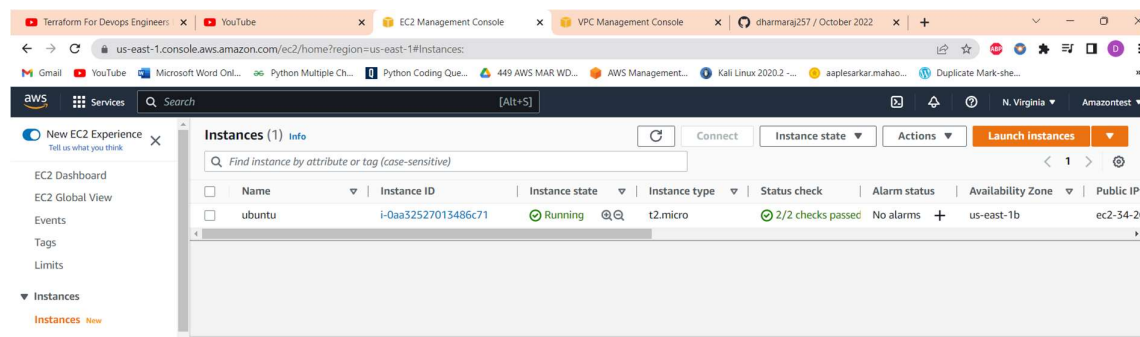


Project: create a simple project creating AWS RDS db instance using terraform resource.

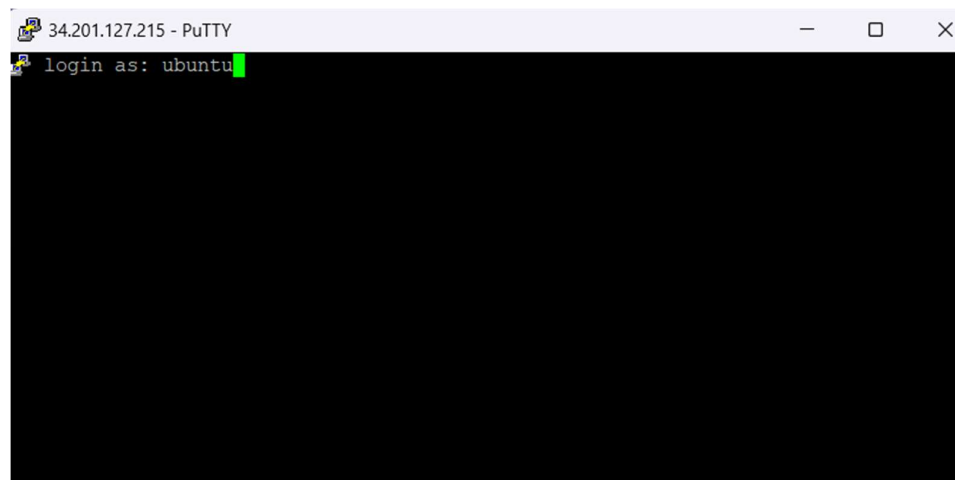
1. Create a ubuntu server

- I. launch instance and give name ubuntu.
- ii. select ubuntu and t2. micro-CPU
- iii. download a new keypair.
- iv. select default VPC
- v. In security group allows ssh, https and http allow anywhere.
- vi. launch instance.



2. Log in to the ubuntu server using putty

- I. copy the public IPv4 and paste on the putty.
- II. Select ssh and go to the authentication and upload the ppk file
- III. Then put password ubuntu and log in.



3. Update the server

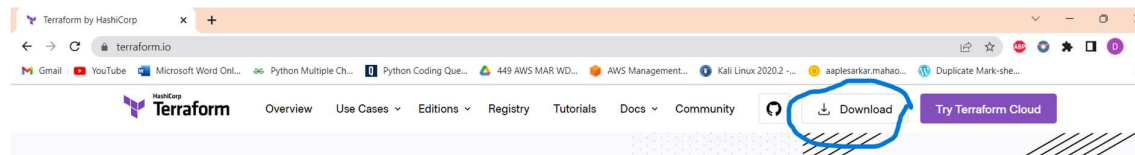
I. use to update server use command

II. Sudo apt update -y

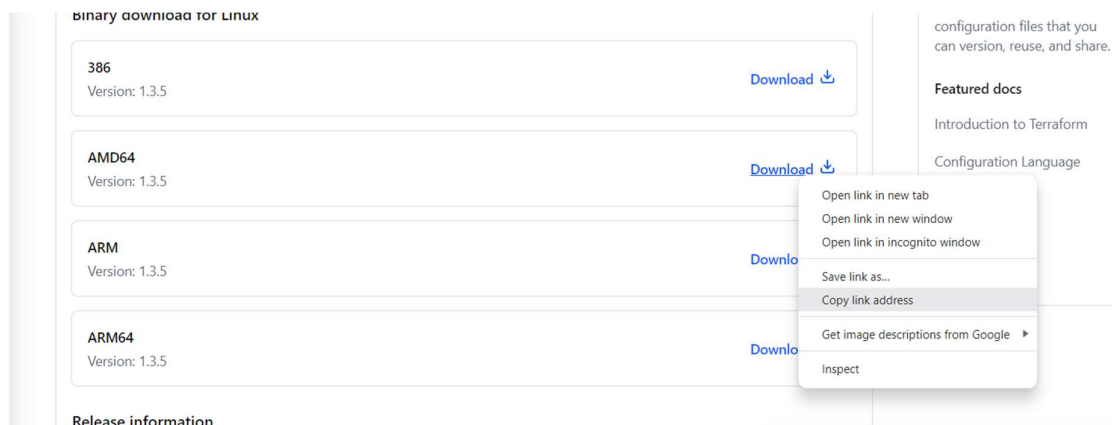
```
ubuntu@ip-172-31-84-44: ~  
ubuntu@ip-172-31-84-44:~$ sudo apt update -y  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]  
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]  
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]  
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]  
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]  
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
```

4. Install the terraform on the server.

I. visit <https://www.terraform.io/> and click the download button



II. select Linux amd64 and copy the link address



III. Paste on the server wget

https://releases.hashicorp.com/terraform/1.3.4/terraform_1.3.4_linux_amd64.zip

```
ubuntu@ip-172-31-84-44: ~  
ubuntu@ip-172-31-84-44:~$ wget https://releases.hashicorp.com/terraform/1.3.5/te  
rraform_1.3.5_linux_amd64.zip
```

IV. Sudo apt install unzip -y

```
ubuntu@ip-172-31-84-44: ~  
ubuntu@ip-172-31-84-44:~$ sudo apt install unzip -y  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
Suggested packages:  
  zip  
The following NEW packages will be installed:  
  unzip
```

V. Sudo unzip terraform_1.3.4_linux_amd64.zip

```
ubuntu@ip-172-31-84-44: ~  
ubuntu@ip-172-31-84-44:~$ ls  
terraform_1.3.5_linux_amd64.zip  
ubuntu@ip-172-31-84-44:~$ sudo unzip terraform_1.3.5_linux_amd64.zip  
Archive:  terraform_1.3.5_linux_amd64.zip  
  inflating: terraform  
ubuntu@ip-172-31-84-44:~$
```

VI. Sudo cp terraform /bin

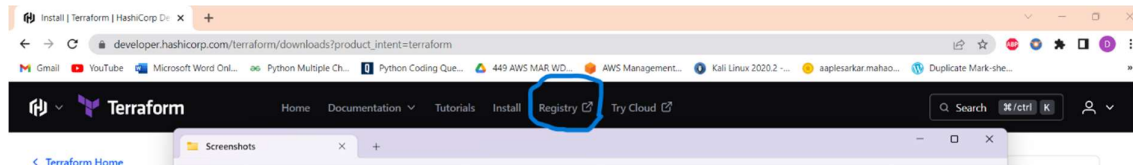
```
ubuntu@ip-172-31-84-44:~$ sudo cp terraform /bin  
ubuntu@ip-172-31-84-44:~$ ls  
terraform terraform_1.3.5_linux_amd64.zip  
ubuntu@ip-172-31-84-44:~$
```

VII. terraform -v

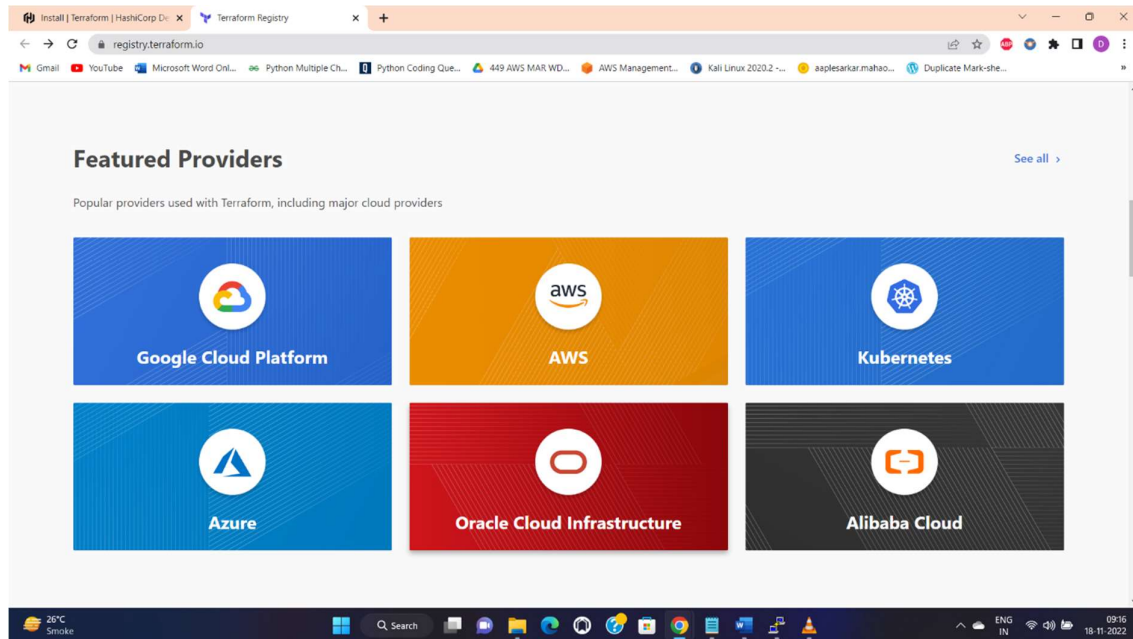
```
ubuntu@ip-172-31-84-44: ~  
ubuntu@ip-172-31-84-44:~$ terraform -v  
Terraform v1.3.5  
on linux_amd64  
ubuntu@ip-172-31-84-44:~$
```

5. Select registry and create rds.tf file

i. Go to the home page and click on the registry



ii. Then select the AWS provider and go to the documentation



iii. search RDS and copy the resource RDS

The screenshot shows the AWS Documentation search interface. The search bar contains 'RDS' and shows '75 matching results'. The 'Resources' tab is selected. On the right, there is a sidebar with 'ON THIS PAGE' links: 'RDS Instance Class Types', 'Example Usage', 'Argument Reference', and 'Attributes Reference'. The main content area shows the 'Example Usage' section for the 'aws_db_instance' resource. It includes a list of related resources on the left and a Terraform configuration snippet on the right.

Example Usage

```
resource "aws_db_instance" "default" {
  allocated_storage = 10
  db_name           = "mydb"
  engine            = "mysql"
  engine_version    = "5.7"
  instance_class     = "db.t3.micro"
  username          = "foo"
  password          = "foobarbaz"
  parameter_group_name = "default.mysql5.7"
  skip_final_snapshot = true
}
```

6. Creating RDS db Instance using terraform script.

I. Sudo mkdir RDS

II. cd RDS and create file sudo nano RDS.tf

The screenshot shows a terminal window with the prompt 'ubuntu@ip-172-31-84-44: ~/rds'. The user has opened a file named 'rds.tf' using 'GNU nano 4.8'. The file contains a Terraform configuration for an AWS RDS instance. The configuration includes provider settings for AWS and a resource definition for 'aws_db_instance'.

```
provider "aws" {
  region = "us-east-2"
  access_key = "AKIAI...VVKDQ"
  secret_key = "0x9lrC...nMZKIvBdnE"
}

resource "aws_db_instance" "default" {
  allocated_storage = 20
  db_name           = "mydb"
  engine            = "mysql"
  engine_version    = "5.7"
  instance_class     = "db.t2.micro"
  username          = "admin"
  password          = "templ2345"
  parameter_group_name = "default.mysql5.7"
  skip_final_snapshot = true
}
```

[Read 18 lines]

Get Help Write Out Where Is Cut Text Justify Cur Pos
Exit Read File Replace Paste Text To Spell Go To Line

III. Sudo terraform init.

```
ubuntu@ip-172-31-84-44: ~/rds
ubuntu@ip-172-31-84-44:~/s3$ cd ..
ubuntu@ip-172-31-84-44:~$ sudo mkdir rds
ubuntu@ip-172-31-84-44:~$ cd rds
ubuntu@ip-172-31-84-44:~/rds$ sudo nano rds.tf
ubuntu@ip-172-31-84-44:~/rds$ sudo nano rds.tf
ubuntu@ip-172-31-84-44:~/rds$ sudo terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.40.0...
```

IV. Sudo terraform apply.

```
ubuntu@ip-172-31-84-44: ~/rds
ubuntu@ip-172-31-84-44:~/rds$ sudo 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:
```

v. Check the Output db instance.

