1. AWS CloudFormation (AWS Service)

AWS CloudFormation performs the crazy party trick of enabling you to manage your complete AWS infrastructure and resources from a text file. The formatted YAML or JSON code you write in the AWS CloudFormation template describes your AWS infrastructure and the resources you need. CloudFormation does the rest, provisioning, configuring and deploying everything for you. It also handles dependencies between resources, removing another piece of complexity from the puzzle.

Without a template, you would have to set everything up manually using the AWS management console or CLI. You would also have to make note of all the resources involved, especially if you wanted to replicate your work for another environment. On the other hand, templates can be used on an ongoing basis, moving you away from the tedium of manually executing multiple steps every time you need to make a change. For example, extending your environment by adding a few more functions are easy with a template.

Sample templates

Pricing

You only pay for what you use, with no minimum fees and no required upfront commitments. There is no additional charge for using AWS CloudFormation with resource providers in the following namespaces: AWS::*, Alexa::*, and Custom::*. In these cases, you pay for AWS resources such as Amazon Elastic Compute Cloud (EC2) instances, Elastic Load Balancing load balancers, etc. created using AWS CloudFormation the same as if you had created them manually.

Provisioning an EC2 instance with CloudFormation

1st of all we want to configure AWS CLI.

After that create **ec2.yaml** and add following code.

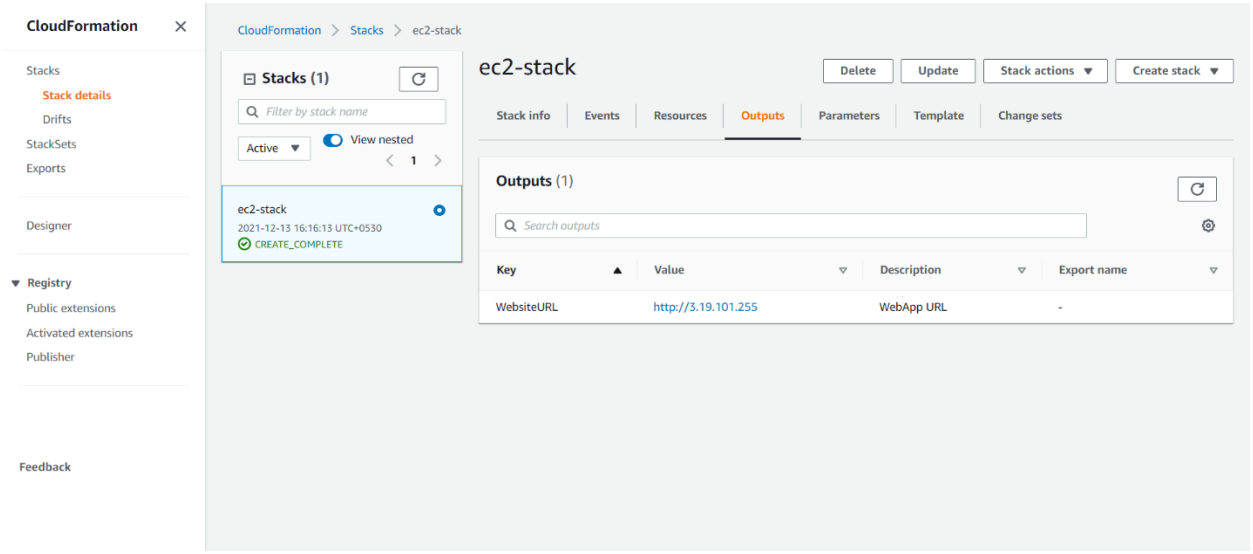
```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: EC2 Instance Create Using CloudFormation

Resources:
  WebAppInstance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-002068ed284fb165b #AMI ID Unique to Region
      InstanceType: t2.micro
      KeyName: ohio-key #Add keypair name
      SecurityGroupIds:
        - !Ref WebAppSecurityGroup

  WebAppSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: !Join [ '-', [ webapp-security-group, dev ] ]
      GroupDescription: 'Allow HTTP/HTTPS and SSH inbound and outbound traffic'
      SecurityGroupIngress:
        - IpProtocol: tcp
```

```
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0

  WebAppEIP:
    Type: AWS::EC2::EIP
    Properties:
      Domain: vpc
      InstanceId: !Ref WebAppInstance
      Tags:
        - Key: Name
          Value: !Join [ '-', [ webapp-eip, dev ] ]

Outputs:
  WebsiteURL:
    Value: !Sub http://${WebAppEIP}
    Description: WebApp URL
```

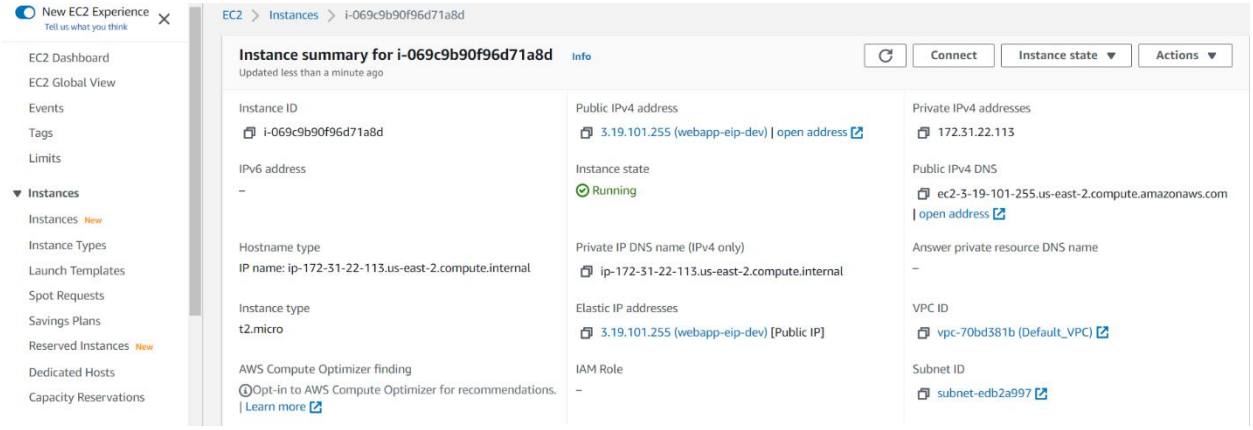
After that run following command for create CloudFormation Stack.

aws cloudformation create-stack --stack-name ec2-stack --template-body file://ec2.yaml --output yaml

Go to AWS console and search CloudFormation and you can see the newly created stack and you can see output details.



After that go to EC2 Service and you can see the newly created EC2 instance.



If you want you can update stack details using following command.

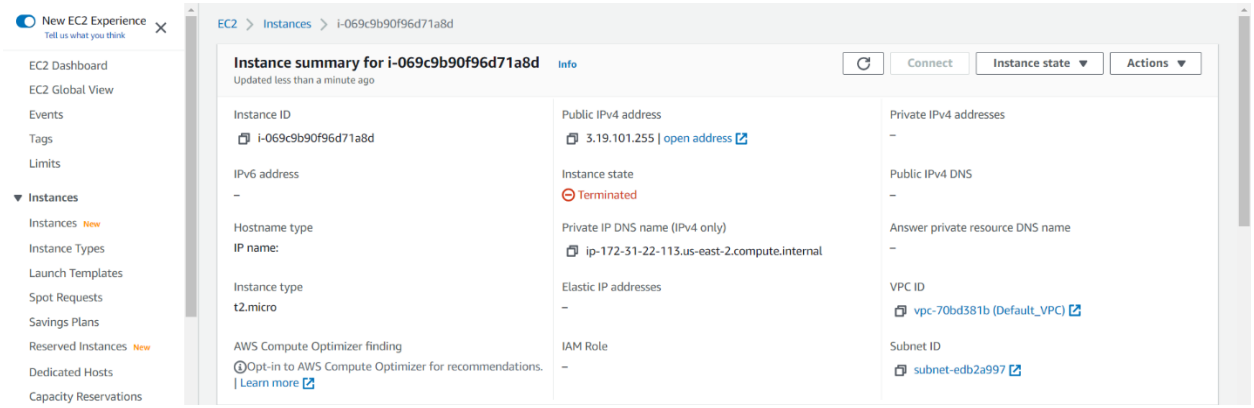
aws cloudformation update-stack --stack-name ec2-stack --template-body file://ec2.yaml --output yaml

Finally, you can destroy AWS Infrastructure using the following command. Please remind that if you destroy you will lose created infrastructure.

aws cloudformation delete-stack --stack-name ec2-stack --output yaml

```
PS C:\Users\LasanthaSanjeewaSilv\Desktop\CloudFormation> aws cloudformation delete-stack --stack-name ec2-stack --output yaml
PS C:\Users\LasanthaSanjeewaSilv\Desktop\CloudFormation>
```

After that go to AWS Console and you can see Your EC2 Instance is terminated.

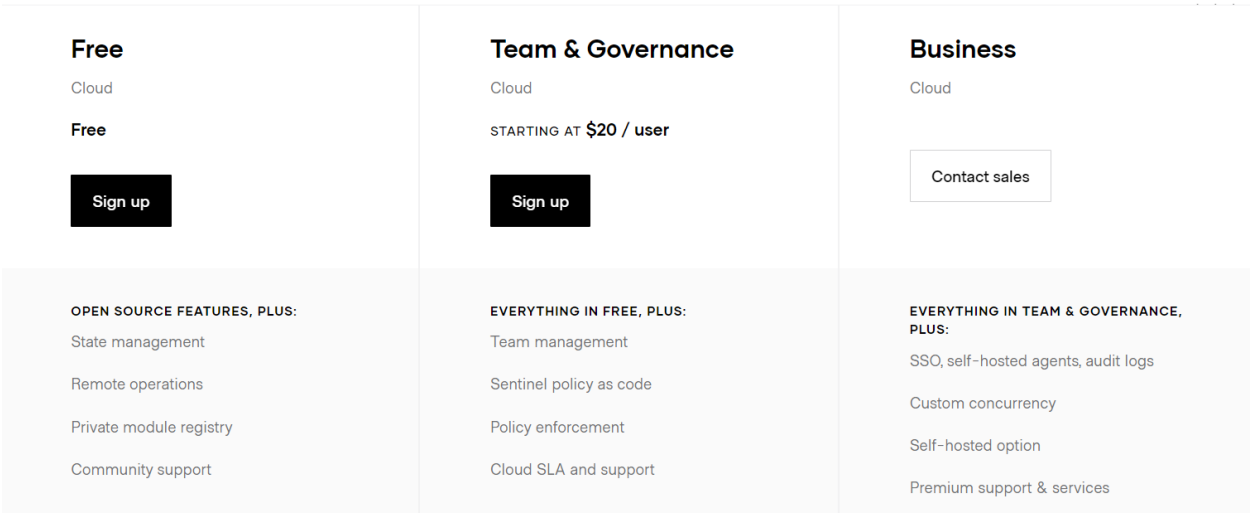


2. Terraform - by HashiCorp

Terraform is an infrastructure as code (IaC) tool that allows you to build, change, and version infrastructure safely and efficiently. This includes low-level components such as compute instances, storage, and networking, as well as high-level components such as DNS entries, SaaS features, etc. Terraform can manage both existing service providers and custom in-house solutions.

Terraform allows you to create infrastructure in configuration files(tf files) that describe the topology of cloud resources. These resources include virtual machines, storage accounts, and networking interfaces.

Pricing



* **We can use Free Version for IaC Provision.**

Provisioning an EC2 instance with Terraform

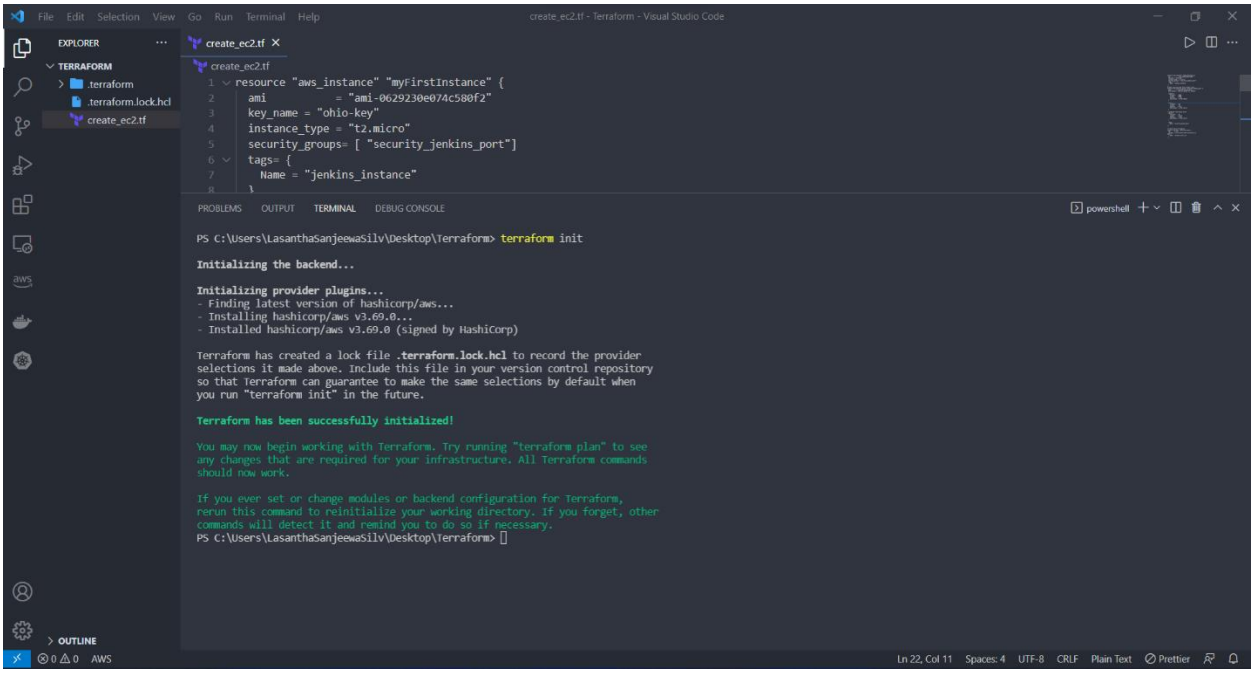
- Create a local directory in your machine and go inside this folder. After that configure aws cli.
- Create **create_ec2.tf** file and added the following code here.

```
resource "aws_instance" "myFirstInstance" {
 ami           = "ami-0629230e074c580f2" #unique to region
 key_name = "ohio-key" #use your pemfile
 instance_type = "t2.micro"
 security_groups= [ "security_jenkins_port"]
 tags= {
  Name = "jenkins_instance"
```

```
 }
}
#Create security group with firewall rules
resource "aws_security_group" "security_jenkins_port" {
 name        = "security_jenkins_port"
 description = "security group for jenkins"

 ingress {
  from_port   = 8080
  to_port     = 8080
  protocol    = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
 }

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

# outbound from jenkis server
 egress {
  from_port   = 0
  to_port     = 65535
  protocol    = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
 }
```

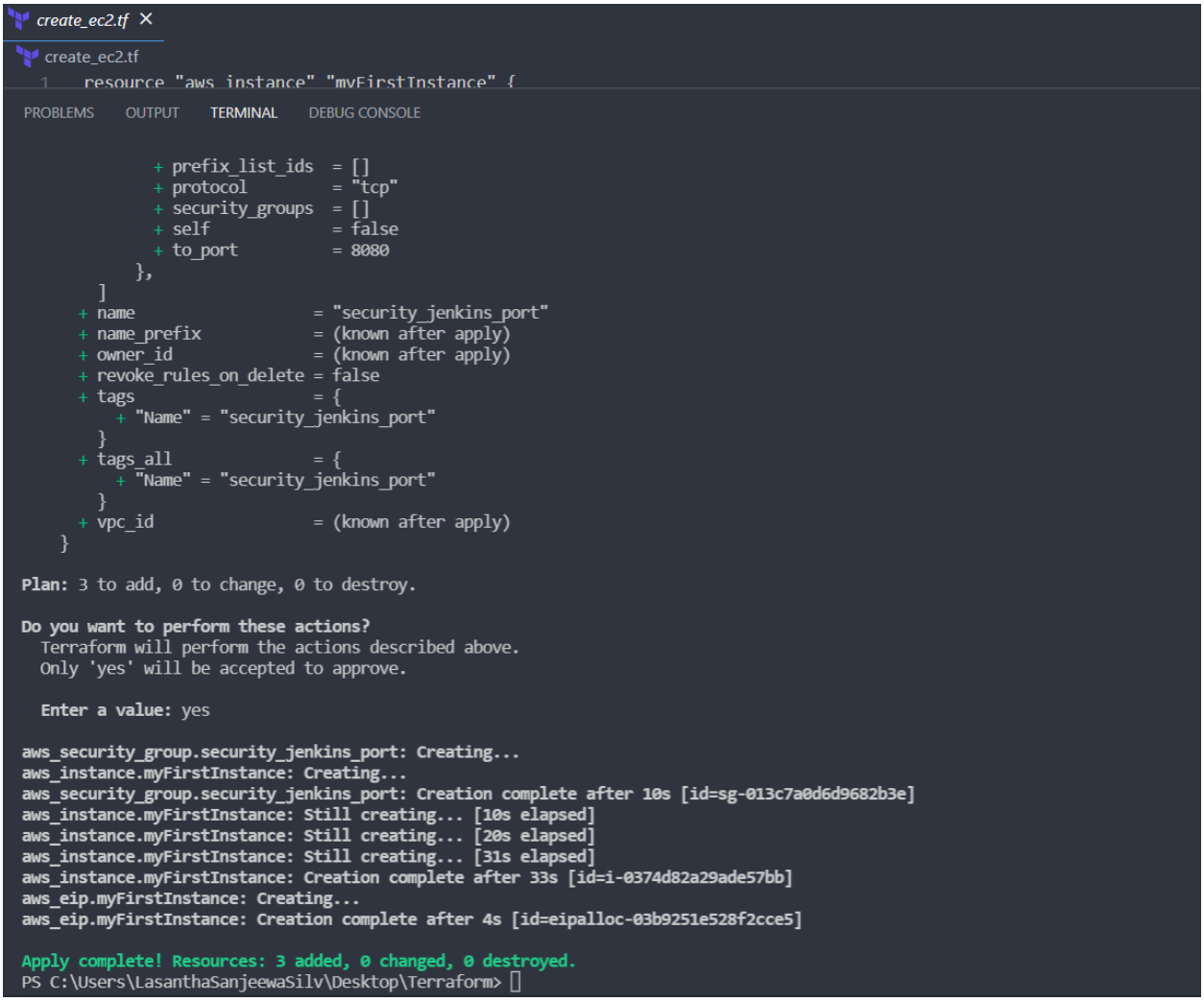
```
  tags= {
    Name = "security_jenkins_port"
  }
}

# Create Elastic IP address
resource "aws_eip" "myFirstInstance" {
  vpc      = true
  instance = aws_instance.myFirstInstance.id
tags= {
    Name = "jenkins_elstic_ip"
  }
}
```

- Run “terraform init” command will initialize terraform project. You can see like that.

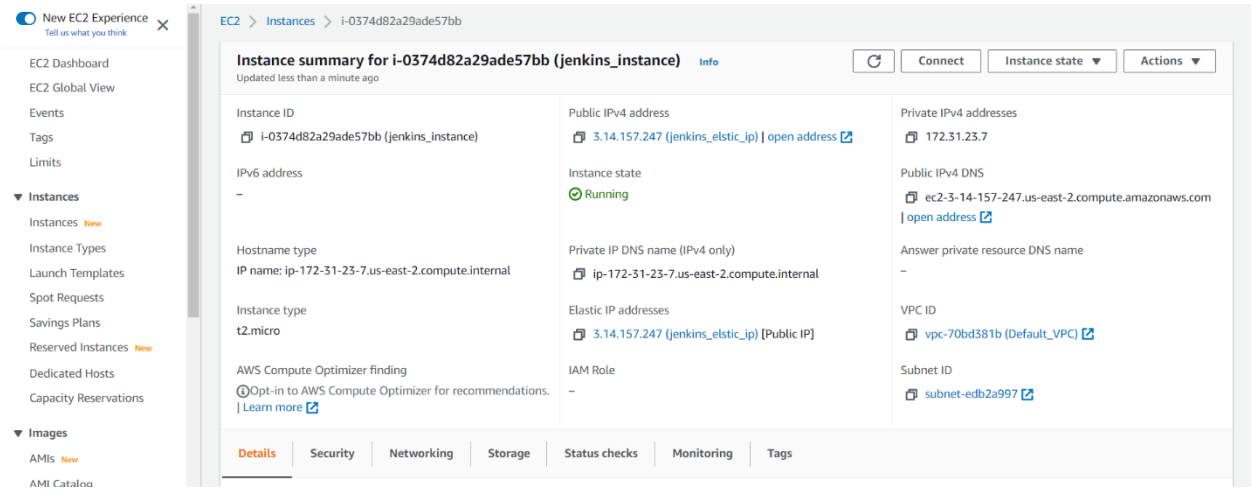


- Run “terraform plan” command will show how many resources will be added.

- Run “terraform apply” command and enter “yes”. After that, you can see your ec2 instance and security group created.



- Go to AWS Console and see newly created EC2 Instance.



- Run “terraform state list” for a view list of the resources created by Terraform.

```
PS C:\Users\LasanthaSanjeewaSilv\Desktop\Terraform> terraform state list
aws_eip.myFirstInstance
aws_instance.myFirstInstance
aws_security_group.security_jenkins_port
PS C:\Users\LasanthaSanjeewaSilv\Desktop\Terraform>
```

- Finally, run “terraform destroy” to remove previously created all services. After that run the “terraform state list” command you haven’t seen anything.

```
Plan: 0 to add, 0 to change, 3 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

aws_eip.myFirstInstance: Destroying... [id=eipalloc-03b9251e528f2cce5]
aws_security_group.security_jenkins_port: Destroying... [id=sg-013c7a0d6d9682b3e]
aws_eip.myFirstInstance: Destruction complete after 4s
aws_instance.myFirstInstance: Destroying... [id=i-0374d82a29ade57bb]
aws_security_group.security_jenkins_port: Still destroying... [id=sg-013c7a0d6d9682b3e, 10s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0374d82a29ade57bb, 10s elapsed]
aws_security_group.security_jenkins_port: Still destroying... [id=sg-013c7a0d6d9682b3e, 20s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0374d82a29ade57bb, 20s elapsed]
aws_security_group.security_jenkins_port: Still destroying... [id=sg-013c7a0d6d9682b3e, 30s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0374d82a29ade57bb, 30s elapsed]
aws_security_group.security_jenkins_port: Still destroying... [id=sg-013c7a0d6d9682b3e, 40s elapsed]
aws_instance.myFirstInstance: Still destroying... [id=i-0374d82a29ade57bb, 40s elapsed]
aws_security_group.security_jenkins_port: Destruction complete after 46s
aws_instance.myFirstInstance: Destruction complete after 47s

Destroy complete! Resources: 3 destroyed.
PS C:\Users\LasanthaSanjeewaSilv\Desktop\Terraform> terraform state list
PS C:\Users\LasanthaSanjeewaSilv\Desktop\Terraform>
```