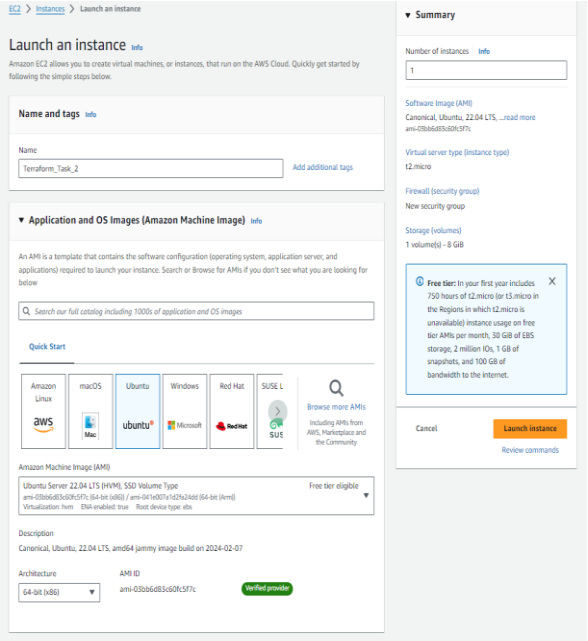
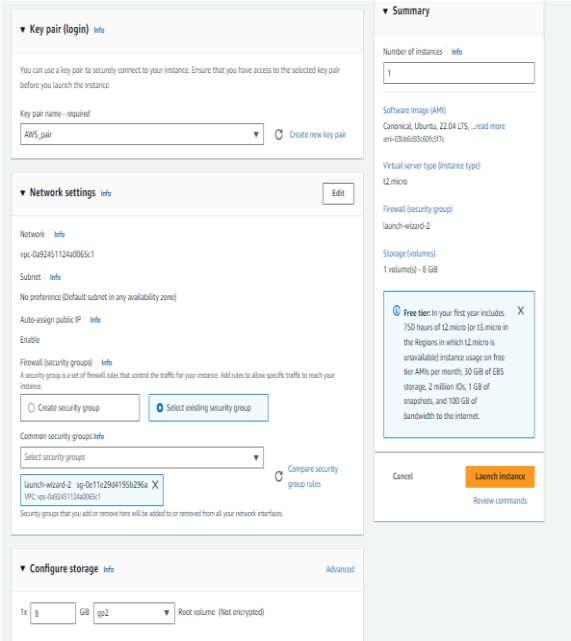
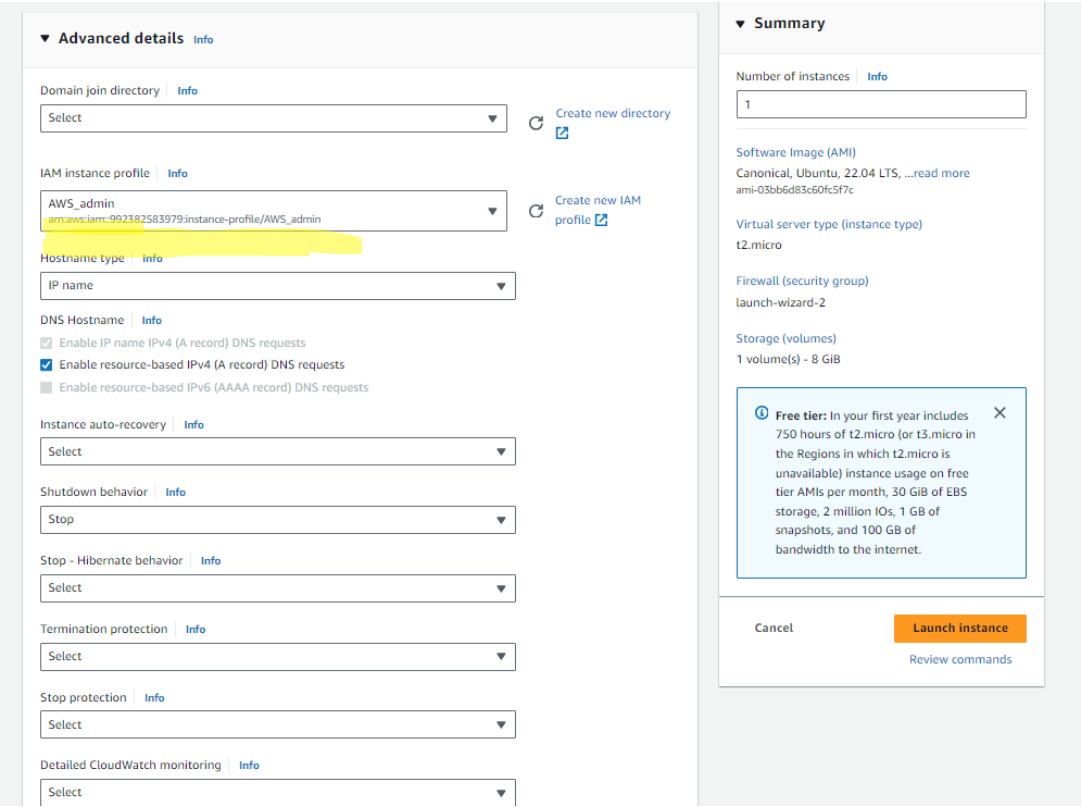
Launch an ec2 instance under a default subnet and VPC using terraform template.

Step 1: creating the Ec2 instance with IAM Role.







Step 2: connect instance using putty and update it .

```
root@ip-172-31-38-223: /home/ubuntu
amd64 Packages [42.1 kB]
Get:19 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse
Translation-en [10.1 kB]
Get:20 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse
amd64 c-n-f Metadata [472 B]
Get:21 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd6
4 Packages [41.7 kB]
Get:22 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Tran
slation-en [10.5 kB]
Get:23 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd6
4 c-n-f Metadata [388 B]
Get:24 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/restricte
d amd64 c-n-f Metadata [116 B]
Get:25 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe
amd64 Packages [24.3 kB]
Get:26 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe
Translation-en [16.5 kB]
Get:27 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe
amd64 c-n-f Metadata [644 B]
Get:28 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/multivers
e amd64 c-n-f Metadata [116 B]
Get:29 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [120
8 kB]
81% [4 Packages store 0 B] [29 Packages 181 kB/1208 kB 15%]
```

Step 3: Install the terraform using following commands.

wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg

echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list

sudo apt update && sudo apt install terraform

```
root@ip-172-31-38-223: /home/ubuntu
root@ip-172-31-38-223:/home/ubuntu# wget -O- https://apt.releases.hashicorp.com/
gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://
apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.
list.d/hashicorp.list
sudo apt update && sudo apt install terraform
--2024-03-05 06:34:20--  https://apt.releases.hashicorp.com/gpg
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 18.172.78.3
0, 18.172.78.65, 18.172.78.129, ...
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|18.172.78.
30|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3980 (3.9K) [binary/octet-stream]
Saving to: 'STDOUT'

-                   100%[===================>]   3.89K  --.-KB/s    in 0s

2024-03-05 06:34:20 (101 MB/s) - written to stdout [3980/3980]

deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.re
leases.hashicorp.com jammy main
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
```

Step 4: Check the version of the terraform.

```
root@ip-172-31-38-223: /home/ubuntu
root@ip-172-31-38-223:/home/ubuntu# terraform --version
Terraform v1.7.4
on linux_amd64
root@ip-172-31-38-223:/home/ubuntu#
```

Step 5: Create a directory and Create .tf file.

```
root@ip-172-31-38-223: ~/task
root@ip-172-31-38-223:~# mkdir task
root@ip-172-31-38-223:~# cd task
root@ip-172-31-38-223:~/task# vi terraform.tf
root@ip-172-31-38-223:~/task#
```

Step 6: Write the terraform script for creating Ec2 instance with default VPC and subnet.

```
root@ip-172-31-38-223: ~/task
root@ip-172-31-38-223:~/task# vi terraform.tf
root@ip-172-31-38-223:~/task# cat terraform.tf
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.38.0"
    }
  }
}
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "Guvi_Terraform_Tast2" {
  ami           = "ami-0ab84d9093b4c0d81"
  instance_type = "t2.micro"
  vpc_security_group_ids = ["sg-0cd50456cb90e5c5b"]
  subnet_id              = "subnet-0c0c732a7d7d2c1f2"
}
root@ip-172-31-38-223:~/task#
```

Step 7: Run the terraform init command.

```
root@ip-172-31-38-223: ~/task
root@ip-172-31-38-223:~/task# terraform init

Initializing the backend...
Initializing modules...
Downloading registry.terraform.io/terraform-aws-modules/ec2-instance/aws 5.6.0 f
or ec2_instance...
- ec2_instance in .terraform/modules/ec2_instance

Initializing provider plugins...
- Finding hashicorp/aws versions matching ">= 4.66.0, 5.38.0"...
- Installing hashicorp/aws v5.38.0...
- Installed hashicorp/aws v5.38.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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.
root@ip-172-31-38-223:~/task#
```

Step 8: Run the terraform plan command.

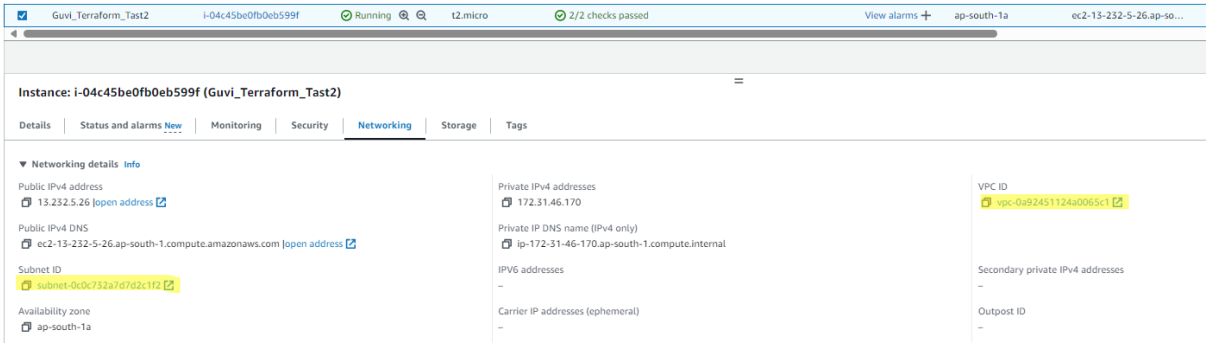
```
root@ip-172-31-38-223: ~/task
root@ip-172-31-38-223:~/task# terraform plan
module.ec2_instance.data.aws_partition.current: Reading...
module.ec2_instance.data.aws_partition.current: Read complete after 0s [id=aws]

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:

  # module.ec2_instance.aws_instance.this[0] will be created
  + resource "aws_instance" "this" {
      + ami                                  = "ami-0ab84d9093b4c0d81"
      + 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)
      + 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                             = "user1"
      + 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)
```

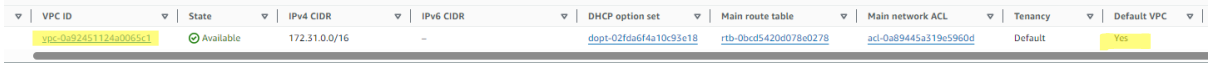
```
root@ip-172-31-38-223: ~/task
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + spot_instance_request_id             = (known after apply)
      + subnet_id                            = "subnet-0c0c732a7d7d2c1f2"
      + tags                                 = {
          + "Name" = "Guvi_Terraform_Tast2"
        }
      + tags_all                             = {
          + "Name" = "Guvi_Terraform_Tast2"
        }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + volume_tags                          = {
          + "Name" = "Guvi_Terraform_Tast2"
        }
      + vpc_security_group_ids               = [
          + "sg-0cd50456cb90e5c5b",
        ]

      + credit_specification (}

      + enclave_options {
          + enabled = (known after apply)
        }

      + metadata_options {
          + http_endpoint               = "enabled"
          + http_protocol_ipv6          = "disabled"
          + http_put_response_hop_limit = 1
          + http_tokens                 = "optional"
          + instance_metadata_tags      = (known after apply)
        }

      + timeouts (}
    }

Plan: 1 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.
root@ip-172-31-38-223:~/task#
```

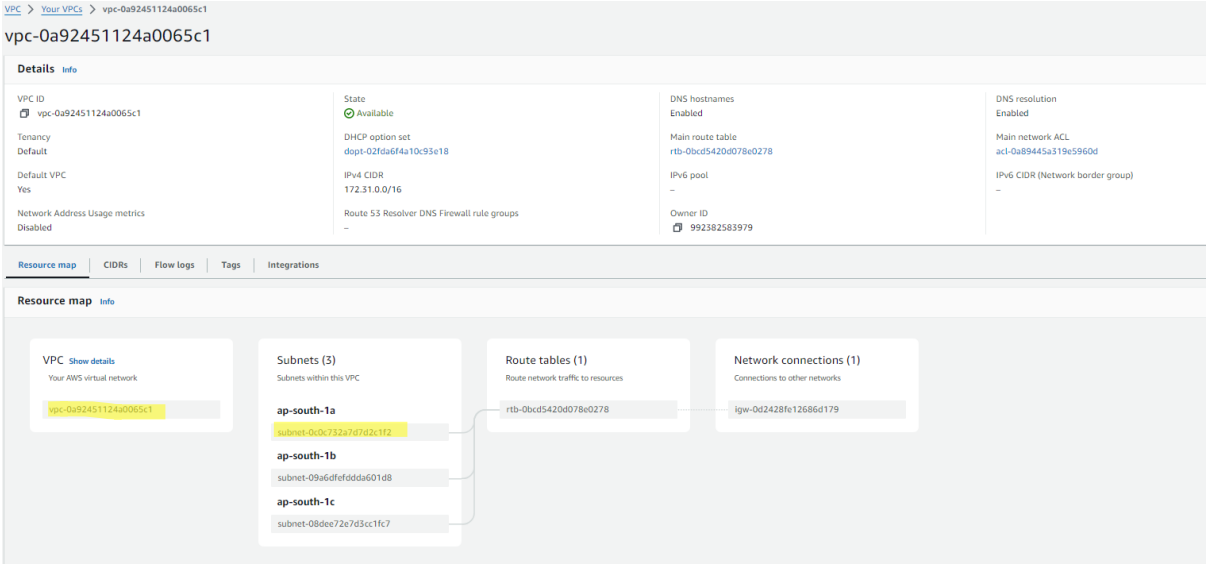
Step 9: Run the terraform apply command:

```
ubuntu@ip-172-31-38-223: ~
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + volume_tags                          = {
          + "Name" = "Guvi_Terraform_Tast2"
        }
      + vpc_security_group_ids               = [
          + "sg-073ba201ad5f0c284",
        ]

      + credit_specification (}

      + enclave_options {
          + enabled = (known after apply)
        }

      + metadata_options {
          + http_endpoint               = "enabled"
          + http_protocol_ipv6          = "disabled"
          + http_put_response_hop_limit = 1
          + http_tokens                 = "optional"
          + instance_metadata_tags      = (known after apply)
        }

      + timeouts (}
    }

Plan: 1 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

module.ec2_instance.aws_instance.this[0]: Creating...
module.ec2_instance.aws_instance.this[0]: Still creating... [10s elapsed]
module.ec2_instance.aws_instance.this[0]: Still creating... [20s elapsed]
module.ec2_instance.aws_instance.this[0]: Still creating... [30s elapsed]
module.ec2_instance.aws_instance.this[0]: Creation complete after 32s [id=i-04c45be0fb0eb
599f]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
root@ip-172-31-38-223:~/task#
```

Step 10: Check the Ec2 Instance created with default VPN and subnet.







root@ip-172-31-38-223: /home/ubuntu/task

```
      - maintenance_options {
          - auto_recovery = "default" -> null
        }

      - metadata_options {
          - http_endpoint               = "enabled" -> null
          - http_protocol_ipv6          = "disabled" -> null
          - http_put_response_hop_limit = 1 -> null
          - http_tokens                 = "optional" -> null
          - instance_metadata_tags      = "disabled" -> null
        }

      - private_dns_name_options {
          - enable_resource_name_dns_a_record    = false -> null
          - enable_resource_name_dns_aaaa_record = false -> null
          - hostname_type                        = "ip-name" -> null
        }

      - root_block_device {
          - delete_on_termination = true -> null
          - device_name           = "/dev/xvda" -> null
          - encrypted             = false -> null
          - iops                  = 3000 -> null
          - tags                  = {} -> null
          - throughput            = 125 -> null
          - volume_id             = "vol-0dc4fe82396b1b8ac" -> null
          - volume_size           = 8 -> null
          - volume_type           = "gp3" -> null
        }

      - timeouts {}
    }

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

module.ec2_instance.aws_instance.this[0]: Destroying... [id=i-0e8640203aea35f90]
```