

## My Steps:

1. Github creating repo.
2. Creating separated modules
3. terraform code development
  - a. Add Credentials (key and gcloud auth)
    - i. gcloud config set project PROJECT\_ID
    - ii. gcloud auth application-default login
  - b. Enabling the necessary APIs: Google Compute Engine API, Kubernetes Engine API, ArtifactRegistry API and Cloud Resource Manager API.
  - c. Adding new SA “ master“ with a key added to the provider  
gcloud iam service-accounts create master --description="Terraform Service account For Developers Final Project" --display-name="Terraform Service Account"
  - d. Setting project's default location

testingservices ▼

Search (/) for resources, docs, products and more

Settings

Usage export

Daily usage reports export as CSV files to a Cloud Storage location of your choice. A Google Service Account will be granted write access to this location. [Learn more](#)

☐ Enable usage export

Default location

Newly created resources, like VM instances, will be deployed to the selected region and/or zone

Region

us-east1 (South Carolina) ▼ ?

Zone

us-east1-b ▼ ?

- e. You will also need to enable some APIs in order to use terraform:
- f. gcloud services enable cloudresourcemanager.googleapis.com

- g. gcloud services enable container.googleapis.com
- h. gcloud services enable artifactregistry.googleapis.com
- i. gcloud services enable compute.googleapis.com
  
- j. Creating VM
  - i. To ssh:
    - gcloud auth activate-service-account  
--key-file=master-terraform-project-iti.json
  
    - gcloud compute ssh management-vm --zone=us-east1-b  
--project=terraform-project-iti --tunnel-through-iap
  
    - Installing prerequisites in the VM like docker, kubectl. And authenticating Artifact Registry. (Could be found in Compute/scripts.sh)
      - gcloud auth configure-docker us-east1-docker.pkg.dev

### **Workload Subnet**

1. Creating the network using terraform.
2. Writing the yaml files for nodeJS and mongoDB