
```

Example successful response

13. Create an Apigee API Product:

Unset

```
apigeecli products create \
--name="alu-api" \
--display-name="alu-api" \
--opgrp="./apiproduct-op-group.json" \
--envs="prod" \
--approval="auto" \
--attrs="access=public" \
--metadata-token
```



```
{
        "name": "alu-api",
        "displayName": "alu-api",
        "approvalType": "auto",
        "attributes": [
                {
```

Example successful response

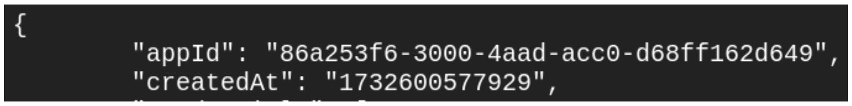
14. Create an Apigee App and credential:

Unset

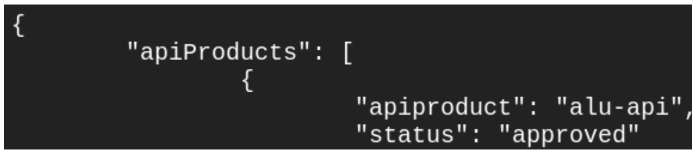
```
apigeecli apps create \
--name="alu-api" \
--email=$DEVELOPER_EMAIL \
--metadata-token

API_KEY=$(apigeecli apps get --name="alu-api" --metadata-token | jq
'.[0].credentials[0].consumerKey' | tr -d '"')

apigeecli apps keys update \
--name="alu-api" \
--dev=$DEVELOPER_EMAIL \
--key=$API_KEY \
--prods="alu-api" \
--metadata-token
```



```
{
        "appId": "86a253f6-3000-4aad-acc0-d68ff162d649",
        "createdAt": "1732600577929",
```



```
{
        "apiProducts": [
                {
                        "apiproduct": "alu-api",
                        "status": "approved"
```

Example successful responses

Test the deployment

To get the base URL of the API, run the following commands.

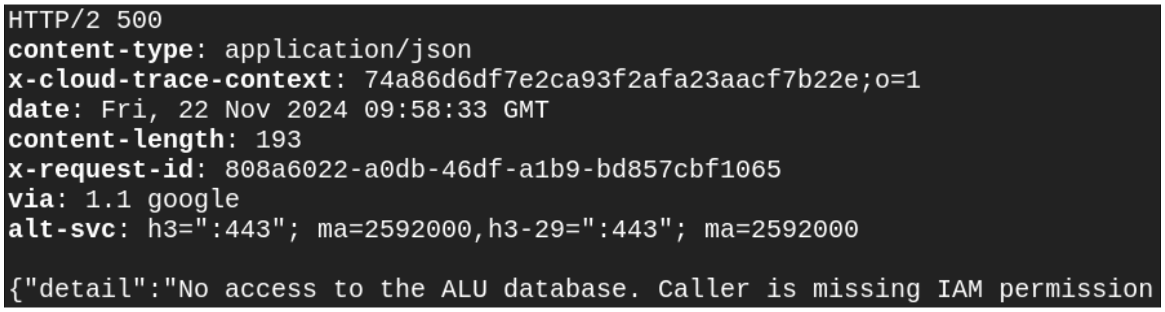
```
Unset
API_HOSTNAME=$(terraform -chdir=terraform/ output apigee_hostname | tr
-d '"')
API_BASE_URL="https://$API_HOSTNAME/alu-api"
echo $API_BASE_URL
```

To get the API key, run the following commands.

```
Unset
API_KEY=$(apigeecli apps get --name="alu-api" --metadata-token | jq
'.[0].credentials[0].consumerKey' | tr -d '"')
echo $API_KEY
```

To test the API, run the below command. For now, you should get an error response since you do not have access to the database. To get the access, please follow the steps in the Submit your GCP IDs section.

```
Unset
curl -i
"$API_BASE_URL/v1/get-landscape?s2cell=4316826463333515264&apikey=$API_K
EY"
```



```
HTTP/2 500
content-type: application/json
x-cloud-trace-context: 74a86d6df7e2ca93f2afa23aacf7b22e;o=1
date: Fri, 22 Nov 2024 09:58:33 GMT
content-length: 193
x-request-id: 808a6022-a0db-46df-a1b9-bd857cbf1065
via: 1.1 google
alt-svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"detail":"No access to the ALU database. Caller is missing IAM permission
```

Example error response

Once you have access, try running the command again and you should get a successful response like the one in the API spec.

Submit your GCP IDs

GCP organization ID

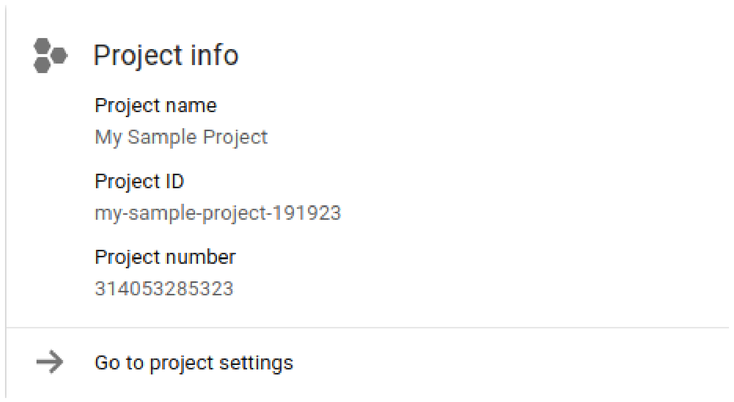
The organization resource ID is a unique identifier for an organization resource and is automatically created when your organization resource is created. Organization resource IDs are formatted as decimal numbers, and cannot have leading zeros.

You can get your organization resource ID using the Google Cloud console, the gcloud CLI, or the Cloud Resource Manager API. (see more)

GCP project ID

A project ID is a unique string used to differentiate your project from all others in Google Cloud. After you enter a project name, the Google Cloud console generates a unique project ID that can be a combination of letters, numbers, and hyphens.

Go to the Dashboard page in the Google Cloud console to get your project ID.



Project info

Project name
My Sample Project

Project ID
my-sample-project-191923

Project number
314053285323

Project info card on the Dashboard page

Submission form

The customer is required to send a GCP organization ID and a GCP project ID to the Google Team. The Google team will add the customer's organization ID to the allow list and give the data access permission to the Service Account in the customer's project.

Please submit your IDs via this Google Form.