Setup Terraform for GCP Deployments.

Overview

This document shows how to setup terraform, configure the Google provider for terraform (so resources may be deployed to GCP), deploy a test resource and then destroy it.

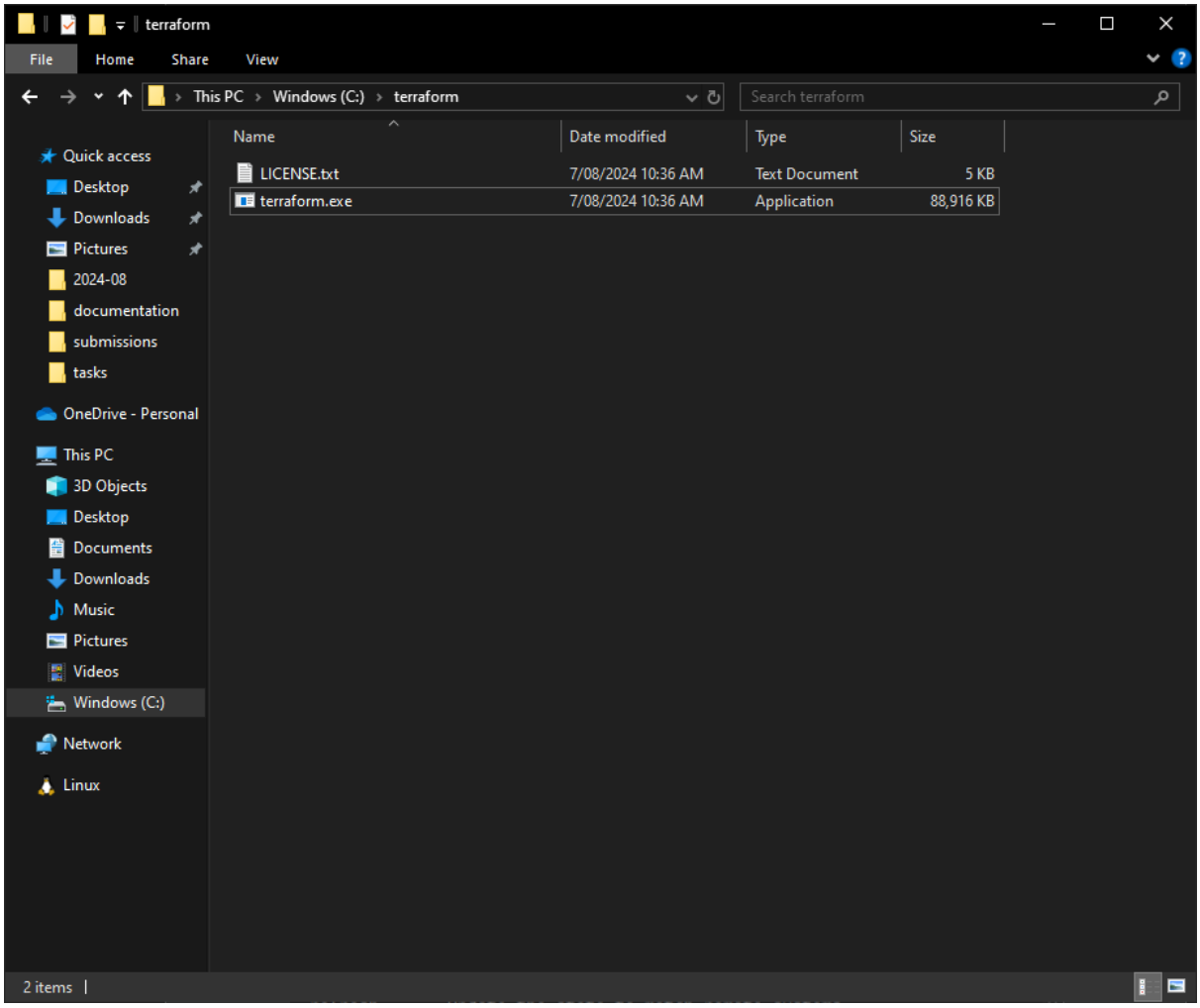
Using Terraform on Windows involves downloading the binary and running it from a CLI, such as the Windows command prompt or git bash. The binary can be used straight up if it or a symlink to is within the current working directory, though it is much easier to use if the directory containing the executable is added to PATH for the user or system.

The Google provider for Terraform uses the gcloud CLI to interact with GCP and deploy resources. This too will require installation. Once authentication is completed through the gcloud cli Terraform can be used to deploy resources to GCP.

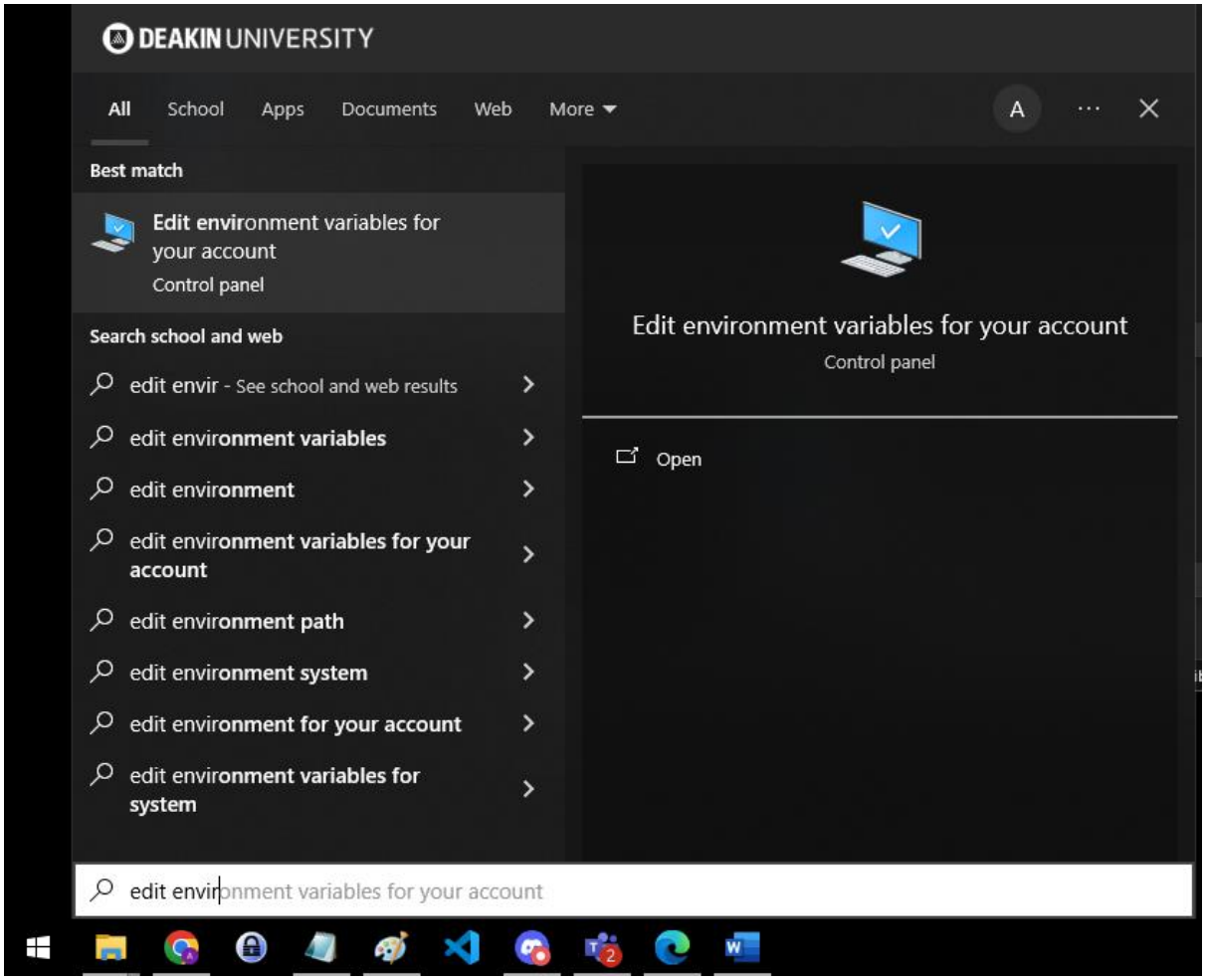
Terraform Installation (windows)

Download the terraform executable (zipped) from the Hashicorp Terraform download page. Unzip it to a directory and make a note of this directory. I put the contents into C:\terraform

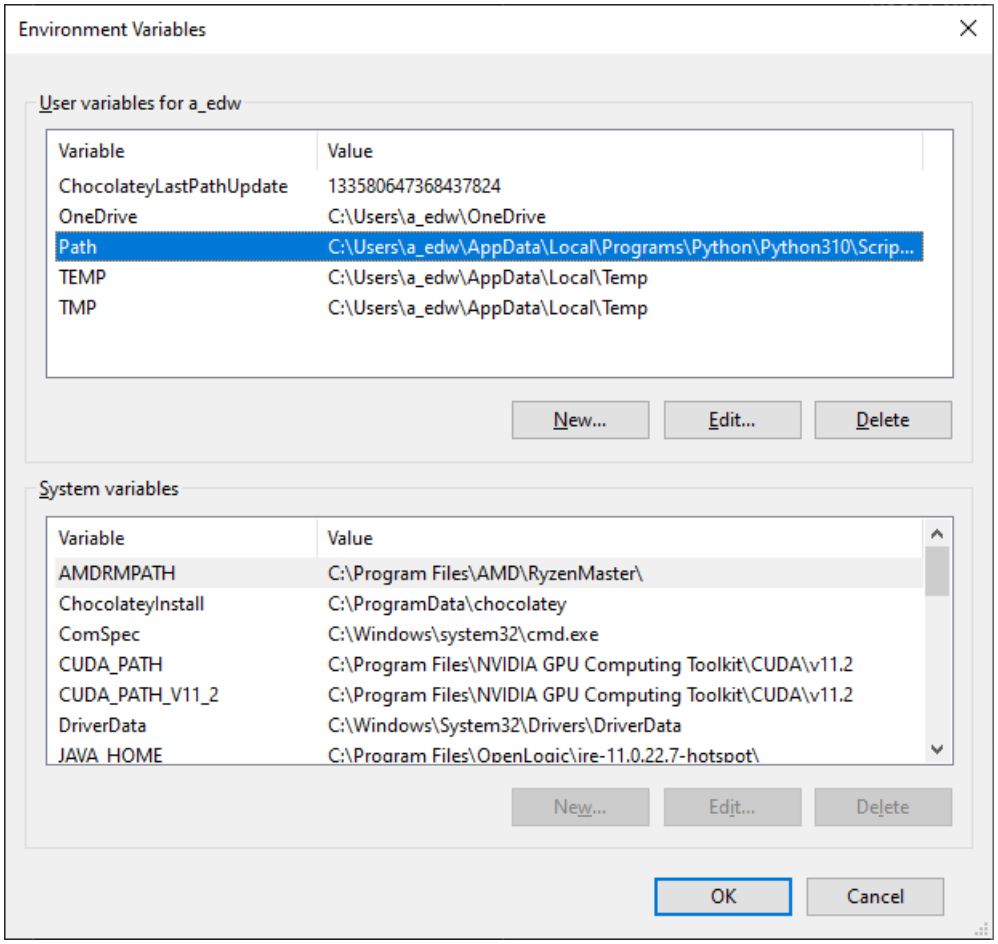
- Hashicorp Terraform download page https://developer.hashicorp.com/terraform/install?product_intent=terraform

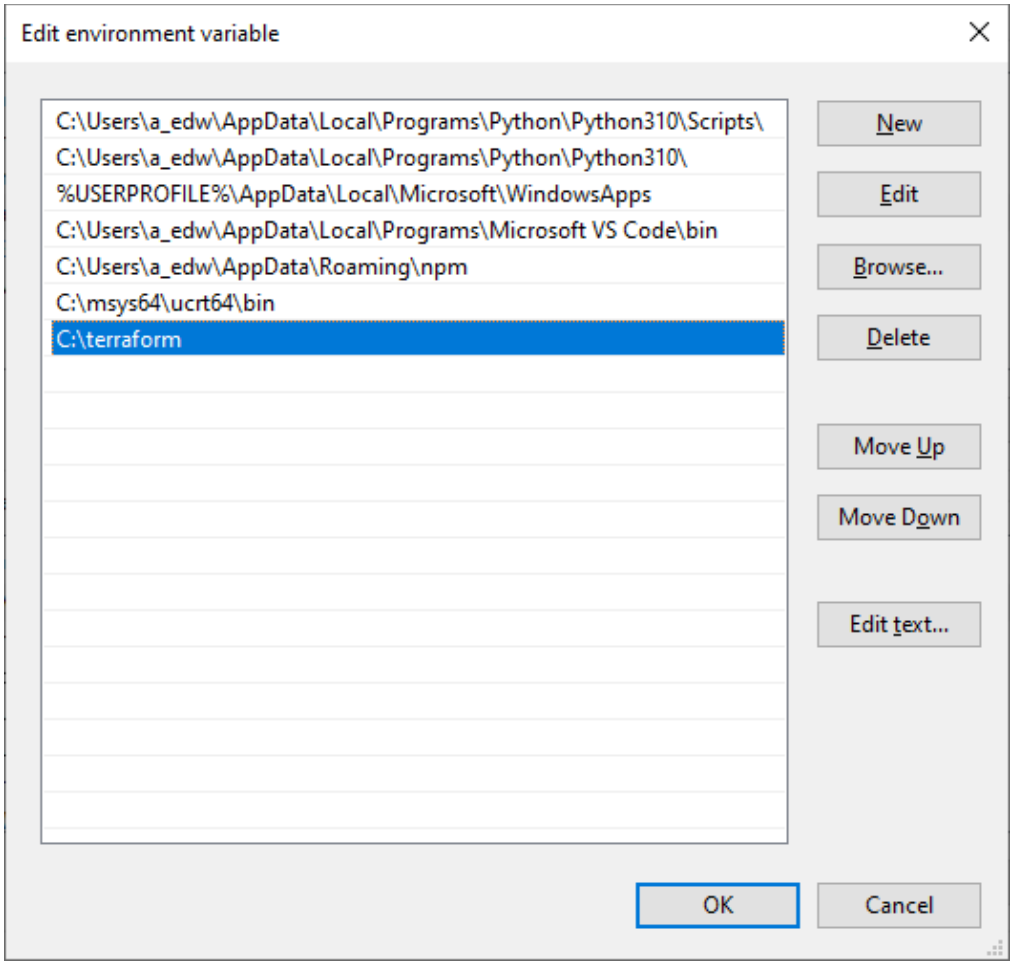


Add terraform to path for user by going to Edit environment variables for your account.



Edit Path and add an item. Add the directory containing the terraform executable.





Click OK and a coupe of times to exit editing environment variables.

Terraform CLI should now be available to run within a command prompt.

Confirm Terraform CLI works as expected by going to a command prompt, typing terraform and pressing enter. As no further commands were defined Terraform should show information about using it.

```
Select Command Prompt
Microsoft Windows [Version 10.0.19045.4651]
(c) Microsoft Corporation. All rights reserved.

C:\Users\a_edw>terraform
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init          Prepare your working directory for other commands
  validate      Check whether the configuration is valid
  plan          Show changes required by the current configuration
  apply         Create or update infrastructure
  destroy       Destroy previously-created infrastructure

All other commands:
  console       Try Terraform expressions at an interactive command prompt
  fmt           Reformat your configuration in the standard style
  force-unlock  Release a stuck lock on the current workspace
  get           Install or upgrade remote Terraform modules
  graph         Generate a Graphviz graph of the steps in an operation
  import        Associate existing infrastructure with a Terraform resource
  login         Obtain and save credentials for a remote host
  logout        Remove locally-stored credentials for a remote host
  metadata      Metadata related commands
  output        Show output values from your root module
  providers     Show the providers required for this configuration
  refresh       Update the state to match remote systems
```

Gcloud CLI installation (Windows)

Download the gcloud cli installer for Windows. Instructions for installation are available on the download page.

- Google gcloud CLI download page https://cloud.google.com/sdk/docs/install

Run through the installer following the prompts.

Once the installer has completed running open a command prompt and run:

gcloud auth application-default login

The above line will either open your browser or provide a link to open. Follow the prompts to authenticate with Google.

```
C:\Windows\SYSTEM32\cmd.exe
C:\Users\a_edw\AppData\Local\Google\Cloud SDK>gcloud auth application-default login
Python was not found; run without arguments to install from the Microsoft Store, or disable this shortcut from Settings
> Manage App Execution Aliases.
Your browser has been opened to visit:

    https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=764086051850-6qr4p6gpi6hn506pt8ejuq83di341hur
.apps.googleusercontent.com&redirect_uri=http%3A%2F%2Flocalhost%3A8085%2F&scope=openid+https%3A%2F%2Fwww.googleapis.com%
2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform+https%3A%2F%2Fwww.googleapis.com%2Fauth
%2Fsqlservice.login&state=mPEil0RwLGvCPnL8aVdYnDDnCIWzmB&access_type=offline&code_challenge=ZbbF7NACPdlHB-M7ZC9ZomK34kEH
lV75c90cTH_P5TA&code_challenge_method=S256


Credentials saved to file: [C:\Users\a_edw\AppData\Roaming\gcloud\application_default_credentials.json]

These credentials will be used by any library that requests Application Default Credentials (ADC).

Quota project "sit-23t1-project-echo-25288b9" was added to ADC which can be used by Google client libraries for billing
and quota. Note that some services may still bill the project owning the resource.

C:\Users\a_edw\AppData\Local\Google\Cloud SDK>
```

Once done Terraform will be authenticated and can now be ran.

First GCP Deployment Using Terraform

Create a new folder and within it create a file named “example.tf”. To it add the following code.

```
provider "google" {
  project = "sit-23t1-project-echo-25288b9" # replace with your project name if different
  region  = "australia-southeast2"
  zone    = "australia-southeast2-a"
}

resource "google_compute_network" "vpc_network" {
  name                    = "terraform-network"
  auto_create_subnetworks = "true"
}

resource "google_compute_instance" "vm_instance" {
  name         = "terraform-instance"
  machine_type = "e2-micro"

  boot_disk {
    initialize_params {
      image = "debian-cloud/debian-11"
    }
  }

  network_interface {
    # A default network is created for all GCP projects
    network =  google_compute_network.vpc_network.id
    access_config {
    }
  }
}
```

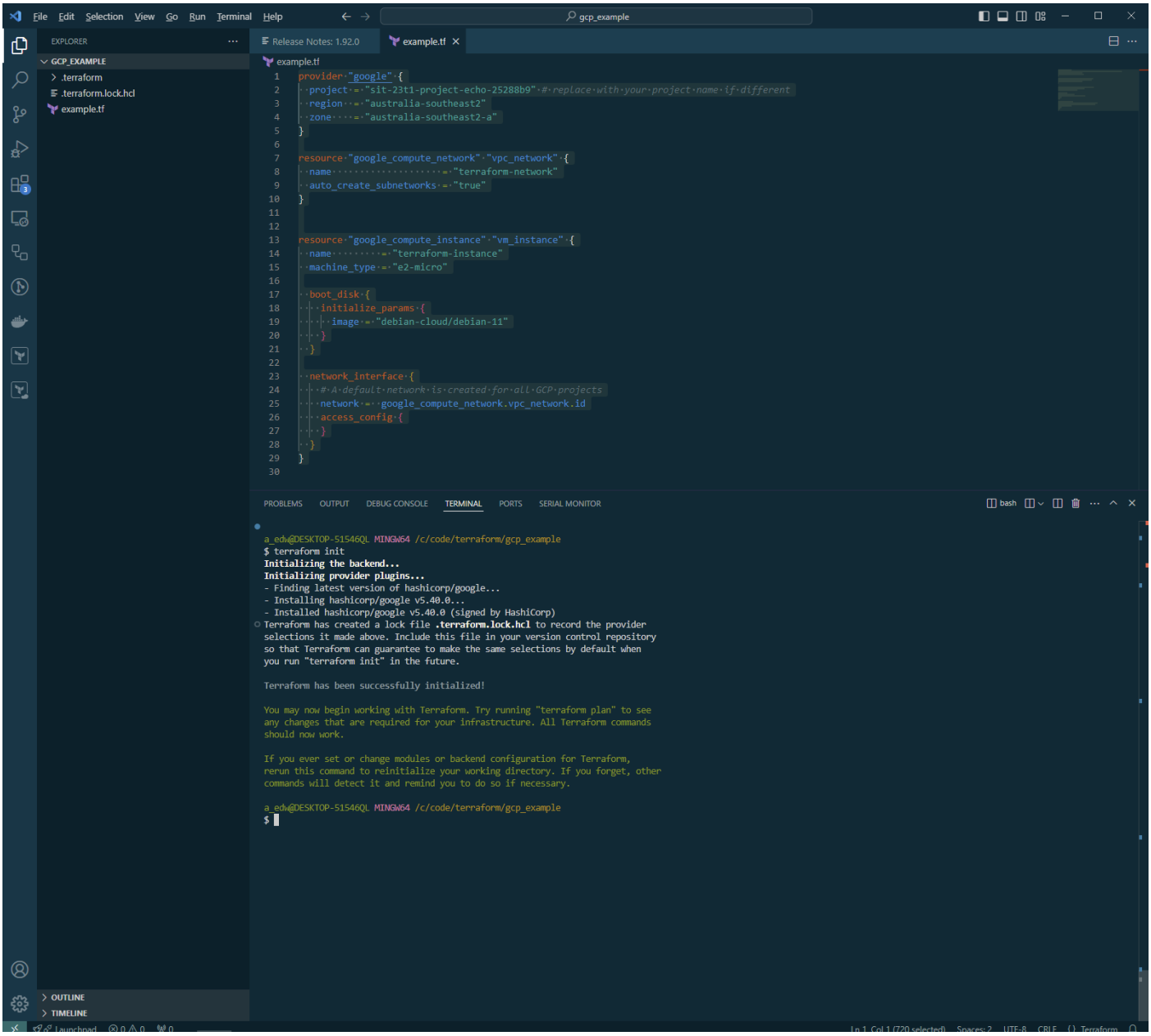
The code defines resources to create a VPC and a Compute Engine Virtual Machine within it. The provider specifies our project within GCP.

Perform the following commands in order:

- terraform init
  - This initialises terraform for the directory including downloading the Google provider executable.
- terraform plan
  - This will take actions to plan the deployment of the resources to GCP and it will provide output showing what it creates.
- terraform apply
  - This is the command to deploy the resources to GCP. It will ask for confirmation of yes to be input before deployment occurs.
- terraform apply -destroy
  - This command will clean up/destroy the resources that were deployed. It too will ask for confirmation to continue.

Examples

init



plan

File Edit Selection View Go Run Terminal Help

gcp_example

EXPLORER

GCP_EXAMPLE

- .terraform
- .terraform.lock.hcl
- example.tf

Release Notes: 1.92.0

example.tf

```tf
provider "google" {
  project = "sit-23t1-project-echo-25288b9" # replace with your project name if different
  region  = "australia-southeast2"
  zone    = "australia-southeast2-a"
}

resource "google_compute_network" "vpc_network" {
  name                    = "terraform-network"
  auto_create_subnetworks = "true"
}


resource "google_compute_instance" "vm_instance" {
  name         = "terraform-instance"
  machine_type = "e2-micro"

  boot_disk {
    initialize_params {
      image = "debian-cloud/debian-11"
    }
  }

  network_interface {
    # A default network is created for all GCP projects
    network =  google_compute_network.vpc_network.id
    access_config {
    }
  }
}

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SERIAL MONITOR

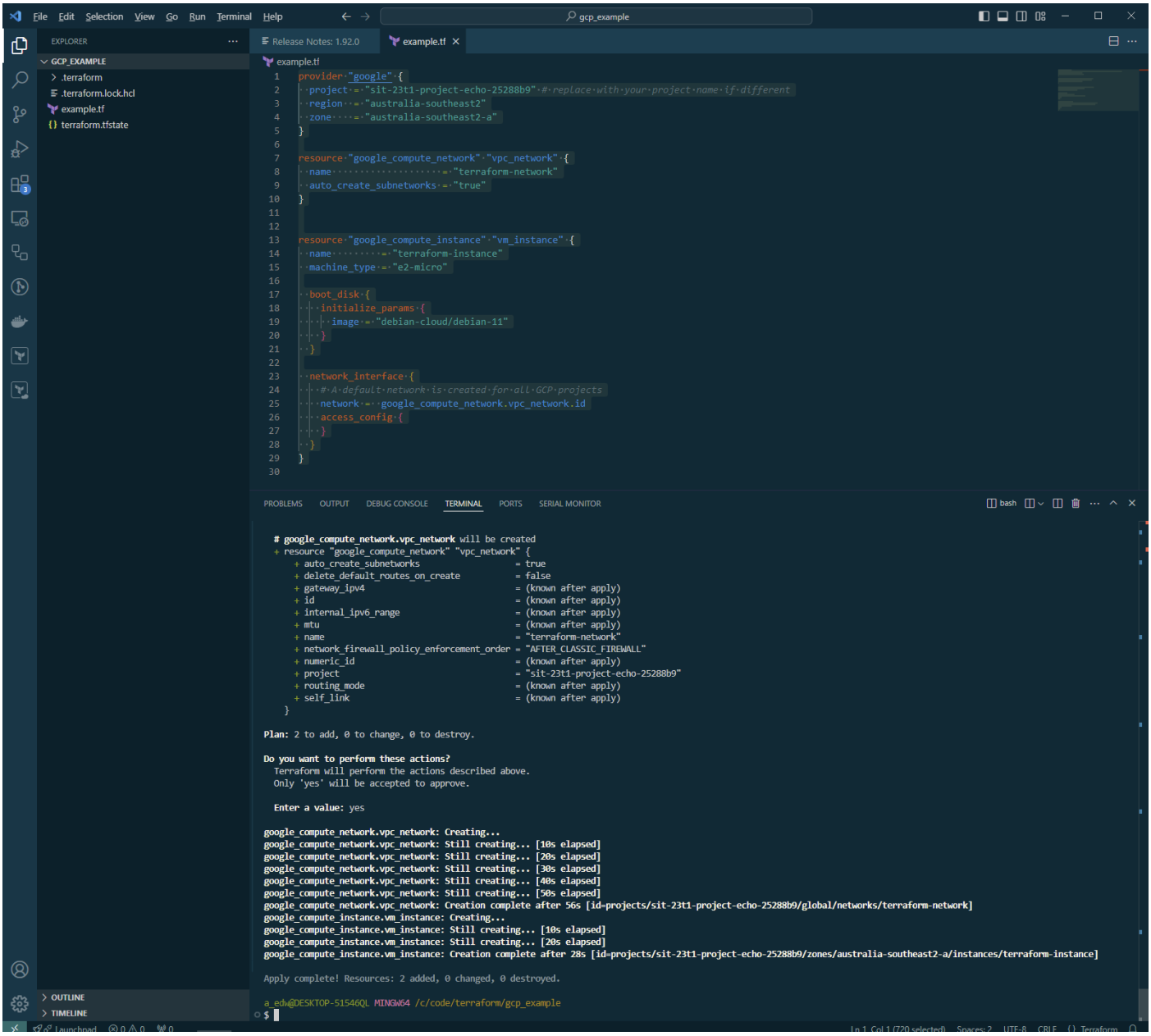
bash

```
      + subnetwork                  = (known after apply)
      + subnetwork                  = (known after apply)
      + subnetwork_project          = (known after apply)
      + subnetwork_project          = (known after apply)

      + access_config {
          + nat_ip       = (known after apply)
          + network_tier = (known after apply)
        }
    }

    + reservation_affinity (known after apply)

    + scheduling (known after apply)
  }

  # google_compute_network.vpc_network will be created
  + resource "google_compute_network" "vpc_network" {
      + auto_create_subnetworks                   = true
      + delete_default_routes_on_create           = false
      + gateway_ipv4                              = (known after apply)
      + id                                        = (known after apply)
      + internal_ipv6_range                       = (known after apply)
      + mtu                                       = (known after apply)
      + name                                      = "terraform-network"
      + network_firewall_policy_enforcement_order = "AFTER_CLASSIC_FIREWALL"
      + numeric_id                                = (known after apply)
      + project                                   = "sit-23t1-project-echo-25288b9"
      + routing_mode                              = (known after apply)
      + self_link                                 = (known after apply)
  }

Plan: 2 to add, 0 to change, 0 to destroy.


Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

a_edu@DESKTOP-51546QL MINGW64 /c/code/terraform/gcp_example
$
```

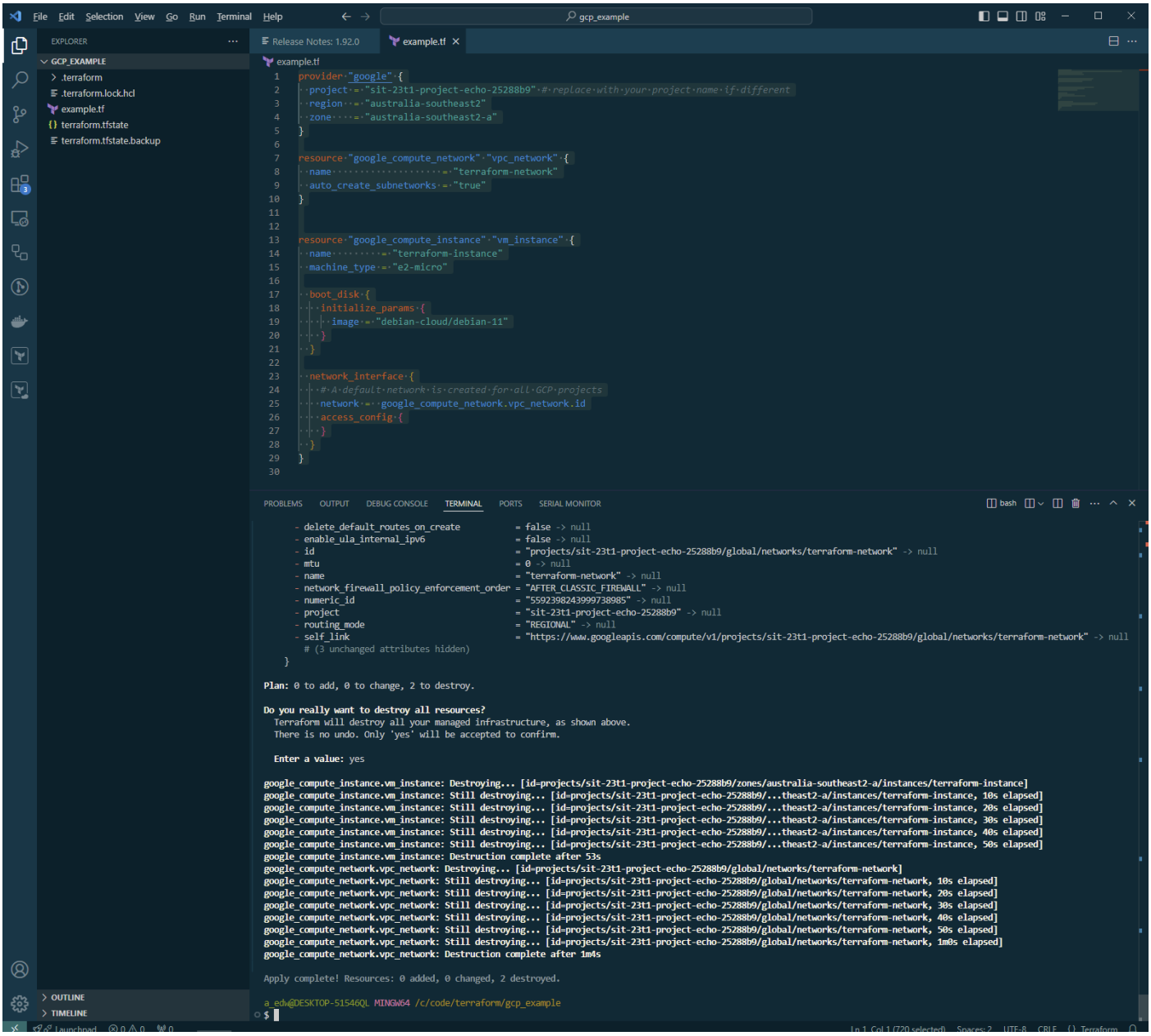
OUTLINE

TIMELINE

apply


File Edit Selection View Go Run Terminal Help
gcp_example
EXPLORER
GCP_EXAMPLE
.terraform
.terraform.lock.hcl
example.tf
terraform.tfstate
Release Notes: 1.92.0
example.tf
provider "google" {
project = "sit-23t1-project-echo-25288b9" # replace with your project name if different
region = "australia-southeast2"
zone = "australia-southeast2-a"
}
resource "google_compute_network" "vpc_network" {
name = "terraform-network"
auto_create_subnetworks = "true"
}
resource "google_compute_instance" "vm_instance" {
name = "terraform-instance"
machine_type = "e2-micro"
boot_disk {
initialize_params {
image = "debian-cloud/debian-11"
}
}
network_interface {
# A default network is created for all GCP projects
network = google_compute_network.vpc_network.id
access_config {
}
}
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SERIAL MONITOR
bash
# google_compute_network.vpc_network will be created
+ resource "google_compute_network" "vpc_network" {
+ auto_create_subnetworks = true
+ delete_default_routes_on_create = false
+ gateway_ipv4 = (known after apply)
+ id = (known after apply)
+ internal_ipv6_range = (known after apply)
+ mtu = (known after apply)
+ name = "terraform-network"
+ network_firewall_policy_enforcement_order = "AFTER_CLASSIC_FIREWALL"
+ numeric_id = (known after apply)
+ project = "sit-23t1-project-echo-25288b9"
+ routing_mode = (known after apply)
+ self_link = (known after apply)
}
Plan: 2 to add, 0 to change, 0 to destroy.
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.
Enter a value: yes
google_compute_network.vpc_network: Creating...
google_compute_network.vpc_network: Still creating... [10s elapsed]
google_compute_network.vpc_network: Still creating... [20s elapsed]
google_compute_network.vpc_network: Still creating... [30s elapsed]
google_compute_network.vpc_network: Still creating... [40s elapsed]
google_compute_network.vpc_network: Still creating... [50s elapsed]
google_compute_network.vpc_network: Creation complete after 56s [id=projects/sit-23t1-project-echo-25288b9/global/networks/terraform-network]
google_compute_instance.vm_instance: Creating...
google_compute_instance.vm_instance: Still creating... [10s elapsed]
google_compute_instance.vm_instance: Still creating... [20s elapsed]
google_compute_instance.vm_instance: Creation complete after 28s [id=projects/sit-23t1-project-echo-25288b9/zones/australia-southeast2-a/instances/terraform-instance]
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
a_edu@DESKTOP-51546QL MINGW64 /c/code/terraform/gcp_example
$
OUTLINE
TIMELINE

apply -destroy



Summary of Installation Steps (the TLDR)

- Download terraform executable and add to PATH
- Download gcloud cli and run gcloud auth application-default login
- Terraform is setup. Create some terraform code and run it to test.

Reference Links

https://developer.hashicorp.com/terraform/install?product_intent=terraform

https://registry.terraform.io/providers/hashicorp/google/latest/docs/guides/getting_started

https://cloud.google.com/sdk/docs/install