

Inlämningsuppgift
Basic Cloud Computing
Jensen CAW21G

Namn: Sheeba

Email:rl.sheebanair@gmail.com

Betyg: _____

Content

I. Local Environment Setup

1. Install Virtual box & Linux -Ubuntu
2. Java installation
3. Visual Studio Installation & Terraform
4. MySQL installation
5. Spring boot installation
6. Terraform Installation in Ubuntu

II. Remote environment setup-AWS

1. Create VPC
2. Create Internet Gateway
3. Create custom route table
4. Create subnet
5. Create subnet with route table
6. Create Security Group
7. Create network interface
8. Assign elastic IP to the network interface
9. Create Ubuntu server and install Java, Spring boot, MySQL.

Virtual machine in Linux

1. Open the web browser and type the following URL, then click Enter

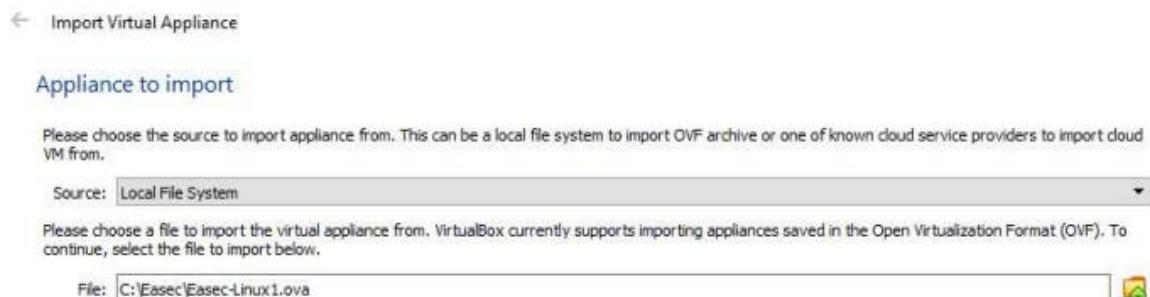
<https://stordevsumj.blob.core.windows.net/easec/easec-linux1.ova>

Create a directory `c:/easec` in order to store the virtual machine

2. In the computer, start virtual box which has been installed before. In that click on import

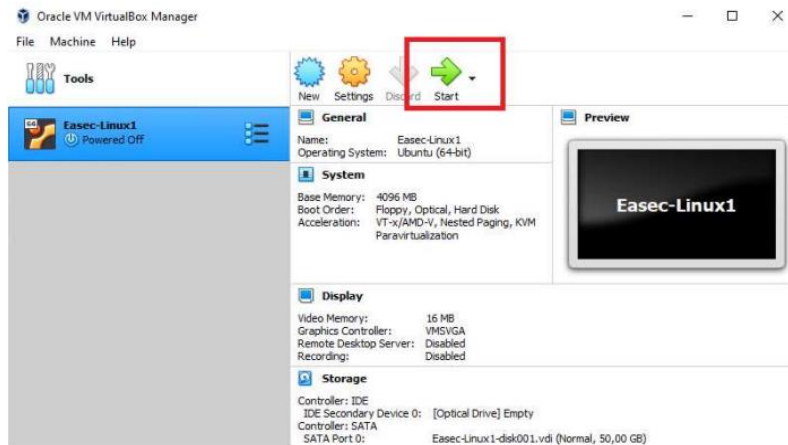


3. Then click on folder icon and browse to `easec-linux1.ova` from the directory and click next



4. In the Appliance to import window click on import button without making any changes,

Once it gets imported click on the start button which is displaying in the virtual box



5. After this log in easec with password.

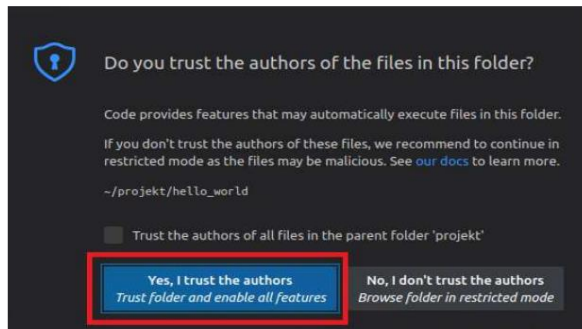
Install Java in the Ubuntu terminal

1. Open virtual machine and login with password. Then click Alt+Ctrl+T to open the Ubuntu terminal. Type **sudo apt-get update** command in the terminal.
2. Type the password for sudo.
3. Write the following command to install java in linux
4. `sudo apt-get install openjdk-11-jdk`
5. Type 'y' in order to confirm installation. Do you want to continue? and click enter.
6. Write following command to confirm the installation
7. `Sudo java -version`

```
easec@easec-linux1: ~
easec@easec-linux1:~$ sudo java -version
openjdk version "11.0.13" 2021-10-19
OpenJDK Runtime Environment (build 11.0.13+8-Ubuntu-0ubuntu1.20.04)
OpenJDK 64-Bit Server VM (build 11.0.13+8-Ubuntu-0ubuntu1.20.04, mixed mode, sharing)
easec@easec-linux1:~$
```

Visual studio code

1. open the webbrowser and type the following url and click on enter
<https://code.visualstudio.com/docs/languages/java>
2. click on Install extension pack for java then click on choose Application to open the Visual studio code link. click on open link.
3. Visual studio code open with Extension pack for java, click on install .
4. Click on terminal and type the command , click enter
Code .



5. Click on yes, I trust the authors. Visual Studio Code opens where we can select file menu, New file. Select the link and choose java.

Install Mysql 8.0 in linux

1. Open virtual machine and login with password. Then click Alt+Ctrl+T to open the Ubuntu terminal. Type **sudo apt-get update** command in the terminal.
2. Type the password for sudo.
3. Type the following command
4. `sudo apt-get install mysql-server`
5. write 'y' in order to install mysql and click enter.
6. Type the following command to configure Mysql 8.0 and click enter
7. `sudo mysql_secure_installation`
8. write 'y' in order to continue with configuration of mysql and type 0.
9. Enter the password and re-enter the password and write 'y' to continue with password
10. Again write 'y' to remove anonymous user and write 'y' for all other questions.
11. Type **sudo mysql** to verify the sql prompt

```
easec@easec-linux1:~$ sudo mysql
[sudo] password for easec:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.28-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> █
```

Install Spring Boot in linux

1. Open virtual machine and login with password. Then click Alt+Ctrl+T to open the Ubuntu terminal. Type **sudo apt-get update** command in the terminal.
2. Type the password for sudo.
3. Type the following command and click enter
4. wget <https://repo.spring.io/release/org/springframework/boot/spring-boot-cli/2.3.3.RELEASE/spring-boot-cli-2.3.3.RELEASE-bin.tar.gz>
5. Type the following command
6. tar -xzf spring-boot-cli-2.3.3.RELEASE-bin.tar.gz
7. Write the command in the terminal and click enter
8. sudo mv spring-boot-cli-2.3.3.RELEASE /opt
9. Write the command and click enter
10. sudo ln -s /opt/spring-2.3.3.RELEASE/bin/spring /usr/bin/spring
11. Then write the command ,press enter
12. sudo ln -s /opt/spring-boot-cli-2.3.3.RELEASE/shell-completion/bash/spring /etc/bash_completion.d/spring

```
easec@easec-linux1:~$ wget https://repo.spring.io/release/org/springframework/boot/spring-boot-cli/2.3.3.RELEASE/spring-boot-cli-2.3.3.RELEASE-bin.tar.gz
--2022-04-20 15:42:24-- https://repo.spring.io/release/org/springframework/boot/spring-boot-cli/2.3.3.RELEASE/spring-boot-cli-2.3.3.RELEASE-bin.tar.gz
Resolving repo.spring.io (repo.spring.io)... 35.243.130.159
Connecting to repo.spring.io (repo.spring.io)|35.243.130.159|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11607228 (11M) [application/x-gzip]
Saving to: 'spring-boot-cli-2.3.3.RELEASE-bin.tar.gz'

spring-boot-cli-2.3.3.RELEASE-bin.t 100%[=====] 11,07M 4,97MB/s in 2,2s
2022-04-20 15:42:27 (4,97 MB/s) - 'spring-boot-cli-2.3.3.RELEASE-bin.tar.gz' saved [11607228/11607228]
```

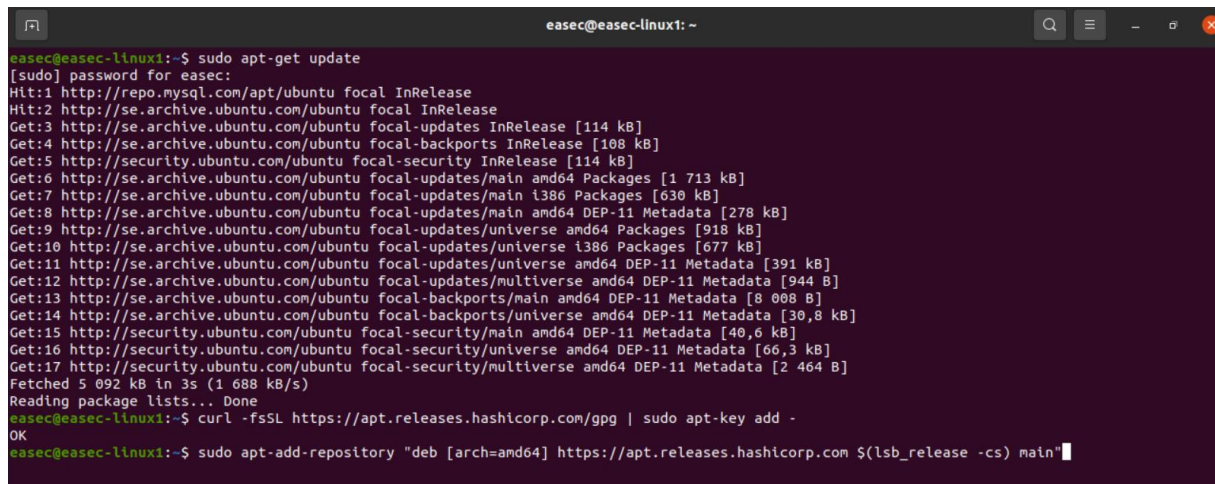
Install Terraform in Ubuntu

1. Open virtual machine and login with password. Then click Alt+Ctrl+T to open the Ubuntu terminal. Type **sudo apt-get update** command in the terminal.

```
easec@easec-linux1:~$ sudo apt-get update
[sudo] password for easec:
Hit:1 http://repo.mysql.com/apt/ubuntu focal InRelease
Hit:2 http://se.archive.ubuntu.com/ubuntu focal InRelease
Get:3 http://se.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 http://se.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:6 http://se.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1 713 kB]
Get:7 http://se.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [630 kB]
Get:8 http://se.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metadata [278 kB]
Get:9 http://se.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [918 kB]
Get:10 http://se.archive.ubuntu.com/ubuntu focal-updates/universe i386 Packages [677 kB]
Get:11 http://se.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11 Metadata [391 kB]
Get:12 http://se.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 DEP-11 Metadata [944 B]
Get:13 http://se.archive.ubuntu.com/ubuntu focal-backports/main amd64 DEP-11 Metadata [8 008 B]
Get:14 http://se.archive.ubuntu.com/ubuntu focal-backports/universe amd64 DEP-11 Metadata [30,8 kB]
Get:15 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [40,6 kB]
Get:16 http://security.ubuntu.com/ubuntu focal-security/universe amd64 DEP-11 Metadata [66,3 kB]
Get:17 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 DEP-11 Metadata [2 464 B]
Fetched 5 092 kB in 3s (1 688 kB/s)
Reading package lists... Done
easec@easec-linux1:~$ curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
```

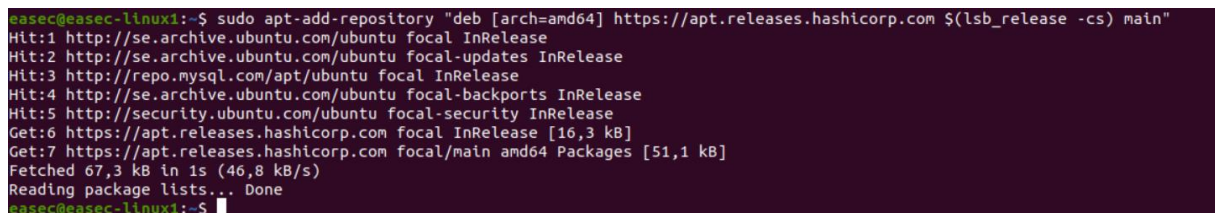

2. Then type the following command, which displays ok.

```
curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
```

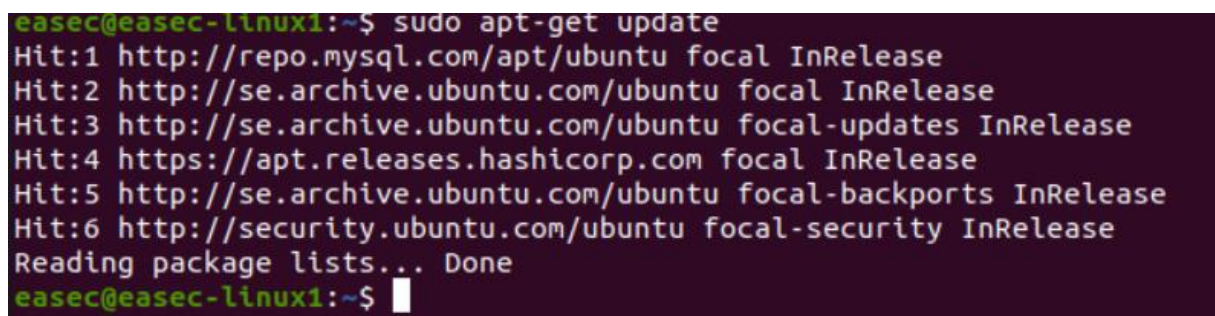
A terminal window titled 'easec@easec-linux1: ~' showing the output of 'sudo apt-get update'. It lists various repositories and their package lists, including focal, focal-updates, focal-backports, focal-security, and focal-updates/main, amd64 Packages. The output shows the progress of downloading package lists from these sources. After the update, the user runs 'curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -', which returns 'OK'. Finally, the user runs 'sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com \$(lsb_release -cs) main"', which also returns 'OK'.

3. Then write the following command in order to add reference to terraform archive

```
sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb_release -cs) main"
```

A terminal window titled 'easec@easec-linux1: ~' showing the output of 'sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com \$(lsb_release -cs) main"'. It lists various repositories and their package lists, including focal, focal-updates, focal-backports, focal-security, and focal-updates/main, amd64 Packages. The output shows the progress of downloading package lists from these sources. After the update, the user runs 'curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -', which returns 'OK'. Finally, the user runs 'sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com \$(lsb_release -cs) main"', which also returns 'OK'.

4. Then again write the sudo apt-get update command in the terminal to update the reference

A terminal window titled 'easec@easec-linux1: ~' showing the output of 'sudo apt-get update'. It lists various repositories and their package lists, including focal, focal-updates, focal-backports, focal-security, and focal-updates/main, amd64 Packages. The output shows the progress of downloading package lists from these sources. After the update, the user runs 'curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -', which returns 'OK'. Finally, the user runs 'sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com \$(lsb_release -cs) main"', which also returns 'OK'.

5. To Install Terraform use the command **sudo apt-get install terraform**

```
easec@easec-linux1:~$ sudo apt-get install terraform
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  terraform
0 upgraded, 1 newly installed, 0 to remove and 62 not upgraded.
Need to get 18,8 MB of archives.
After this operation, 63,3 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com focal/main amd64 terraform amd64 1.1.8 [18,8 MB]
Fetched 18,8 MB in 1s (13,0 MB/s)
Selecting previously unselected package terraform.
(Reading database ... 182547 files and directories currently installed.)
Preparing to unpack .../terraform_1.1.8_amd64.deb ...
Unpacking terraform (1.1.8) ...
Setting up terraform (1.1.8) ...
easec@easec-linux1:~$
```

6.To verify the installation of Terraform ,write the command and click enter

terraform -help

```
easec@easec-linux1:~$ terraform -help
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
  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      Experimental support for module integration testing
  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.
```

7.Write the following command in order to install Unzip package

Sudo apt-get install unzip

```
easec@easec-linux1:~$ sudo apt-get install unzip
Reading package lists... Done
Building dependency tree
Reading state information... Done
unzip is already the newest version (6.0-25ubuntu1).
unzip set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 62 not upgraded.
easec@easec-linux1:~$
```

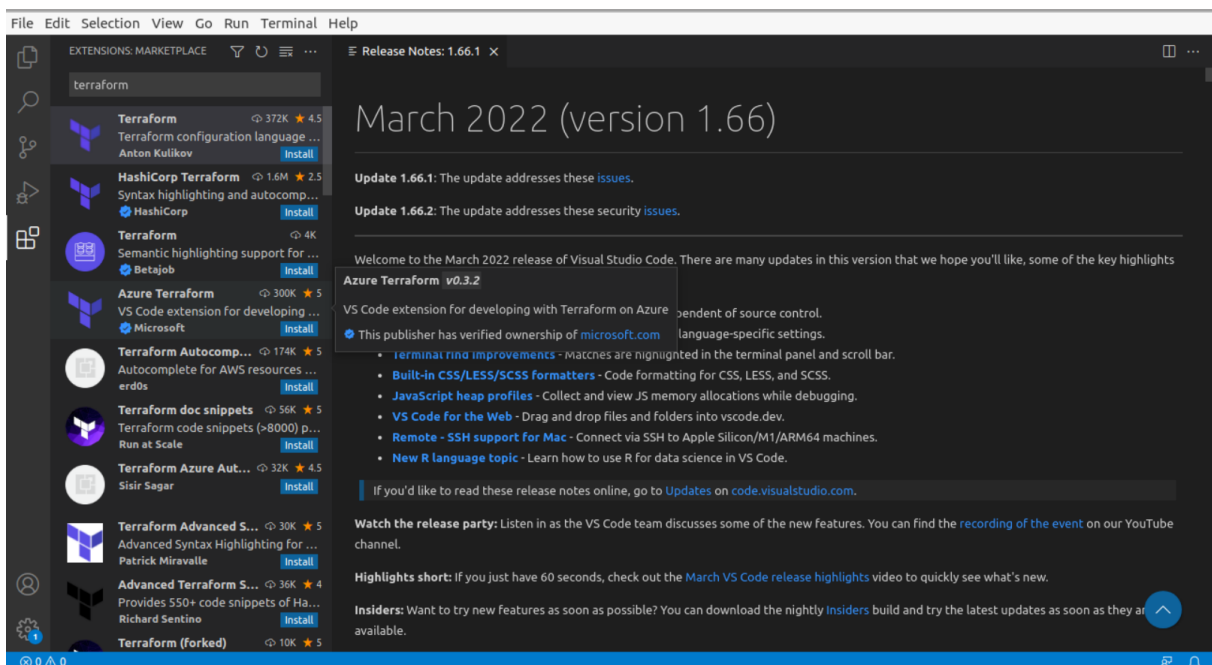
8.The command used to check the version of Terraform

terraform -v

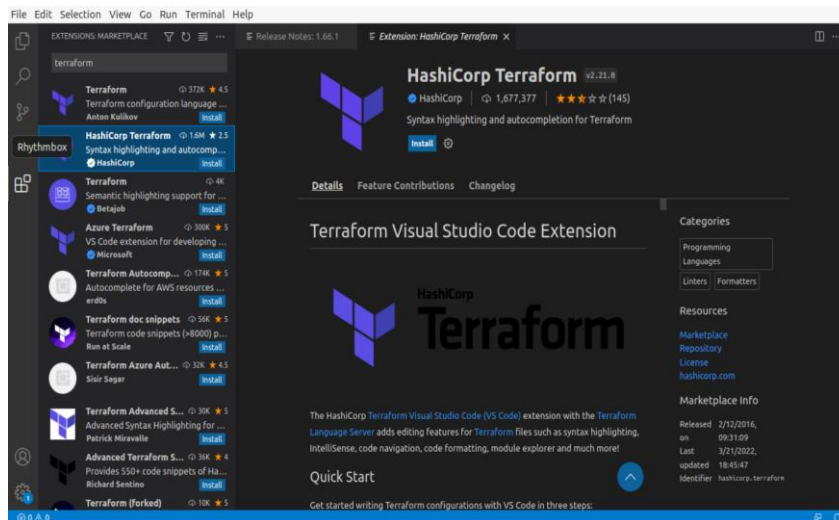
```
easec@easec-linux1:~$ terraform -v
Terraform v1.1.8
on linux_amd64
easec@easec-linux1:~$
```

Using IaC Framework(Terraform in visual studio code)

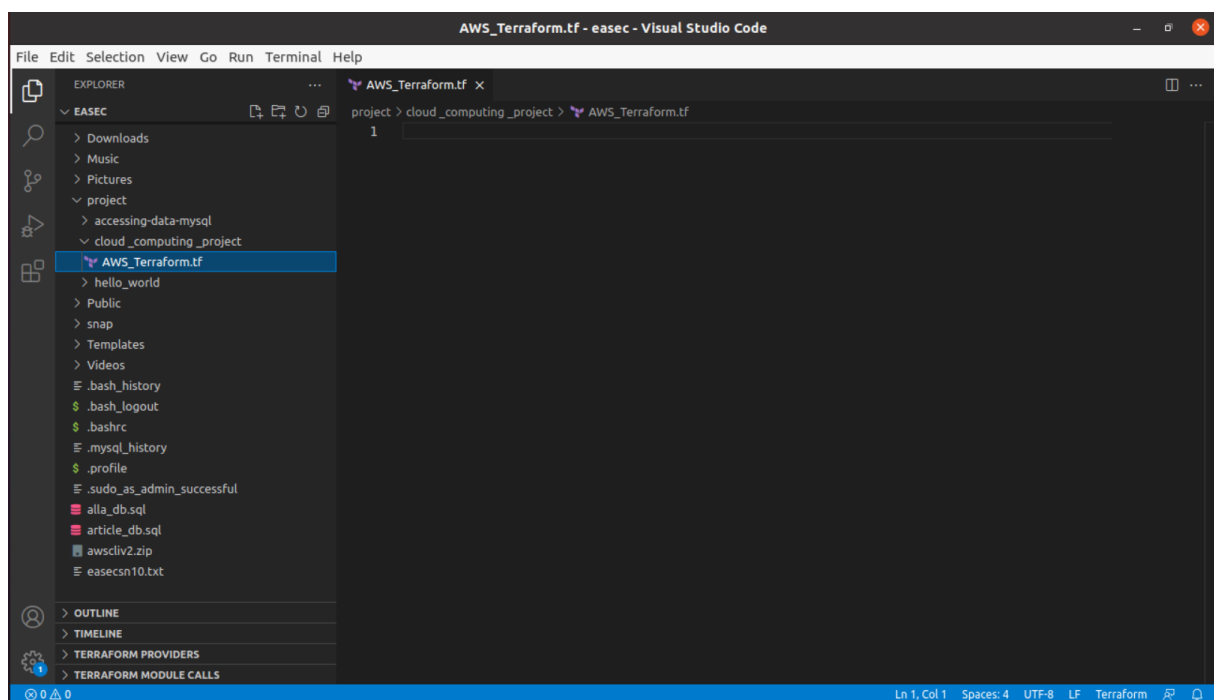
1.In Visual studio code click on extensions and type terraform ,then select Hashicorp Terraform



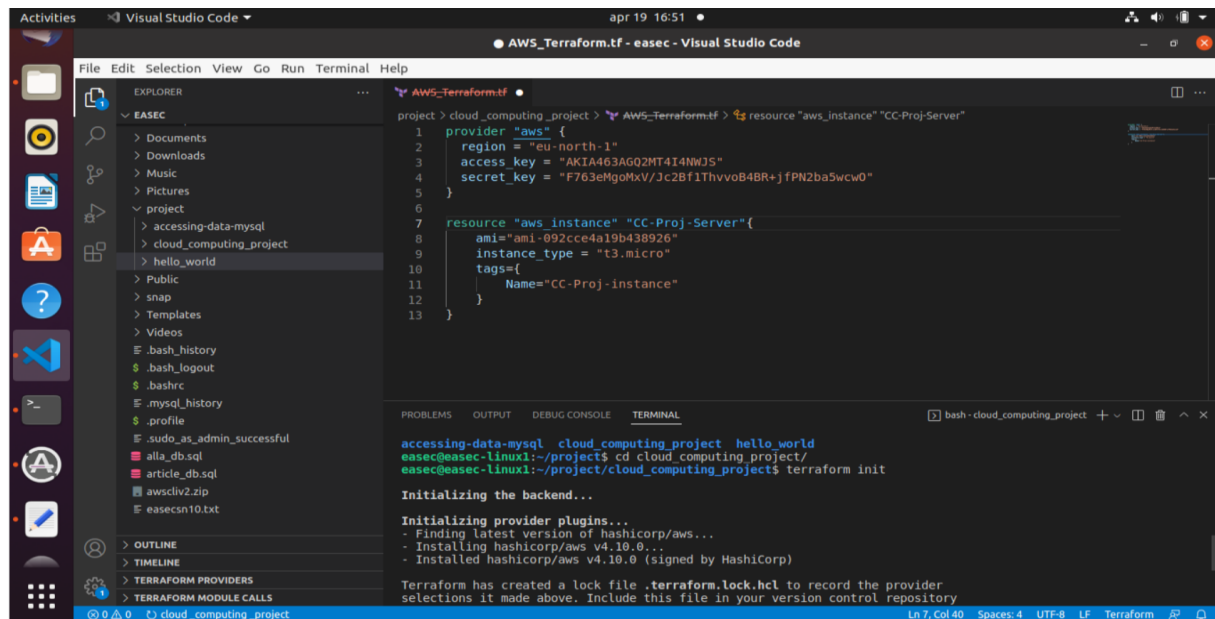
2.Click on Install button and then close extension:hashicorp terraform .



3. Then in virtual machine click on files and create new folder. I have created a folder inside `home/easec/project/cloud_computing_project`.



4. In Visual studio code Click on file –new file then save the file as `AWS_Terraform.tf` inside the folder `cloud_computing_project`. When you click on explorer you can able to see your folder with given filename.



5. In the terraform file type the following code

```
provider "aws" {
  region = "eu-north-1"
  access key = " "
  secret key = " "
}
```

6. Write the aws instance resources with ami and instance type in the same terraform file

```
resource "aws_instance" "CC-Proj-Server"{
  ami="ami-092cce4a19b438926"
  instance_type = "t3.micro"
  tags={
    Name="CC-Proj-instance"
  }
}
```

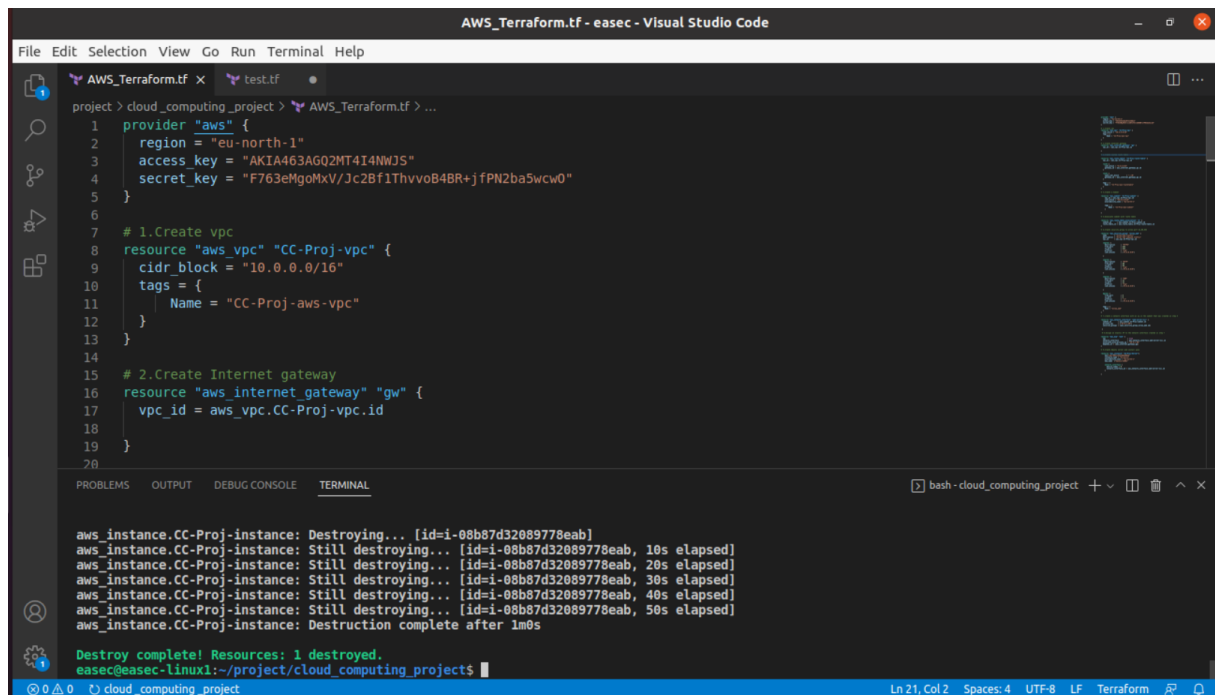
Remote Environment Setup-AWS

- Create VPC
- Create Internet Gateway
- Create custom route table
- Create subnet
- Create subnet with route table
- Create Security Group
- Create network interface
- Assign elastic IP to the network interface
- Create Ubuntu server and install Java, Spring boot, MySQL.

1. Created different resources in Terraform file

First created a resource vpc and then internet gateway. These resources can be created in any order.

We can refer these resources from AWS providers documentation



The screenshot shows a Visual Studio Code window titled "AWS_Terraform.tf - eassec - Visual Studio Code". The editor displays a Terraform configuration file with the following content:

```
1 provider "aws" {
2   region = "eu-north-1"
3   access_key = "AKIA463AGQ2MT4I4NMJS"
4   secret_key = "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"
5 }
6
7 # 1.Create vpc
8 resource "aws_vpc" "CC-Proj-vpc" {
9   cidr_block = "10.0.0.0/16"
10  tags = {
11    Name = "CC-Proj-aws-vpc"
12  }
13 }
14
15 # 2.Create Internet gateway
16 resource "aws_internet_gateway" "gw" {
17   vpc_id = aws_vpc.CC-Proj-vpc.id
18 }
19
20
```

The terminal at the bottom shows the output of the Terraform command, indicating that the resources were destroyed successfully:

```
aws_instance.CC-Proj-instance: Destroying... [id=i-08b87d32089778eab]
aws_instance.CC-Proj-instance: Still destroying... [id=i-08b87d32089778eab, 10s elapsed]
aws_instance.CC-Proj-instance: Still destroying... [id=i-08b87d32089778eab, 20s elapsed]
aws_instance.CC-Proj-instance: Still destroying... [id=i-08b87d32089778eab, 30s elapsed]
aws_instance.CC-Proj-instance: Still destroying... [id=i-08b87d32089778eab, 40s elapsed]
aws_instance.CC-Proj-instance: Still destroying... [id=i-08b87d32089778eab, 50s elapsed]
aws_instance.CC-Proj-instance: Destruction complete after 1m0s
Destroy complete! Resources: 1 destroyed.
eassec@eassec-linux1:~/project/cloud_computing_projects$
```

2. Then we created route table resource in this vpc_id is the name we created in vpc with .id should be added to it. Same way as gateway_id.

```
project > cloud_computing_project > AWS_Terraform.tf > ...
21 # 3.Create custom route table
22
23 resource "aws_route_table" "CC-Proj-route-table" {
24     vpc_id = aws_vpc.CC-Proj-vpc.id
25
26     route {
27         cidr_block = "0.0.0.0/0"
28         gateway_id = aws_internet_gateway.gw.id
29     }
30
31     route {
32         ipv6_cidr_block = ":::/0"
33         gateway_id = aws_internet_gateway.gw.id
34     }
35
36     tags = {
37         Name = "CC-Proj-aws-routetable"
38     }
39 }
```

3.In this step we created subnet and subnet route table for our vpc with availability zone which one we selected in our console

```
project > cloud_computing_project > AWS_Terraform.tf > ...
40
41 # 4.Create a Subnet
42
43 resource "aws_subnet" "CC-Proj-subnet" {
44     vpc_id = aws_vpc.CC-Proj-vpc.id
45     cidr_block = "10.0.0.0/16"
46     availability_zone = "eu-north-1"
47
48     tags = {
49         Name = "CC-Proj-aws-subnet"
50     }
51 }
52
53
54 # 5.Associate subnet with route table
55
56 resource "aws_route_table_association" "a" {
57     subnet_id = aws_subnet.CC-Proj-subnet.id
58     route_table_id = aws_route_table.CC-Proj-route-table.id
59 }
```

4.Here we created a security group for different ports such as HTTPS,HTTP and SSH with port number.


```

AWS_Terraform.tf x test.tf
project > cloud_computing_project > AWS_Terraform.tf > ...
61 # 6.Create security group to allow port 22,80,443
62
63 resource "aws_security_group" "allow_web" {
64     name           = "allow_web_traffic"
65     description    = "Allow web inbound traffic"
66     vpc_id         = aws_vpc.CC-Proj-vpc.id
67
68     ingress {
69         description    = "HTTPS"
70         from_port      = 443
71         to_port        = 443
72         protocol        = "tcp"
73         cidr_blocks     = ["0.0.0.0/0"]
74     }
75
76     ingress {
77         description    = "HTTP"
78         from_port      = 80
79         to_port        = 80
80     }

```

5.Create network interface and assign elastic IP to the network interface with subnet id.

```

AWS_Terraform.tf test.tf
project > cloud_computing_project > AWS_Terraform.tf > ...
105 }
106
107 # 7.create a network interface with an ip in the subnet that was created in step 4
108
109 resource "aws_network_interface" "web-server-nic" {
110     subnet_id      = aws_subnet.CC-Proj-subnet.id
111     private_ips    = ["10.0.1.50"]
112     security_groups = [aws_security_group.allow_web.id]
113 }
114
115
116 # 8.Assign an elastic IP to the network interface created in step 7
117
118 resource "aws_eip" "one" {
119     vpc              = true
120     network_interface = aws_network_interface.web-server-nic.id
121     associate_with_private_ip = "10.0.1.50"
122     depends_on       = [aws_internet_gateway.gw]
123 }
124

```

6.In this final step we created Ubuntu server and installed Java, Springboot and Mysql with different commands and finally run the terraform command

```

125 # 9.Create Ubuntu server and install java,springboot,mysql
126
127 resource "aws_instance" "CC-Proj-Server"{
128     ami="ami-092cce4a19b438926"
129     instance_type = "t3.micro"
130     availability_zone = "eu-north-1"
131     key_name = "eassec srl0401"
132
133     network_interface {
134         device_index = 0
135         network_interface_id = aws_network_interface.web-server-nic.id
136     }
137
138     user_data = <<-EOF
139     #!/bin/bash
140     sudo apt update -y
141     sudo apt-get install openjdk-11-jdk -y
142
143     wget https://repo.spring.io/release/org/springframework/boot/spring-boot-cli/2.3.3.RELEASE/spring-boot-cli-2.3.3.RELEASE
144     tar -xzf spring-boot-cli-2.3.3.RELEASE-bin.tar.gz

```

```

wget https://repo.spring.io/release/org/springframework/boot/spring-boot-cli/2.3.3.RELEASE/spring-boot-cli-2.3.3.RELEASE
tar -xzf spring-boot-cli-2.3.3.RELEASE-bin.tar.gz
sudo mv spring-boot-cli-2.3.3.RELEASE /opt/
sudo ln -s /opt/spring-2.3.3.RELEASE/bin/spring /usr/bin/spring
sudo ln -s /opt/spring-boot-cli-2.3.3.RELEASE/shell-completion/bash/spring
/etc/bash_completion.d/spring

sudo apt update -y
sudo apt-get install mysql-server
sudo dpkg --configure -a
sudo mysql_secure_installation -y
EOF

tags={
    Name="CC-Proj-instance"
}
}

```

The screenshot shows the Visual Studio Code interface. The top editor pane displays a Terraform configuration file named `test.tf` with the following content:

```

16 sudo apt update -y
17 sudo apt-get install openjdk-11-jdk -y
18
19 wget https://repo.spring.io/release/org/springframework/boot/spring-boot-cli/2.3.3.RELEASE/spring-boot-cli-2.3.3.RELEASE
20 tar -xzf spring-boot-cli-2.3.3.RELEASE-bin.tar.gz
21 sudo mv spring-boot-cli-2.3.3.RELEASE /opt/
22 sudo ln -s /opt/spring-2.3.3.RELEASE/bin/spring /usr/bin/spring
23 sudo ln -s /opt/spring-boot-cli-2.3.3.RELEASE/shell-completion/bash/spring
24 /etc/bash_completion.d/spring
25
26 sudo apt update -y
27 sudo apt-get install mysql-server
28 sudo dpkg --configure -a
29 sudo mysql_secure_installation -y
30 EOF
31

```

The bottom pane shows the **TERMINAL** output, which includes the Terraform plan and execution results:

```

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

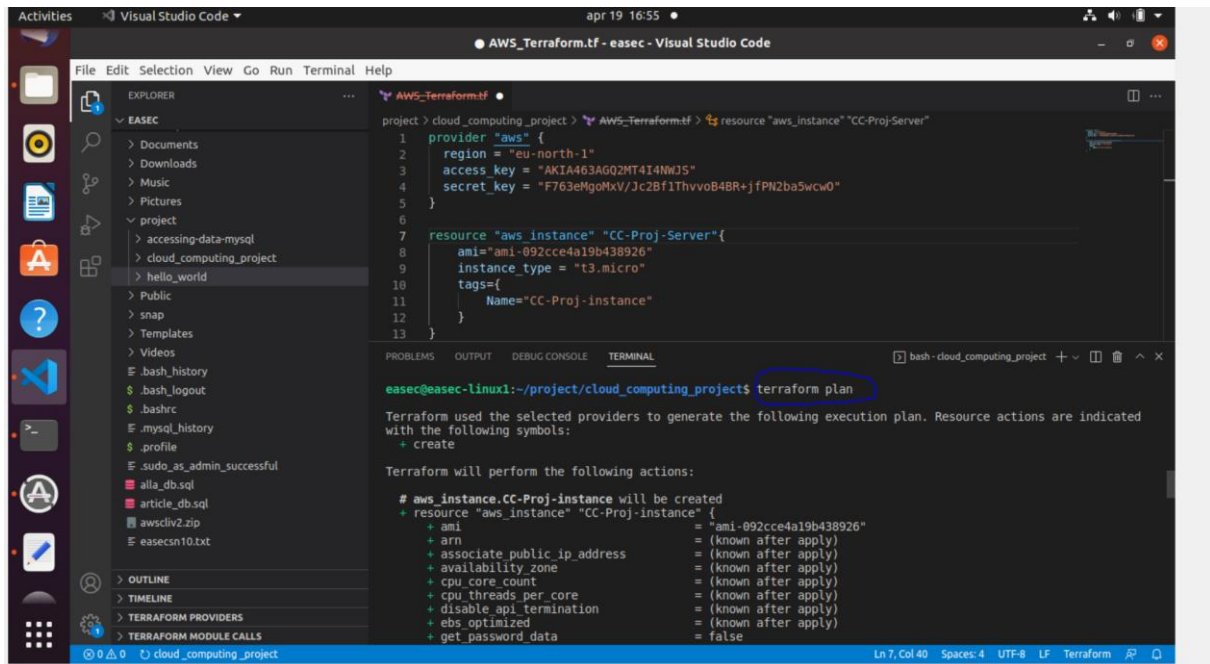
aws_instance.CC-Proj-Server: Creating...
aws_instance.CC-Proj-Server: Still creating... [10s elapsed]
aws_instance.CC-Proj-Server: Creation complete after 13s [id=i-07100883ceb9b45e7]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
eassec@eassec-linux1:~/project$

```

7. Click on Terminal menu, select New Terminal which will get displayed at the bottom of visual studio code. Then type command `cd project`, click enter.

Then type `cd cloud_computing_project` in order to move to our directory where we stored our Terraform file, `AWS_Terraform.tf`



The screenshot shows the Visual Studio Code interface with the `AWS_Terraform.tf` file open. The file contains the following Terraform configuration:

```
1 provider "aws" {
2   region = "eu-north-1"
3   access_key = "AKIA463AGQ2MT4I4NMJ5"
4   secret_key = "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"
5 }
6
7 resource "aws_instance" "CC-Proj-Server"{
8   ami="ami-092cce4a19b438926"
9   instance_type = "t3.micro"
10  tags={
11    Name="CC-Proj-instance"
12  }
13 }
```

The terminal shows the output of the `terraform plan` command:

```
easec@easec-linux1:~/project/cloud_computing_projects$ terraform plan
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

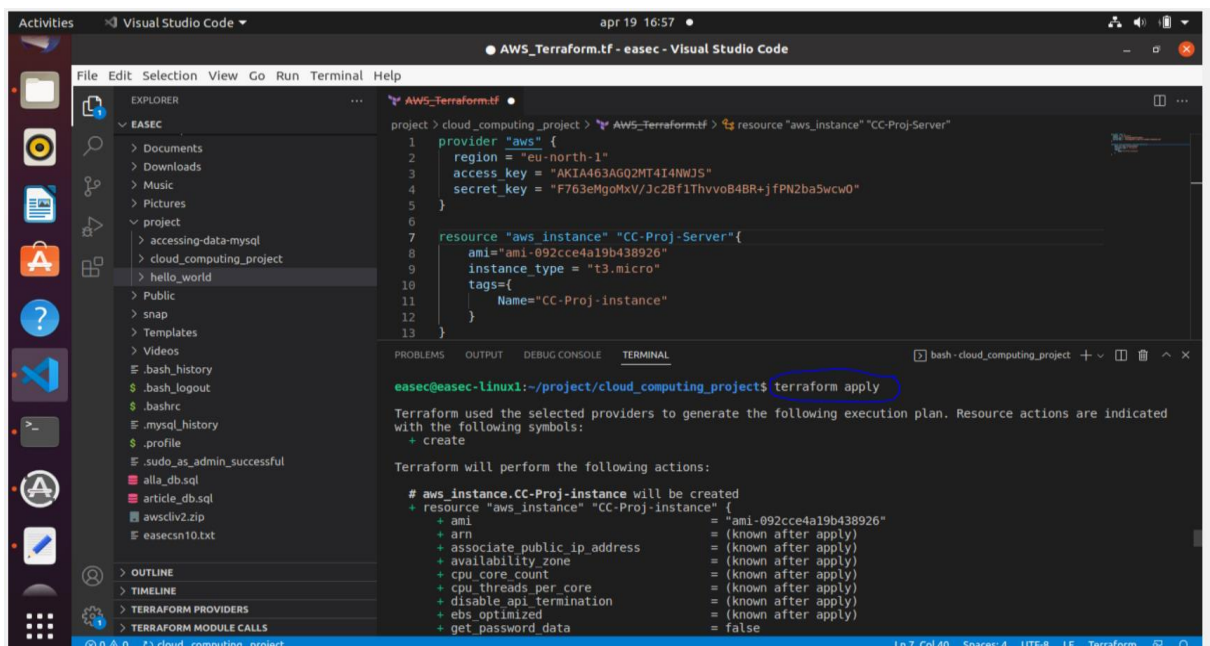
Terraform will perform the following actions:

# aws_instance.CC-Proj-instance will be created
+ resource "aws_instance" "CC-Proj-instance" {
+   ami               = "ami-092cce4a19b438926"
+   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_termination = (known after apply)
+   ebs_optimized     = (known after apply)
+   get_password_data  = false
}
```

8. Then write the following command and click enter in order to initialize terraform

`terraform init`

9. Type the command **terraform apply** then click enter. Enter your value: yes type yes to perform actions.



The screenshot shows the Visual Studio Code interface with the `AWS_Terraform.tf` file open. The file contains the same Terraform configuration as in the previous screenshot.

The terminal shows the output of the `terraform apply` command:

```
easec@easec-linux1:~/project/cloud_computing_projects$ 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.CC-Proj-instance will be created
+ resource "aws_instance" "CC-Proj-instance" {
+   ami               = "ami-092cce4a19b438926"
+   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_termination = (known after apply)
+   ebs_optimized     = (known after apply)
+   get_password_data  = false
}
```

10. Open the browser and type the following URL, then login with IAM user ,AccountId and Password.

<https://console.aws.amazon.com>

11.Type EC2 in the search box and click on instances(running) and it shows the running instances in AWS

