Terraform Set-Up and HandsOn

Install `terraform`

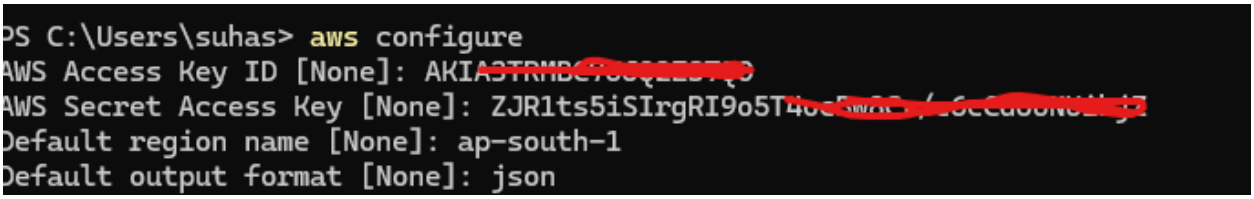
```
choco install terraform -y
```

Install `AWS CLI` Open **PowerShell (Admin)** and run:

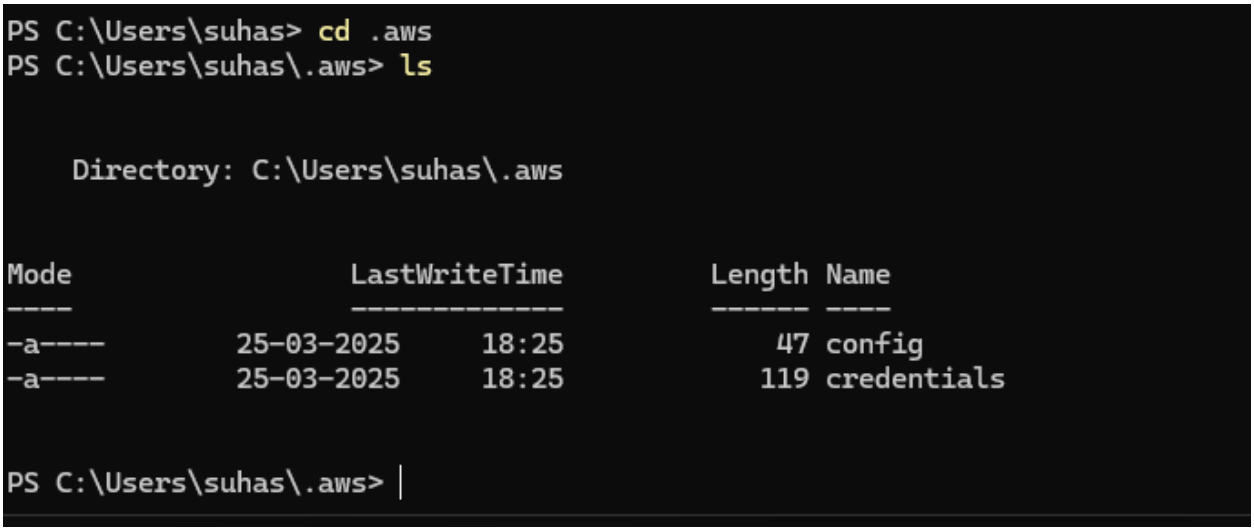
```
choco install awscli -y
```

```
aws --version
```

Configure `aws iam user in local cli`



Credentials will be save in .aws folder



Terraform Initialization

Create a `main.tf` file

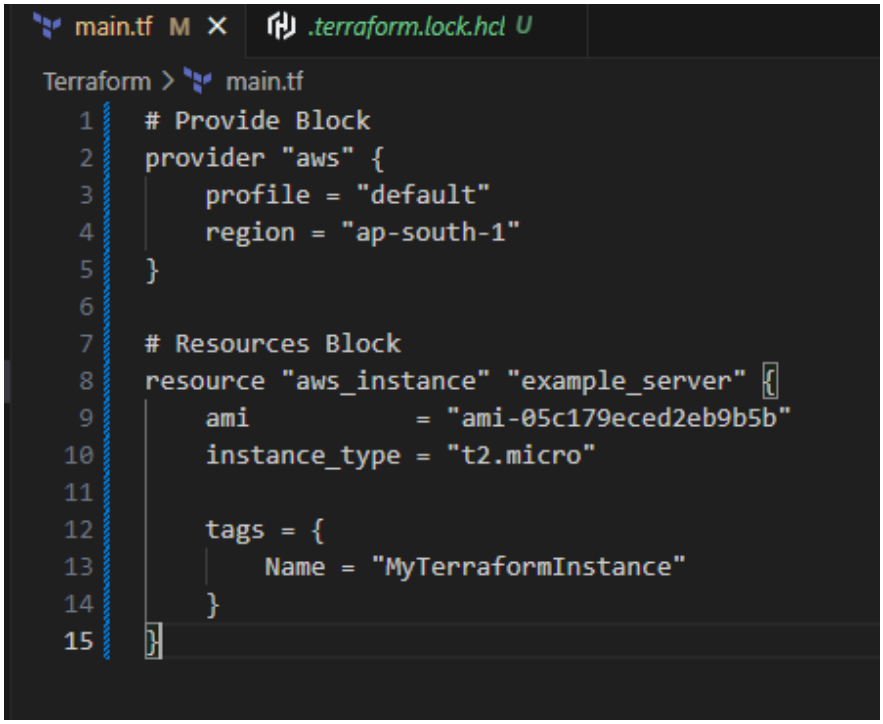
then initialize the terraform follow below image

```
PS D:\Studies\Programming\GIT-Workspace\SRE-and-DevOps-Concepts\Terraform> terraform init
Initializing the backend...
Initializing provider plugins...

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.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS D:\Studies\Programming\GIT-Workspace\SRE-and-DevOps-Concepts\Terraform> terraform plan

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
PS D:\Studies\Programming\GIT-Workspace\SRE-and-DevOps-Concepts\Terraform>
```

```
PS D:\Studies\Programming\GIT-Workspace\SRE-and-DevOps-Concepts\Terraform> terraform destroy

No changes. No objects need to be destroyed.

Either you have not created any objects yet or the existing objects were already deleted outside of Terraform.

Destroy complete! Resources: 0 destroyed.
PS D:\Studies\Programming\GIT-Workspace\SRE-and-DevOps-Concepts\Terraform> terraform apply

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

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

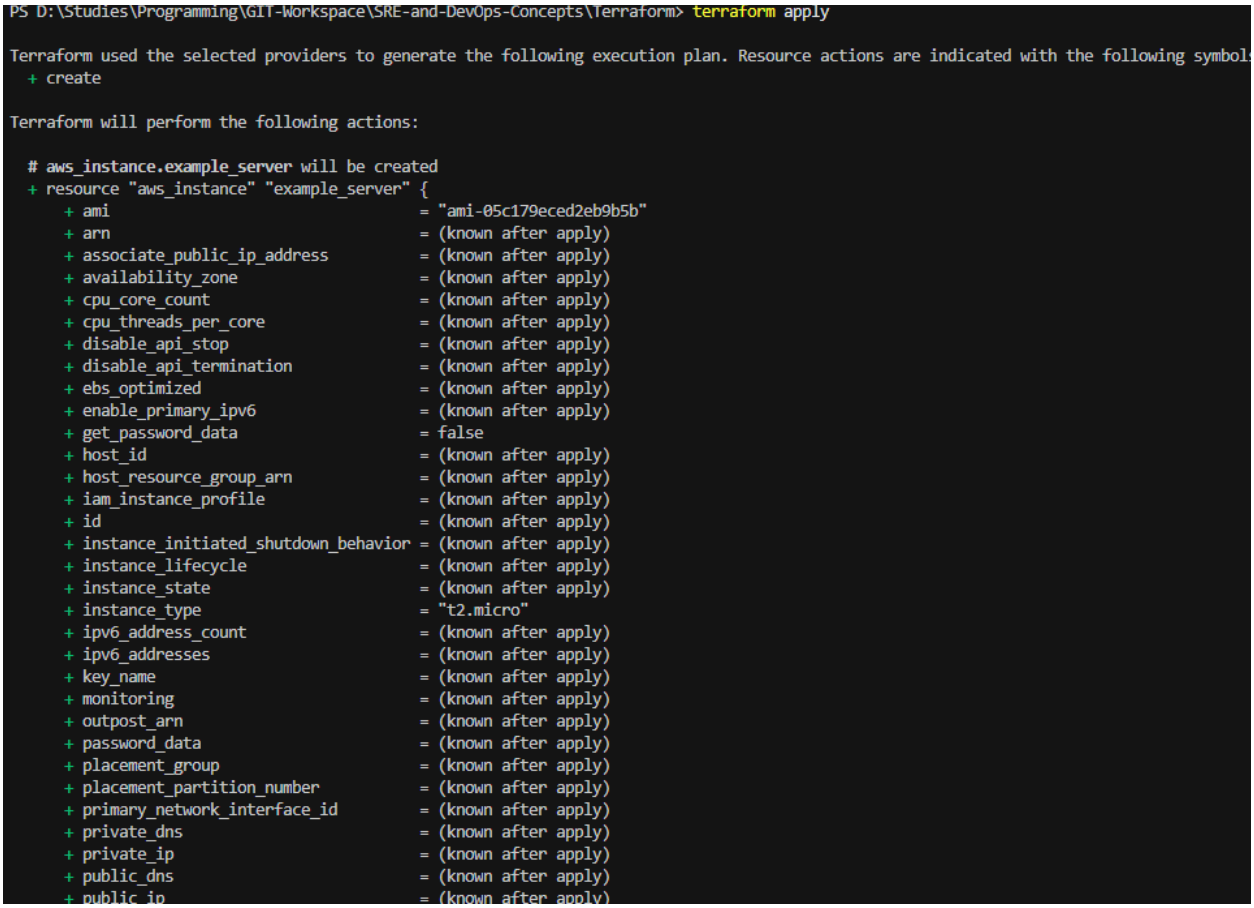
Create a EC2 instance

Line 1-5 AWS configuration,



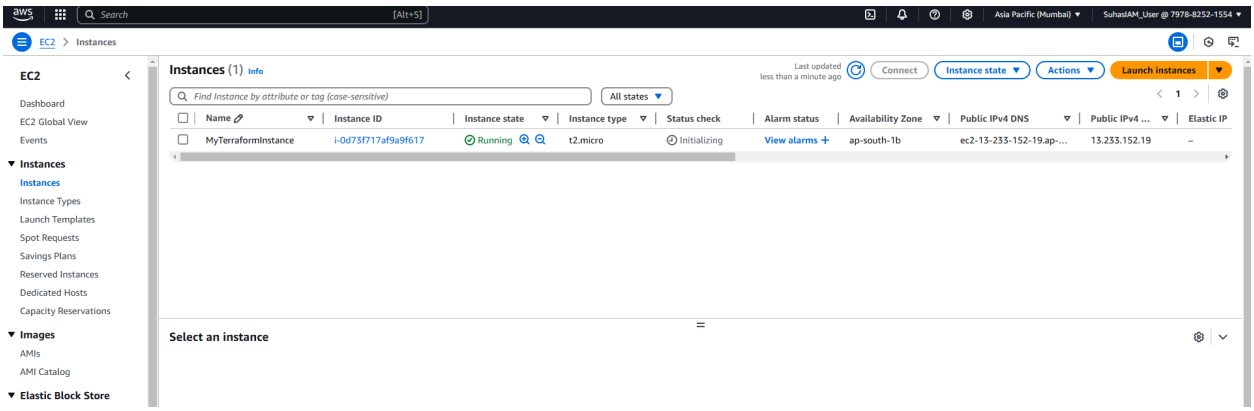
Line 7-15 aws_instance creation for more info on the syntax can refer https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance website .

Then apply the changes and destroy after using



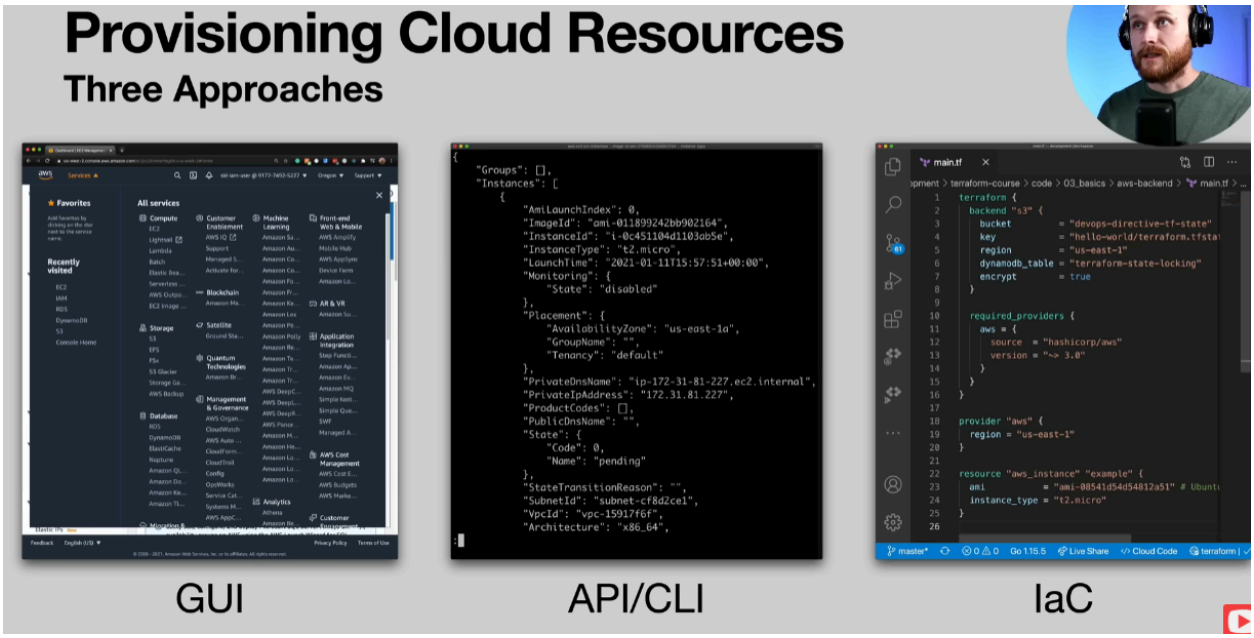
```
PS D:\Studies\Programming\GIT-Workspace\SRE-and-DevOps-Concepts\Terraform> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols
  + create

Terraform will perform the following actions:

  # aws_instance.example_server will be created
  + resource "aws_instance" "example_server" {
      + ami                                  = "ami-05c179eced2eb9b5b"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
```

```
aws_instance.example_server: Creating...
aws_instance.example_server: Still creating... [10s elapsed]
aws_instance.example_server: Creation complete after 12s [id=i-0d73f717af9a9f617]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS D:\Studies\Programming\GIT-Workspace\SRE-and-DevOps-Concepts\Terraform> terraform destroy
aws_instance.example_server: Refreshing state... [id=i-0d73f717af9a9f617]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.example_server will be destroyed
  - resource "aws_instance" "example_server" {
      - ami                                  = "ami-05c179eced2eb9b5b" -> null
      - arn                                  = "arn:aws:ec2:ap-south-1:797882521554:instance/i-0d73f717af9a9f617" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "ap-south-1b" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-0d73f717af9a9f617" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - monitoring                           = false -> null
      - placement_partition_number           = 0 -> null
      - primary_network_interface_id         = "eni-0b79773b3a03025a3" -> null
      - private_dns                          = "ip-172-31-1-188.ap-south-1.compute.internal" -> null
      - private_ip                           = "172.31.1.188" -> null
      - public_dns                           = "ec2-13-233-152-19.ap-south-1.compute.amazonaws.com" -> null
      - public_ip                            = "13.233.152.19" -> null
      - secondary_private_ips                = [] -> null
      - security_groups                      = [
          - "default",
        ] -> null
```



Tutorial video for reference https://www.youtube.com/watch?v=nvNqfgojocs

```
C:\Users\suhas>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
  modules       Show all declared modules in a working directory
  output        Show output values from your root module
  providers     Show the providers required for this configuration
  refresh       Update the state to match remote systems
  show          Show the current state or a saved plan
  state         Advanced state management
  taint         Mark a resource instance as not fully functional
  test          Execute integration tests for Terraform modules
  untaint       Remove the 'tainted' state from a resource instance
  version       Show the current Terraform version
  workspace     Workspace management

Global options (use these before the subcommand, if any):
  -chdir=DIR    Switch to a different working directory before executing the
                given subcommand.
  -help         Show this help output, or the help for a specified subcommand.
  -version      An alias for the "version" subcommand.

C:\Users\suhas>
```


Provisioning Cloud Resources
Three Approaches
GUI
API/CLI
IaC