

**SCENARIO:-**

Mr. Bobby is an IT Administrator of Alchemist group Pvt. Ltd. His organization is passionate about adopting IaaS using public cloud service providers. Their majority of clients are e-commerce and OTP service providers. Initially, they want to set up two virtual windows servers using Amazon EC2 which can be resizable and provide compute capacity along with a web-scale cloud computing solution.

Bobby is planning to create IaaS as below for E-Commerce clients. You are required to provide the solution to Bobby with proper step by step demonstration. Consider the following attached scenario and perform the following tasks using AWS EC2 Service:

- Launch a web server with termination protection enabled
- Monitor Your EC2 instance
- Modify the security group that your web server is using to allow HTTP access
- Resize your Amazon EC2 instance to scale
- Explore EC2 limits
- Test termination protection
- Terminate your EC2 instance

**AIM:-** To automate above Infrastructure using Infrastructure As A Code - Terraform.

**Steps:-**

**1. Configuring terraform code.**

```
- CODE:-      (main.tf)
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
  required_version = ">= 0.13.5"
}

# Configure the AWS Provider
provider "aws" {
  region      = "us-east-1"
  access_key  = "ACCESS_KEY"
  secret_key  = "SECRET_KEY"
}

# Create a Instance
resource "aws_instance" "vm" {
```

**ICT GANPAT UNIVERSITY**  
**COMPUTER SCIENCE DEPARTMENT**  
**Cloud Computing Essentials (2CSE710)**

---

```
ami                = "ami-0cfff7528ff583bf9a"
instance_type      = "t2.micro"
key_name           = "MyKey"
security_groups    = ["${aws_security_group.allow_http.name}"]
disable_api_termination = true
monitoring         = true

tags = {
    Name = "Amazon Linux 2"
}
}

resource "aws_security_group" "allow_http" {
    name = "allow_http"

    ingress {
        from_port    = 22
        to_port      = 22
        protocol      = "tcp"
        cidr_blocks   = ["0.0.0.0/0"]
    }

    ingress {
        from_port    = 80
        to_port      = 80
        protocol      = "tcp"
        cidr_blocks   = ["0.0.0.0/0"]
    }

    ingress {
        from_port    = 0
        to_port      = 0
        protocol      = "-1"
        cidr_blocks   = ["0.0.0.0/0"]
        ipv6_cidr_blocks = [ "::/0" ]
    }

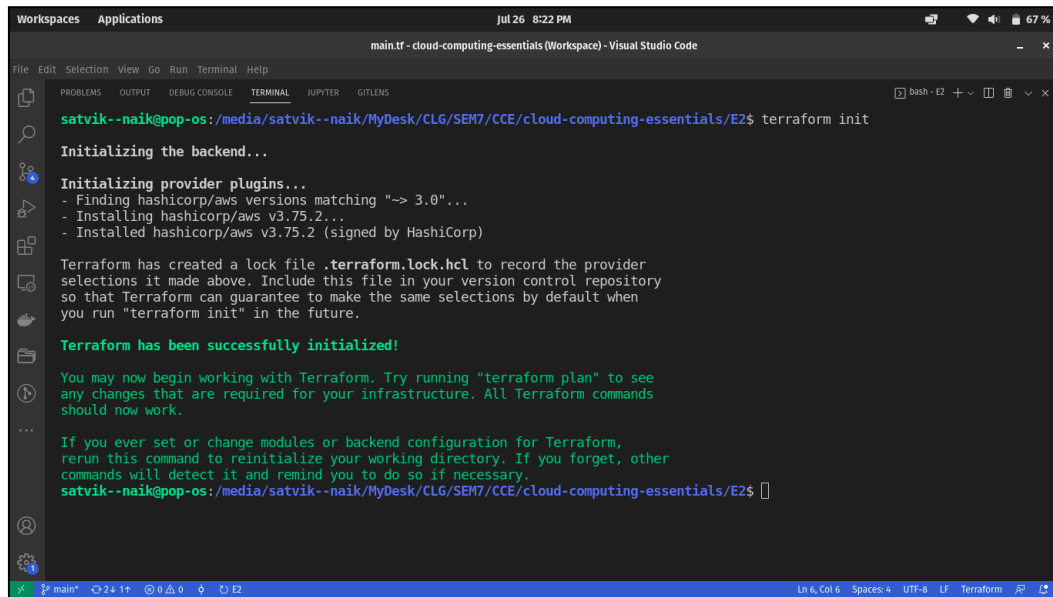
    egress {
        from_port    = 0
        to_port      = 0
        protocol      = "-1"
        cidr_blocks   = ["0.0.0.0/0"]
        ipv6_cidr_blocks = [ "::/0" ]
    }
}
```

- ***disable\_api\_termination*** If true, enables EC2 Instance Termination Protection.
- ***monitoring*** If true, the launched EC2 instance will have detailed monitoring enabled.

## 2. Create an instance via terraform.

Terraform has 5 main commands, which are as follows:

- **terraform init**
  - This prepares your working directory for other commands.



```
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E2$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 3.0"...
- Installing hashicorp/aws v3.75.2...
- Installed hashicorp/aws v3.75.2 (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.

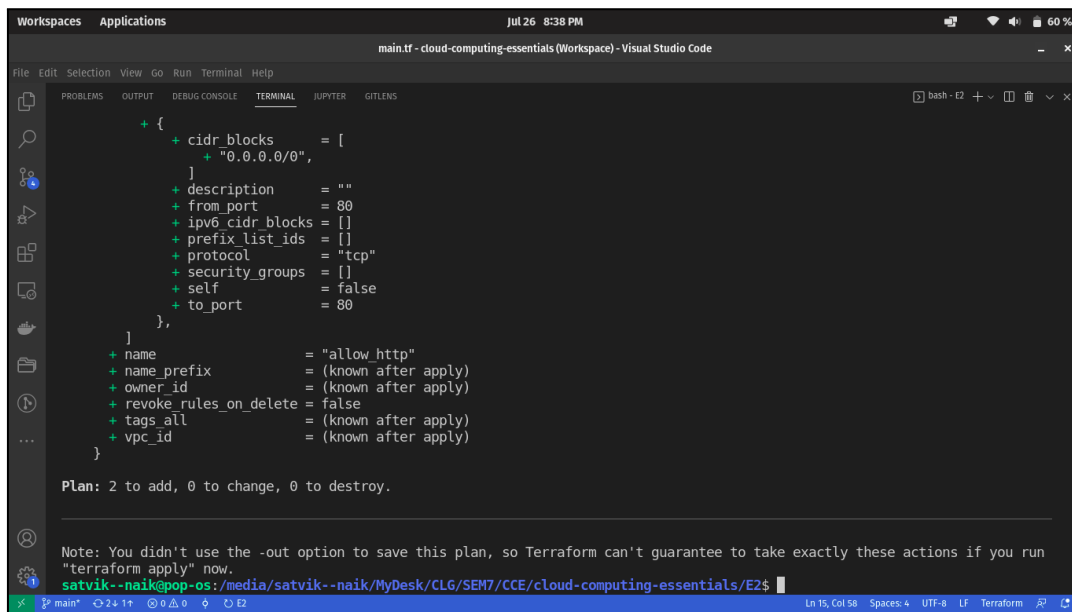
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E2$
```

- **terraform validate**
  - Checks whether the configuration is valid or not.

```
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E2$ terraform validate
Success! The configuration is valid.
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E2$
```

- **terraform plan**
  - It shows changes required by the current configuration.

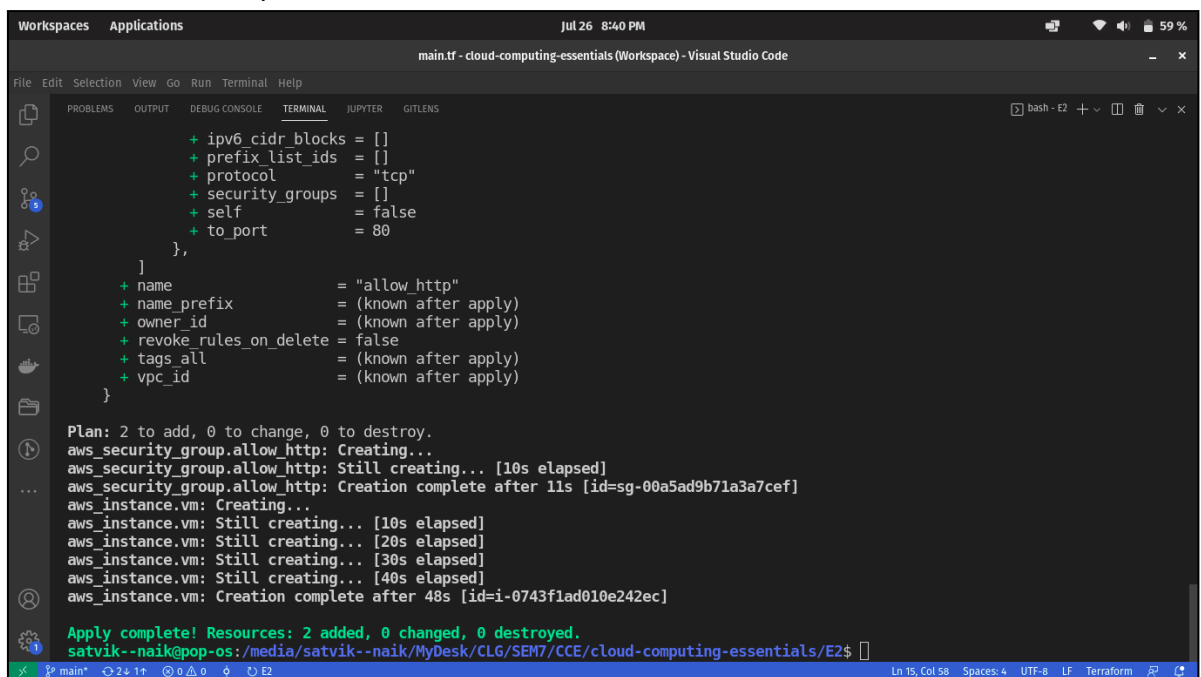
**ICT GANPAT UNIVERSITY**  
**COMPUTER SCIENCE DEPARTMENT**  
**Cloud Computing Essentials (2CSE710)**



The screenshot shows the Visual Studio Code interface with a workspace named 'main.tf - cloud-computing-essentials'. The terminal window displays the output of a Terraform plan command. It shows a plan to add two resources: 'aws\_security\_group.allow\_http' and 'aws\_instance.vm'. The plan indicates that 2 resources will be added, 0 changed, and 0 destroyed. A note at the bottom states: '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.'

```
+ {  
  + cidr_blocks = [  
    + "0.0.0.0/0",  
  ]  
  + description = ""  
  + from_port   = 80  
  + ipv6_cidr_blocks = []  
  + prefix_list_ids = []  
  + protocol     = "tcp"  
  + security_groups = []  
  + self         = false  
  + to_port      = 80  
},  
+ name           = "allow http"  
+ name_prefix    = (known after apply)  
+ owner_id       = (known after apply)  
+ revoke_rules_on_delete = false  
+ tags_all       = (known after apply)  
+ vpc_id         = (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.  
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E2$
```

- terraform apply
  - Creates or updates infrastructure.

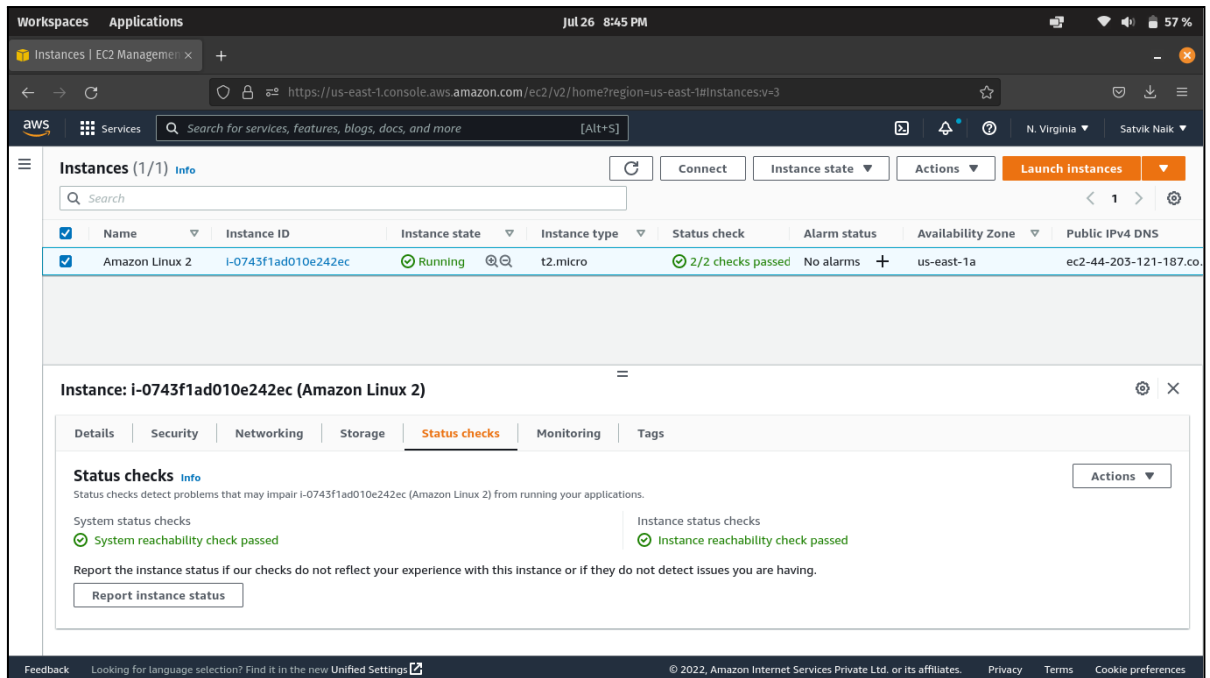


The screenshot shows the Visual Studio Code interface with the same workspace. The terminal window displays the output of a Terraform apply command. It shows the successful creation of the resources 'aws\_security\_group.allow\_http' and 'aws\_instance.vm'. The output includes status messages like 'Creating...', 'Still creating...', and 'Creation complete after' with timestamps. At the bottom, it states 'Apply complete! Resources: 2 added, 0 changed, 0 destroyed.'

```
+ ipv6_cidr_blocks = []  
+ prefix_list_ids = []  
+ protocol        = "tcp"  
+ security_groups = []  
+ self           = false  
+ to_port         = 80  
},  
+ name           = "allow_http"  
+ name_prefix    = (known after apply)  
+ owner_id       = (known after apply)  
+ revoke_rules_on_delete = false  
+ tags_all       = (known after apply)  
+ vpc_id         = (known after apply)  
}  
  
Plan: 2 to add, 0 to change, 0 to destroy.  
aws_security_group.allow_http: Creating...  
aws_security_group.allow_http: Still creating... [10s elapsed]  
aws_security_group.allow_http: Creation complete after 11s [id=sg-00a5ad9b71a3a7cef]  
aws_instance.vm: Creating...  
aws_instance.vm: Still creating... [10s elapsed]  
aws_instance.vm: Still creating... [20s elapsed]  
aws_instance.vm: Still creating... [30s elapsed]  
aws_instance.vm: Still creating... [40s elapsed]  
aws_instance.vm: Creation complete after 48s [id=i-0743f1ad010e242ec]  
  
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.  
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E2$
```

**ICT GANPAT UNIVERSITY**  
**COMPUTER SCIENCE DEPARTMENT**  
**Cloud Computing Essentials (2CSE710)**

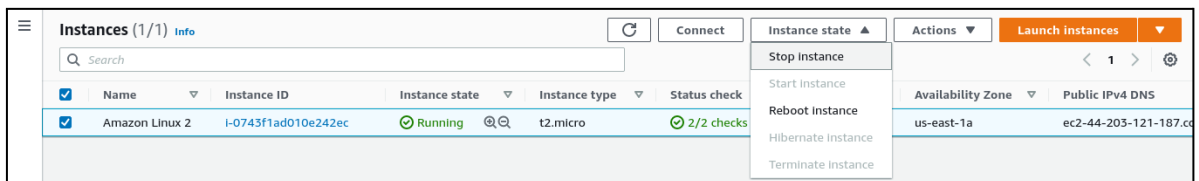
- After applying the code, the instance has been successfully **created** & is in a **Running** state.



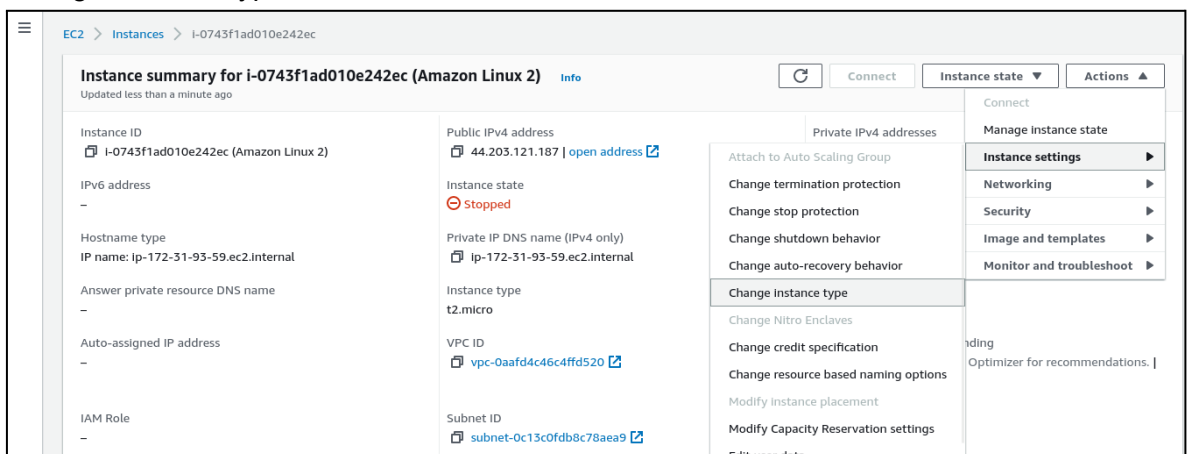
### 3. Resizing Amazon EC2 instance to scale.

- In order to resize amazon EC2 instance to scale, you must **stop the existing instance** and then you should **change the instance type**.

- Stop instance



- Change instance type



**ICT GANPAT UNIVERSITY**  
**COMPUTER SCIENCE DEPARTMENT**  
**Cloud Computing Essentials (2CSE710)**

- By default t2.micro is selected, for scaling other instance types could be selected and further applied. (i.e. t2.xlarge, t2.2xlarge etc...)

The screenshot shows the 'Change instance type' page in the AWS Management Console. The breadcrumb trail is 'EC2 > Instances > i-0743f1ad010e242ec > Change instance type'. The page title is 'Change instance type' with an 'Info' link. A note states: 'You can change the instance type only if the current instance type and the instance type that you want are compatible.' The 'Instance ID' is 'i-0743f1ad010e242ec (Amazon Linux 2)'. The 'Current instance type' is 't2.micro'. The 'Instance type' dropdown menu is set to 't2.micro'. There is a checkbox for 'EBS-optimized' which is currently unchecked, with a note below it: 'EBS-optimized is not supported for this instance type'. At the bottom right, there are 'Cancel' and 'Apply' buttons.

- Select the particular instance type and click on '**Apply**' & then start your instance.

#### 4. Connect to instance via EC2 instance Connect.

- Go to the instance and click on '**Connect**'.

The screenshot shows the 'Instance summary' page for instance 'i-0743f1ad010e242ec (Amazon Linux 2)'. The breadcrumb trail is 'EC2 > Instances > i-0743f1ad010e242ec'. The page title is 'Instance summary for i-0743f1ad010e242ec (Amazon Linux 2)' with an 'Info' link. A note says 'Updated less than a minute ago'. There are buttons for 'Refresh', 'Connect', 'Instance state', and 'Actions'. The instance details are as follows:

| Instance ID                          | Public IPv4 address                        | Private IPv4 addresses |
|--------------------------------------|--|------------------------|
| i-0743f1ad010e242ec (Amazon Linux 2) | 3.86.109.58   <a href="#">open address</a> | 172.31.93.59           |

Instance state: Running

Public IPv4 DNS: [ec2-3-86-109-58.compute-1.amazonaws.com | open address](#)

- Select EC2 instance connect.

The screenshot shows the 'Connect to instance' page for instance 'i-0743f1ad010e242ec (Amazon Linux 2)'. The breadcrumb trail is 'EC2 > Instances > i-0743f1ad010e242ec > Connect to instance'. The page title is 'Connect to instance' with an 'Info' link. A note states: 'Connect to your instance i-0743f1ad010e242ec (Amazon Linux 2) using any of these options'. There are four tabs: 'EC2 Instance Connect' (selected), 'Session Manager', 'SSH client', and 'EC2 serial console'. The instance details are as follows:

| Instance ID                          | Public IP address |
|--------------------------------------|-------------------|
| i-0743f1ad010e242ec (Amazon Linux 2) | 3.86.109.58       |

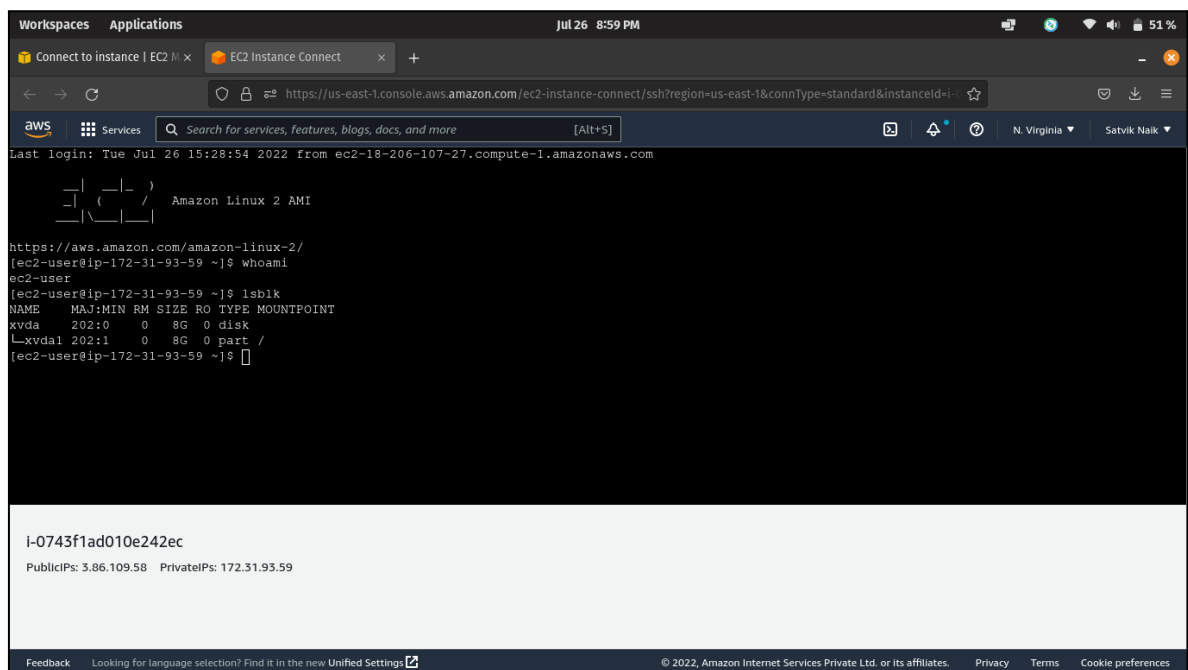
User name:

Connect using a custom user name, or use the default user name ec2-user for the AMI used to launch the instance.

**Note:** In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

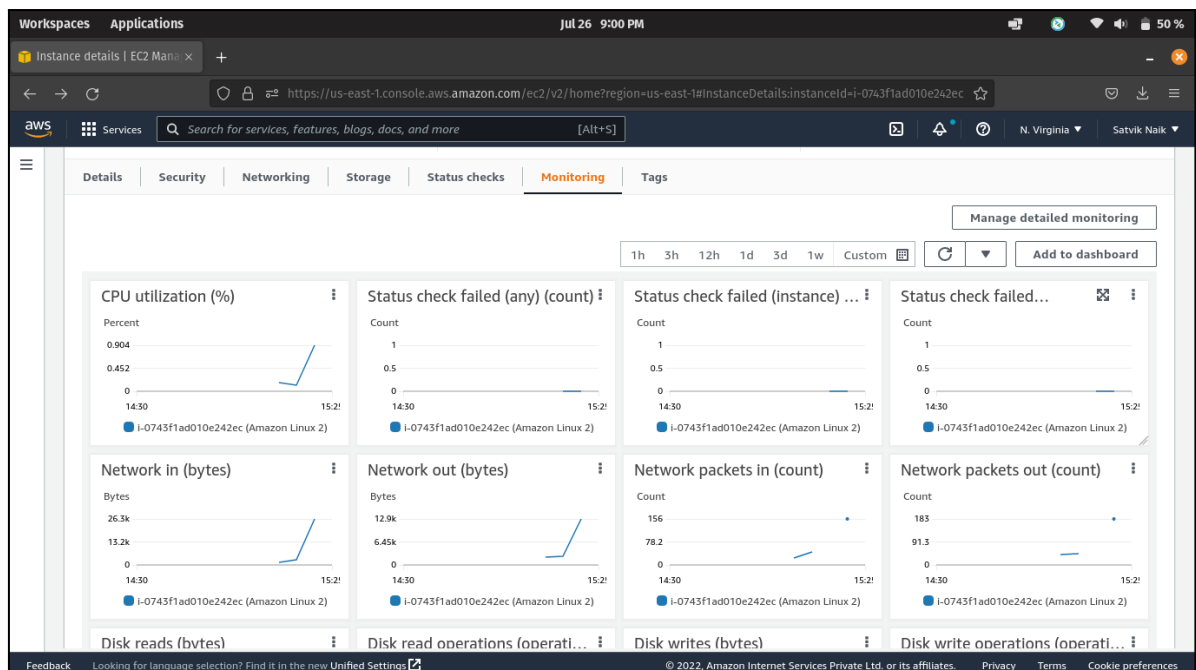
At the bottom right, there are 'Cancel' and 'Connect' buttons.

**ICT GANPAT UNIVERSITY**  
**COMPUTER SCIENCE DEPARTMENT**  
**Cloud Computing Essentials (2CSE710)**



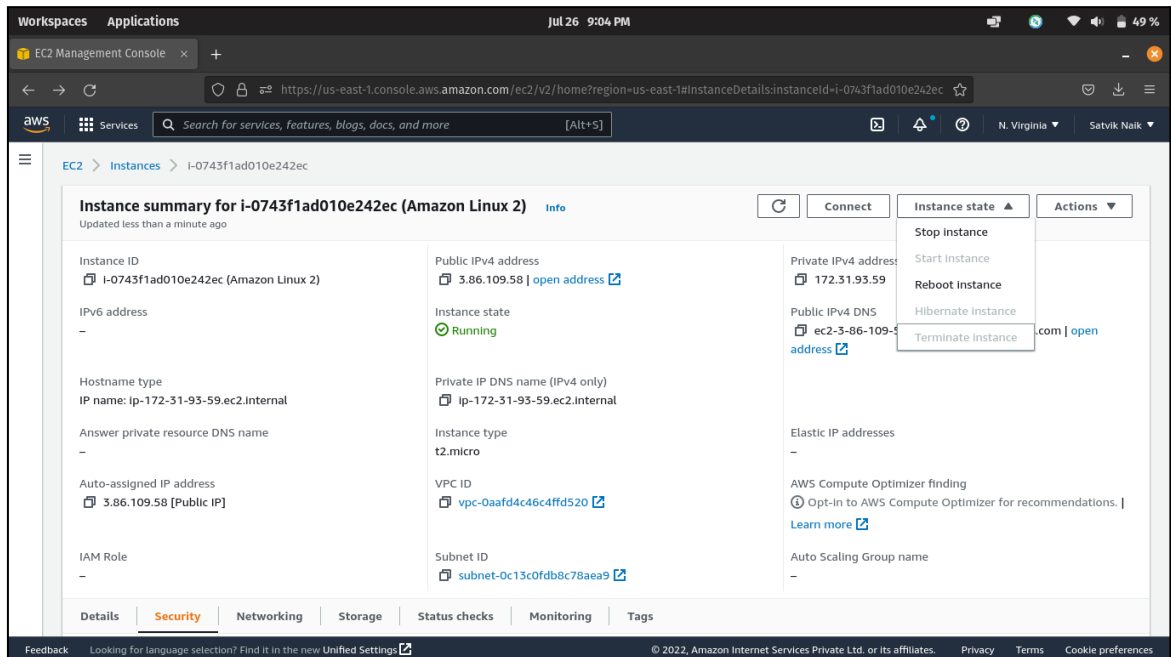
## 5. Monitoring EC2 Instance.

- EC2 instances can be monitored by visualizing various parameters such as disk R/W operations, CPU usage, network packets etc...

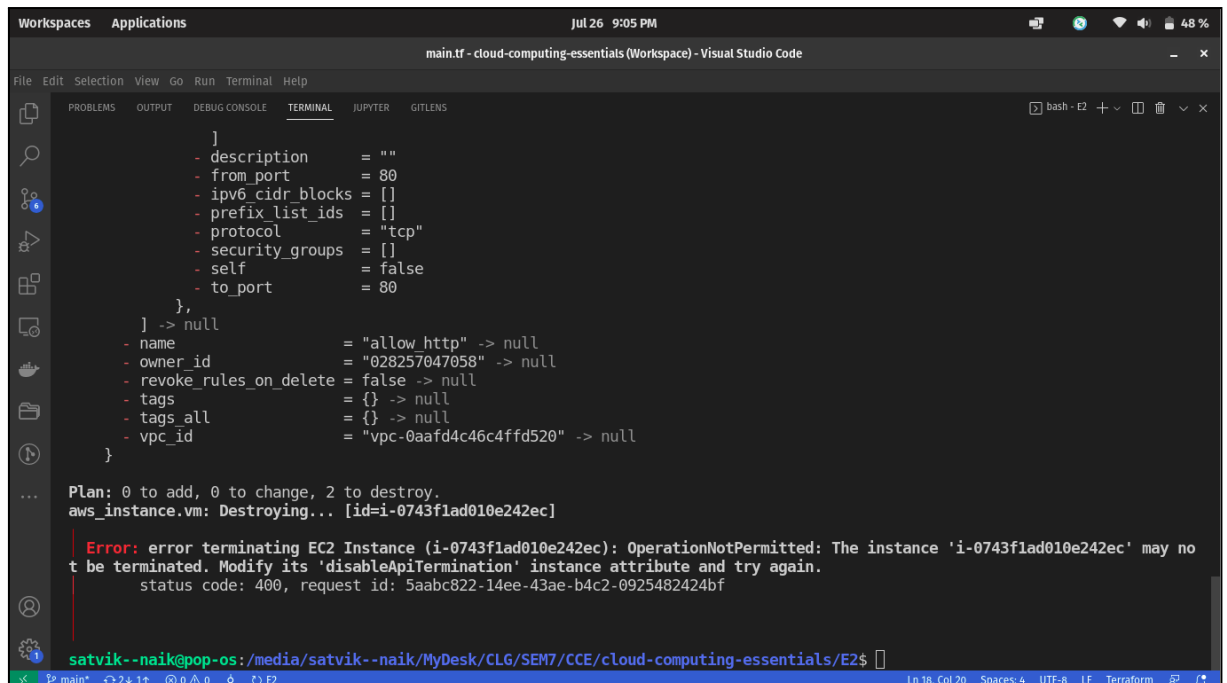


## 6. Test termination protection.

- In case you try to terminate the instance, it will prompt you with a message showing ***“Termination protection is enabled for one or more selected instances.”***



- In case if you try to do it via ***“terraform destroy”*** command then also it will show you the following error message. (i.e. OperationNotPermitter).

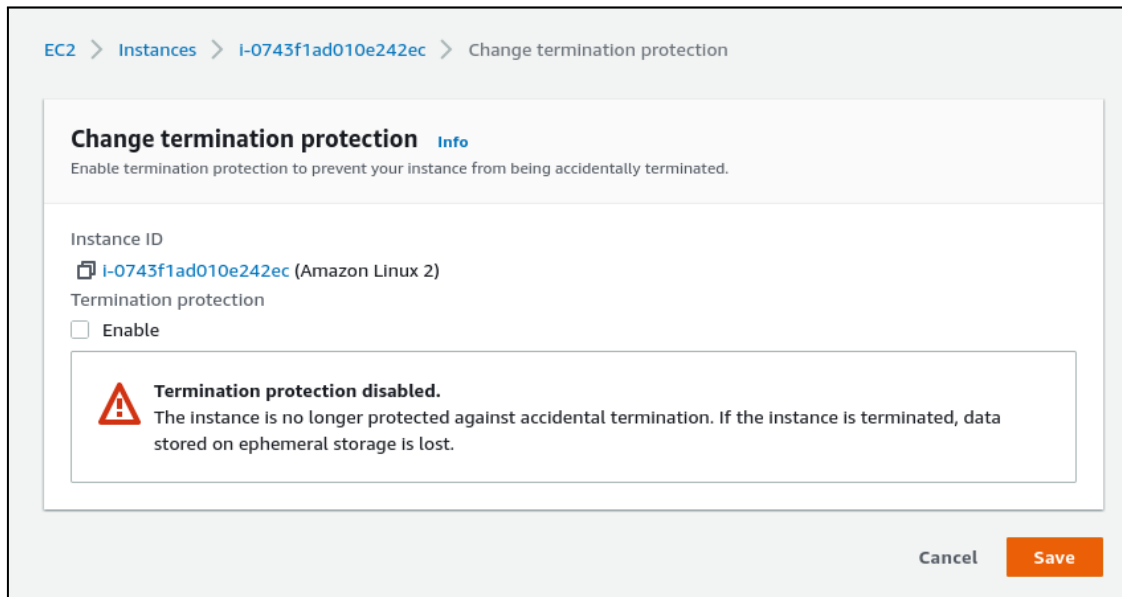
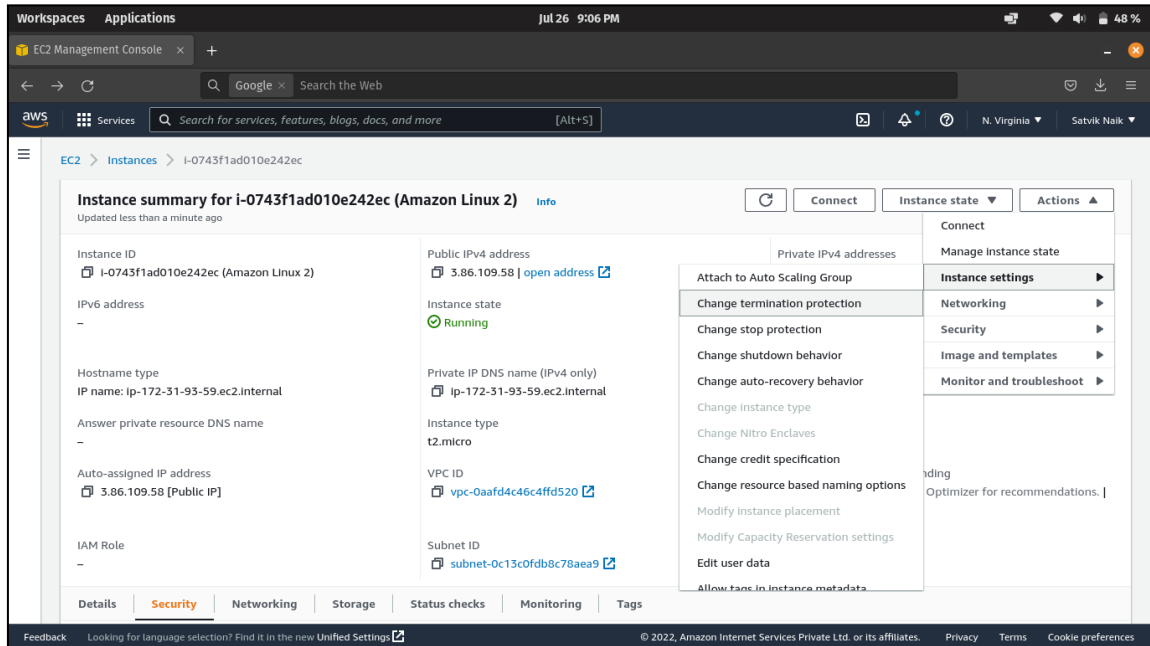




**ICT GANPAT UNIVERSITY**  
**COMPUTER SCIENCE DEPARTMENT**  
**Cloud Computing Essentials (2CSE710)**

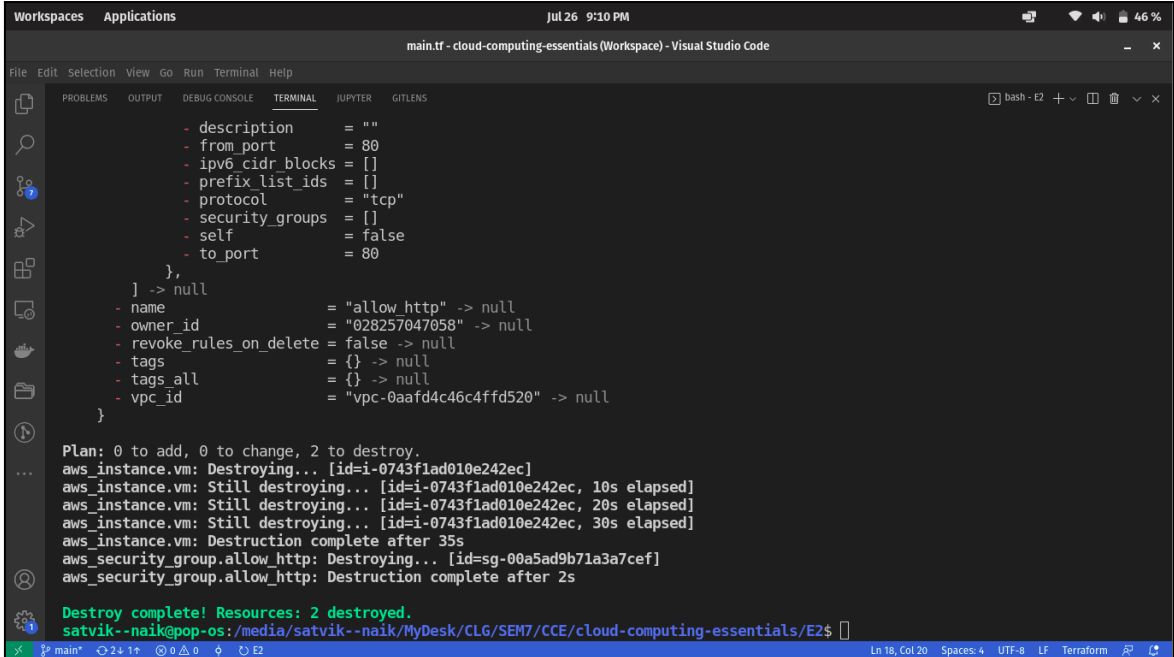
## 7. Terminate your EC2 Instance.

- In order to terminate your instance, first disable the termination protection.



Click on **'Save'**.

- Now on trying to give the '**terraform destroy**' command, it will terminate the instance.



```
main.tf - cloud-computing-essentials (Workspace) - Visual Studio Code

- description      = ""
- from_port        = 80
- ipv6_cidr_blocks = []
- prefix_list_ids  = []
- protocol         = "tcp"
- security_groups  = []
- self             = false
- to_port          = 80
},
] -> null
- name              = "allow_http" -> null
- owner_id          = "028257047058" -> null
- revoke_rules_on_delete = false -> null
- tags              = {} -> null
- tags_all          = {} -> null
- vpc_id            = "vpc-0aafd4c46c4ffd520" -> null
}

Plan: 0 to add, 0 to change, 2 to destroy.
aws_instance.vm: Destroying... [id=i-0743f1ad010e242ec]
aws_instance.vm: Still destroying... [id=i-0743f1ad010e242ec, 10s elapsed]
aws_instance.vm: Still destroying... [id=i-0743f1ad010e242ec, 20s elapsed]
aws_instance.vm: Still destroying... [id=i-0743f1ad010e242ec, 30s elapsed]
aws_instance.vm: Destruction complete after 35s
aws_security_group.allow_http: Destroying... [id=sg-00a5ad9b71a3a7cef]
aws_security_group.allow_http: Destruction complete after 2s

Destroy complete! Resources: 2 destroyed.
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E2$
```

## 8. EC2 limits.

- 20 instances per region.
- 5 Elastic IP Addresses.
- Price variations.