Jenkins-based Terraform deployment pipeline on AWS

Implementing the **Blue/Green Deployment Pipeline** using **Terraform**, **Jenkins**, and **AWS Cloud Infrastructure**. The setup automatically provisions EC2 instances, Load Balancer, Security Groups, and optionally destroys infrastructure. It uses **Jenkins pipelines** to trigger infrastructure changes based on user input (blue or green environment).

Installing **Jenkins**, a leading open-source automation server, **EC2 instance or physical/virtual machine**. Jenkins enables continuous integration/continuous delivery (CI/CD) pipelines to automate the build, test, and deployment of applications.

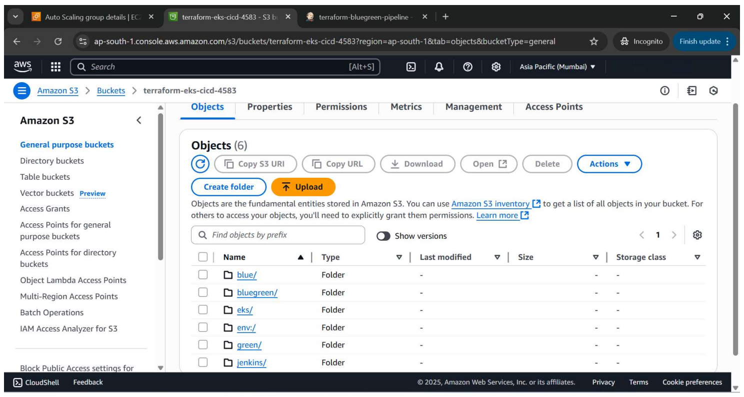
This automation includes:

- Using **Terraform** to provision AWS infrastructure.
- Using **Jenkins** as the CI/CD tool to trigger Terraform commands.
- Managing blue and green environments dynamically using variables.
- Leveraging **ALB (Application Load Balancer)** and EC2/ASG for deployment.

Prerequisites:

Before executing this setup, ensure the following:

- AWS Account
- AWS IAM Credentials in Jenkins
- GitHub Repository
- Jenkins Installed on EC2
- Terraform Installed (on Jenkins)
- SSH Key Pair
- Public Subnets and VPC
- S3 Bucket for Terraform Backend


Auto Scaling group details | EC
terraform-eks-cicd-4583 - S3 b
terraform-bluegreen-pipeline
ap-south-1.console.aws.amazon.com/s3/buckets/terraform-eks-cicd-4583?region=ap-south-1&tab=objects&bucketType=general
Incognito
Finish update
Search [Alt+S]
Asia Pacific (Mumbai)
Amazon S3 > Buckets > terraform-eks-cicd-4583
Objects
Properties
Permissions
Metrics
Management
Access Points
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets Preview
Access Grants
Access Points for general purpose buckets
Access Points for directory buckets
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
Show versions
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Objects (6)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
blue/	Folder	-	-	-
bluegreen/	Folder	-	-	-
eks/	Folder	-	-	-
env:/	Folder	-	-	-
green/	Folder	-	-	-
jenkins/	Folder	-	-	-

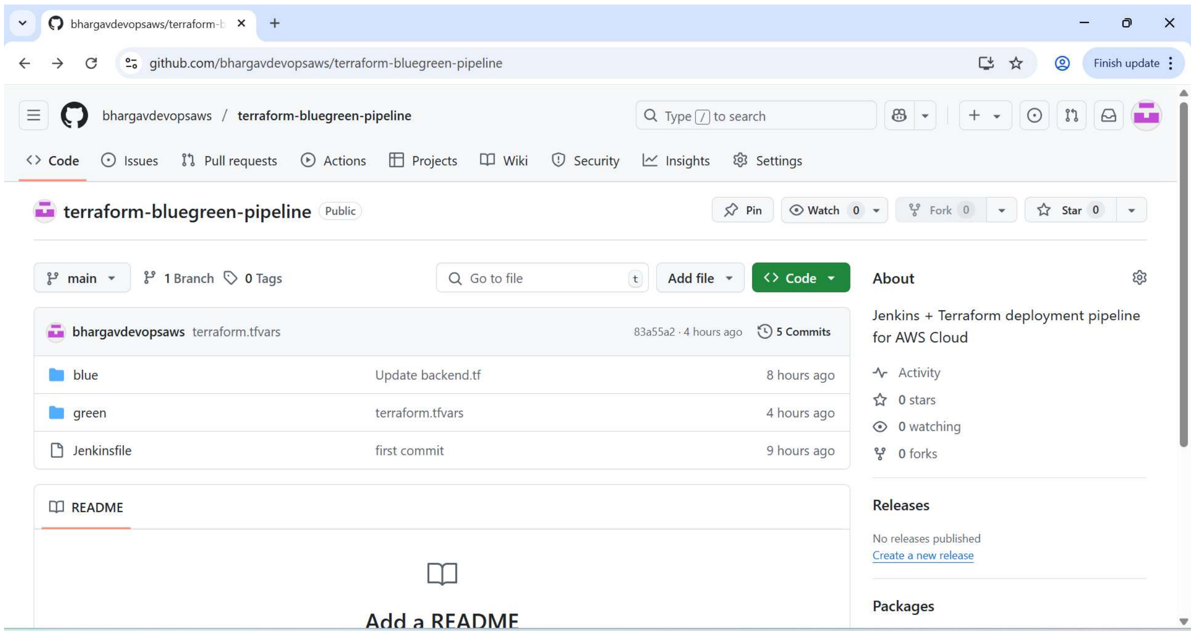
Tools Used:

- Terraform - Infrastructure provisioning (IaaC).
- Jenkins - CI/CD tool to automate deployment.
- GitHub - Version control for Terraform configurations.
- AWS - Cloud provider hosting the infrastructure.

Commit and Push Blue/Green Code to GitHub

Folder Structure:

```
terraform-bluegreen-pipeline/

├── Jenkinsfile

├── blue/

│   ├── main.tf

│   ├── variables.tf

│   ├── terraform.tfvars

│   ├── backend.tf

│   └── outputs.tf

├── green/

│   ├── main.tf

│   ├── variables.tf

│   ├── terraform.tfvars

│   ├── backend.tf

│   └── outputs.tf
```


github.com/bhargavdevopsaws/terraform-bluegreen-pipeline

bhargavdevopsaws / terraform-bluegreen-pipeline

terraform-bluegreen-pipeline Public

main · 1 Branch · 0 Tags

bhargavdevopsaws terraform.tfvars · 83a55a2 · 4 hours ago · 5 Commits

blue	Update backend.tf	8 hours ago
green	terraform.tfvars	4 hours ago
Jenkinsfile	first commit	9 hours ago

About

Jenkins + Terraform deployment pipeline for AWS Cloud

Activity · 0 stars · 0 watching · 0 forks

Releases: No releases published. Create a new release

README: Add a README

Blue:

Main.tf:

```
##############################
# EC2 Instance
##############################
resource "aws_instance" "web" {
  ami           = "ami-021a584b49225376d"
  instance_type = "t2.medium"
  subnet_id     = element(var.subnet_ids, 0)
  key_name      = "awsdevops"

  tags = {
    Name        = "app-${var.environment}"
    Environment = var.environment
  }
}

##############################
# Security Group (Allow HTTP)
##############################
resource "aws_security_group" "alb_sg" {
  name        = "alb-sg"
  description = "Allow HTTP"
  vpc_id      = var.vpc_id

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

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

#############################
# Application Load Balancer
#############################
resource "aws_lb" "app_alb" {
  name               = "app-alb-${var.environment}"
  internal           = false
  load_balancer_type = "application"
  security_groups    = [aws_security_group.alb_sg.id]
  subnets            = var.subnet_ids

  tags = {
    Environment = var.environment
  }
}

#############################
# Target Group
#############################
resource "aws_lb_target_group" "app_tg" {
  name     = "${var.environment}-tg"
  port     = 80
  protocol = "HTTP"
  vpc_id   = var.vpc_id

  health_check {
    path                = "/"
    interval            = 30
    timeout             = 5
    healthy_threshold   = 2
    unhealthy_threshold = 2
    matcher             = "200-399"
  }

  tags = {
    Environment = var.environment
  }
}

#############################
# Attach Instance to Target Group
#############################
resource "aws_lb_target_group_attachment" "attach_app" {
  target_group_arn = aws_lb_target_group.app_tg.arn
  target_id        = aws_instance.web.id
  port             = 80
}

#############################
# Listener (on port 80)
#############################
resource "aws_lb_listener" "my_listener" {
  load_balancer_arn = aws_lb.app_alb.arn
  port              = 80
  protocol          = "HTTP"

  default_action {
    type = "fixed-response"

    fixed_response {
      content_type = "text/plain"
      message_body = "Default response"
      status_code  = "200"
    }
  }
}
```

```
##############################
# Listener Rule (for Blue/Green)
##############################
resource "aws_lb_listener_rule" "bluegreen_weight" {
  listener_arn = aws_lb_listener.my_listener.arn
  priority     = var.environment == "blue" ? 100 : 200

  action {
    type = "forward"
    forward {
      target_group {
        arn    = aws_lb_target_group.app_tg.arn
        weight = var.environment == "blue" ? 100 : 0
      }
    }
  }

  condition {
    path_pattern {
      values = ["/*"]
    }
  }
}
```

Variables.tf:

```
variable "aws_region" {
  default     = "ap-south-1"
  description = "AWS Region"
}
variable "instance_type" {
  default     = "t2.medium"
  description = "ubuntu"
}
variable "subnet_ids" {
  type        = list(string)
  description = "List of subnet IDs for ALB and EC2"
}
variable "vpc_id" {
  type        = string
  description = "VPC ID for ALB and EC2"
}
variable "environment" {
  default     = "blue"
  description = "Environment name: blue or green"
}
```

Backend.tf:

```
terraform {

  backend "s3" {

    bucket         = "terraform-eks-cicd-4583"
    key            = "blue/terraform.tfstate"
    region         = "ap-south-1"
  }
}
```

Variables.tf:

```
provider "aws" {
  region = var.aws_region
}
variable "aws_region" {
  default     = "ap-south-1"
  description = "AWS Region"
}

variable "instance_type" {
  default     = "t2.medium"
  description = "ubuntu"
}

variable "subnet_ids" {
  type        = list(string)
  description = "List of subnet IDs for ALB and EC2"
}

variable "vpc_id" {
  type        = string
  description = "VPC ID for ALB and EC2"
}

variable "environment" {
  default     = "blue"
  description = "Environment name: blue or green"
```

Terraform.tfvars:

```
aws_region    = "ap-south-1"
instance_type = "t2.micro"
environment   = "blue"
vpc_id        = "vpc-0c07faa337fa997e9"
subnet_ids = ["subnet-0976223a60da6c274" , "subnet-0a82d80eebc5ba227"]
```

Green:

Main.tf:

```
locals {
  is_blue  = var.environment == "blue"
  is_green = var.environment == "green"
}

provider "aws" {
  region = "ap-south-1"
}

# ======================
# Security Group for ALB
# ======================
resource "aws_security_group" "alb_sg" {
  name        = "alb-sg-${var.environment}"
  description = "Allow HTTP traffic"
  vpc_id      = var.vpc_id

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

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

# ======================
# ALB
# ======================
resource "aws_lb" "app_alb" {
  name               = "alb-${var.environment}"
  internal           = false
  load_balancer_type = "application"
  security_groups    = [aws_security_group.alb_sg.id]
  subnets            = var.subnet_ids

  tags = {
    Environment = var.environment
  }
}


# ======================
# Target Group
# ======================
resource "aws_lb_target_group" "app_tg" {
  name     = "${var.environment}-tg"
  port     = 80
  protocol = "HTTP"
  vpc_id   = var.vpc_id

  health_check {
    path                = "/"
    interval            = 30
    timeout             = 5
    healthy_threshold   = 2
    unhealthy_threshold = 2
    matcher             = "200-399"
  }

  tags = {
    Environment = var.environment
  }
}

# ======================
# Listener
# ======================
resource "aws_lb_listener" "app_listener" {
  load_balancer_arn = aws_lb.app_alb.arn
  port              = 80
  protocol          = "HTTP"

  default_action {
    type = "forward"
    target_group_arn = aws_lb_target_group.app_tg.arn
  }
}

# ======================
# Blue Deployment (EC2)
# ======================
resource "aws_instance" "web" {
  count         = local.is_blue ? 1 : 0
  ami           = var.ami_id
  instance_type = var.instance_type
  subnet_id     = element(var.subnet_ids, 0)
  key_name      = "awsdevops"

  vpc_security_group_ids = [aws_security_group.alb_sg.id]

  tags = {
    Name = "app-${var.environment}"
  }
}

resource "aws_lb_target_group_attachment" "attach_instance" {
  count            = local.is_blue ? 1 : 0
  target_group_arn = aws_lb_target_group.app_tg.arn
  target_id        = aws_instance.web[0].id
  port             = 80
}

# =======================
# Green Deployment (ASG)
# =======================
resource "aws_launch_template" "app" {
  count         = local.is_green ? 1 : 0
  name_prefix   = "app-${var.environment}-"
  image_id      = var.ami_id
  instance_type = var.instance_type

  tag_specifications {
    resource_type = "instance"
    tags = {
      Name = "app-${var.environment}"
    }
  }
}

resource "aws_autoscaling_group" "app" {
  count               = local.is_green ? 1 : 0
  name                = "asg-${var.environment}"
  desired_capacity    = var.asg_min_size
  max_size            = var.asg_max_size
  min_size            = var.asg_min_size
  vpc_zone_identifier = var.subnet_ids

  launch_template {
    id      = aws_launch_template.app[0].id
    version = "$Latest"
  }

  target_group_arns = [aws_lb_target_group.app_tg.arn]

  tag {
    key                 = "Name"
    value               = "app-${var.environment}"
    propagate_at_launch = true
  }
}
```

Variables.tf:

```
variable "aws_region" {
  default     = "ap-south-1"
  description = "AWS Region"
}
variable "instance_type" {
  default     = "t2.micro"
  description = "EC2 instance type"
}
```

```
variable "ami_id" {
  description = "AMI ID for the instance or launch template"
  type        = string
}
variable "vpc_id" {
  description = "VPC ID"
  type        = string
}
variable "subnet_ids" {
  description = "List of subnet IDs"
  type        = list(string)
}
variable "key_name" {
  description = "Key pair name"
  type        = string
}
variable "environment" {
  description = "Environment name: blue or green"
  type        = string
}
variable "asg_min_size" {
  default     = 1
  description = "Minimum size of the ASG"
}
variable "asg_max_size" {
  default     = 2
  description = "Maximum size of the ASG"
}
```

Terraform.tfvars:

```
environment     = "green"
vpc_id          = "vpc-0c07faa337fa997e9"
subnet_ids      = ["subnet-0976223a60da6c274", "subnet-0861408dec083aeea"]
instance_type   = "t2.medium"
asg_min_size    = 1
asg_max_size    = 2
```

Backend.tf:

```
terraform {
  backend "s3" {
    bucket         = "terraform-eks-cicd-4583"
    key            = "bluegreen/terraform.tfstate"
    region         = "ap-south-1"
  }
}
```

Outputs.tf:

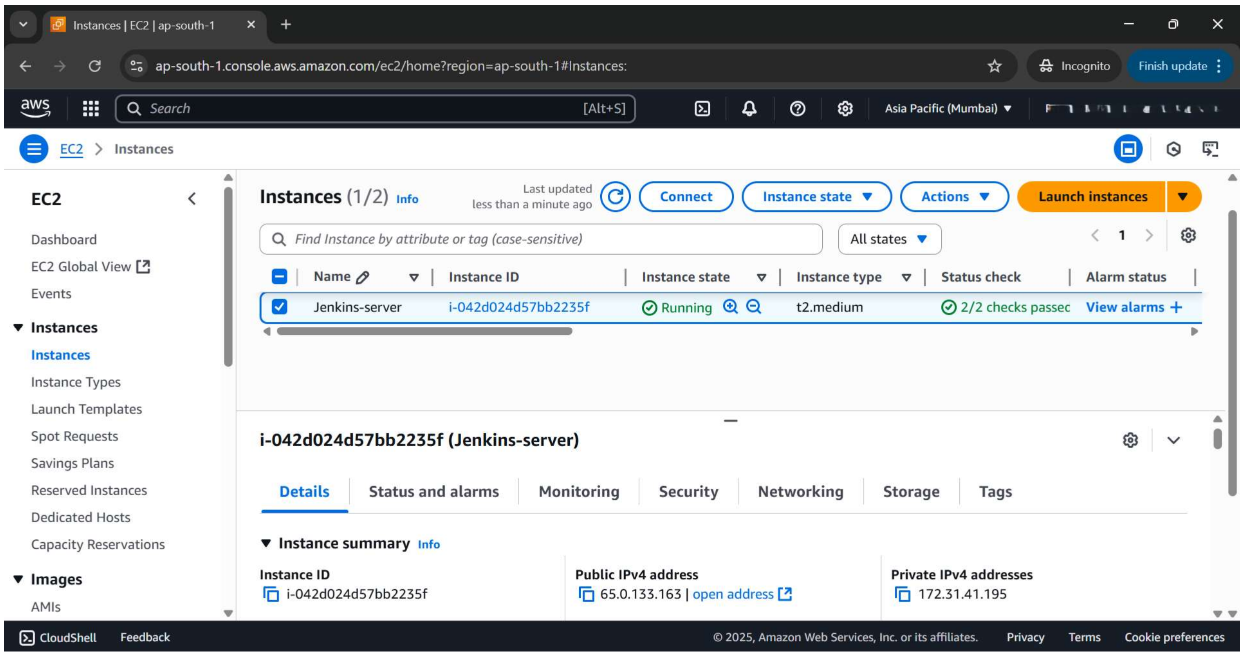
```
output "alb_dns_name" {
  value = aws_lb.app_alb.dns_name
}

output "instance_public_ip" {
  value = length(aws_instance.web) > 0 ? aws_instance.web[0].public_ip : ""
}
```

Jenkinsfile:

```
pipeline {
  agent any

  parameters {
    string(name: 'ENVIRONMENT', defaultValue: 'blue', description: 'Environment (blue/green)')
  }

  stages {

    stage('Checkout') {
      steps {
        git branch: 'main', url: 'https://github.com/bhargavdevopsaws/terraform-bluegreen-pipeline.git'
      }
    }

    stage('Verify Environment Folder') {
      steps {
        sh 'echo "Root directory content:" && ls -l'
        sh "echo \"Contents of '${params.ENVIRONMENT}/':\" && ls -l ${params.ENVIRONMENT}"
      }
    }

    stage('Terraform Init') {
      steps {
        withCredentials([[ $class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'aws-access' ]]) {
          dir("${params.ENVIRONMENT}") {
            sh 'terraform init'
          }
        }
      }
    }

    stage('Terraform Plan') {
      steps {
        withCredentials([[ $class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'aws-access' ]]) {
          dir("${params.ENVIRONMENT}") {
            sh 'terraform plan -var-file="terraform.tfvars" -out=tfplan'
          }
        }
      }
    }

    stage('Terraform Apply') {
      steps {
        withCredentials([[ $class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'aws-access' ]]) {
          dir("${params.ENVIRONMENT}") {
            sh 'terraform apply tfplan'
          }
        }
      }
    }

    stage('Terraform Output') {
      steps {
        withCredentials([[ $class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'aws-access' ]]) {
          dir("${params.ENVIRONMENT}") {
            sh 'terraform output'
          }
        }
      }
    }
  }
}
```

Installation of Jenkins & Terraform on EC2:

Create an Ubuntu **EC2 instance and install Jenkins & Terraform:**



Name	Instance ID	Instance state	Instance type	Status check	Alarm status
Jenkins-server	i-042d024d57bb2235f	Running	t2.medium	2/2 checks passec	View alarms

i-042d024d57bb2235f (Jenkins-server)

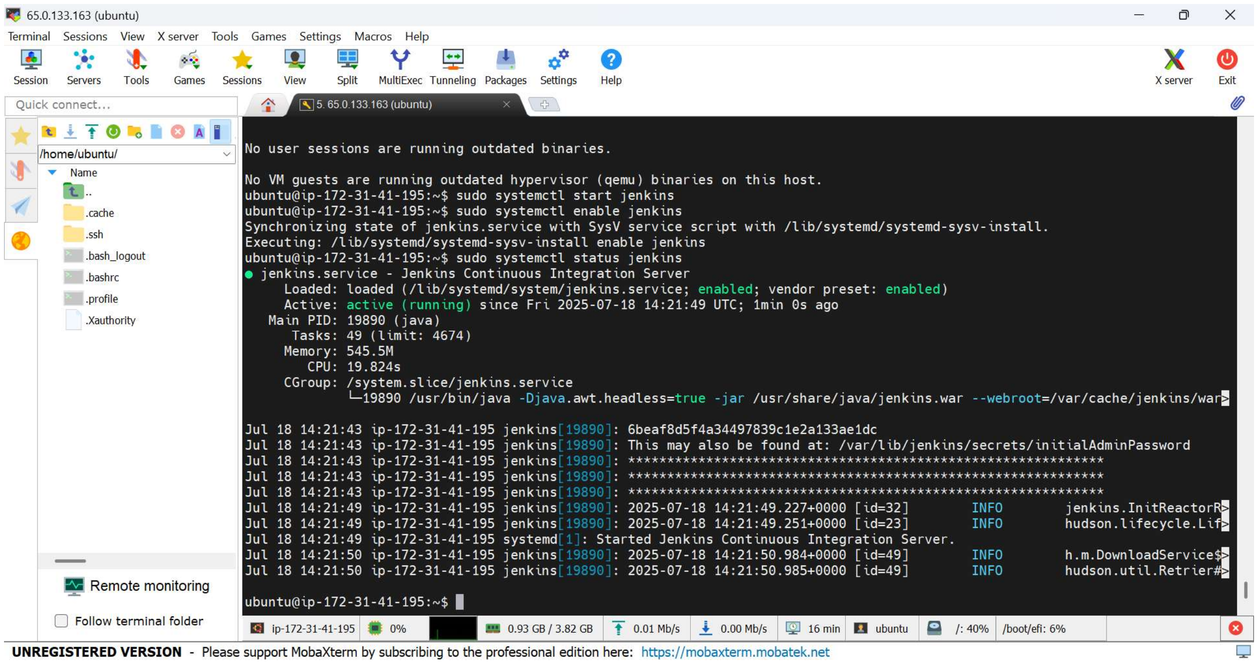
Instance ID	Public IPv4 address	Private IPv4 addresses
i-042d024d57bb2235f	65.0.133.163 | open address	172.31.41.195

```
# 1. Update your system
sudo apt update && sudo apt upgrade -y

# 2. Install Java (Jenkins requires Java 11 or 17+)
sudo apt install openjdk-17-jdk -y

# 3. Verify Java installation
java -version

# 4. Add Jenkins GPG key
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null

# 5. Add Jenkins repository to the sources list
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null

# 6. Update package list again
sudo apt update

# 7. Install Jenkins
sudo apt install jenkins -y

# 8. Start Jenkins service
```

sudo systemctl start jenkins

# 9. Enable Jenkins to start on boot
sudo systemctl enable jenkins

# 10. Check Jenkins status (should be active/running)
sudo systemctl status Jenkins



```
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-41-195:~$ sudo systemctl start jenkins
ubuntu@ip-172-31-41-195:~$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable jenkins
ubuntu@ip-172-31-41-195:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2025-07-18 14:21:49 UTC; 1min 0s ago
   Main PID: 19890 (java)
      Tasks: 49 (limit: 4674)
     Memory: 545.5M
        CPU: 19.824s
     CGroup: /system.slice/jenkins.service
             └─19890 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war>

Jul 18 14:21:43 ip-172-31-41-195 jenkins[19890]: 6beaf8d5f4a34497839c1e2a133ae1dc
Jul 18 14:21:43 ip-172-31-41-195 jenkins[19890]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jul 18 14:21:43 ip-172-31-41-195 jenkins[19890]: *************************************************************
Jul 18 14:21:43 ip-172-31-41-195 jenkins[19890]: *************************************************************
Jul 18 14:21:43 ip-172-31-41-195 jenkins[19890]: *************************************************************
Jul 18 14:21:49 ip-172-31-41-195 jenkins[19890]: 2025-07-18 14:21:49.227+0000 [id=32]        INFO        jenkins.InitReactorR>
Jul 18 14:21:49 ip-172-31-41-195 jenkins[19890]: 2025-07-18 14:21:49.251+0000 [id=23]        INFO        hudson.lifecycle.Lif>
Jul 18 14:21:49 ip-172-31-41-195 systemd[1]: Started Jenkins Continuous Integration Server.
Jul 18 14:21:50 ip-172-31-41-195 jenkins[19890]: 2025-07-18 14:21:50.984+0000 [id=49]        INFO        h.m.DownloadService$>
Jul 18 14:21:50 ip-172-31-41-195 jenkins[19890]: 2025-07-18 14:21:50.985+0000 [id=49]        INFO        hudson.util.Retrier#>

ubuntu@ip-172-31-41-195:~$
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

# Allow Jenkins Port (Default: 8080)

sudo ufw allow 8080
sudo ufw reload

# Access Jenkins Web Interface
Open your browser and go to: http://<your-server-ip>:8080

# To get the initial admin password: Use this password to unlock Jenkins on the web interface.
sudo cat /var/lib/jenkins/secrets/initialAdminPassword

Terraform :

Install Terraform on Ubuntu 22.04
Step 1: Update system packages

sudo apt update && sudo apt upgrade -y

Step 2: Install required dependencies

```
sudo apt install -y gnupg software-properties-common curl
```

Step 3: Add the HashiCorp GPG key

```
curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/hashicorp-archive-keyring.gpg
```

Step 4: Add the official HashiCorp Linux repository

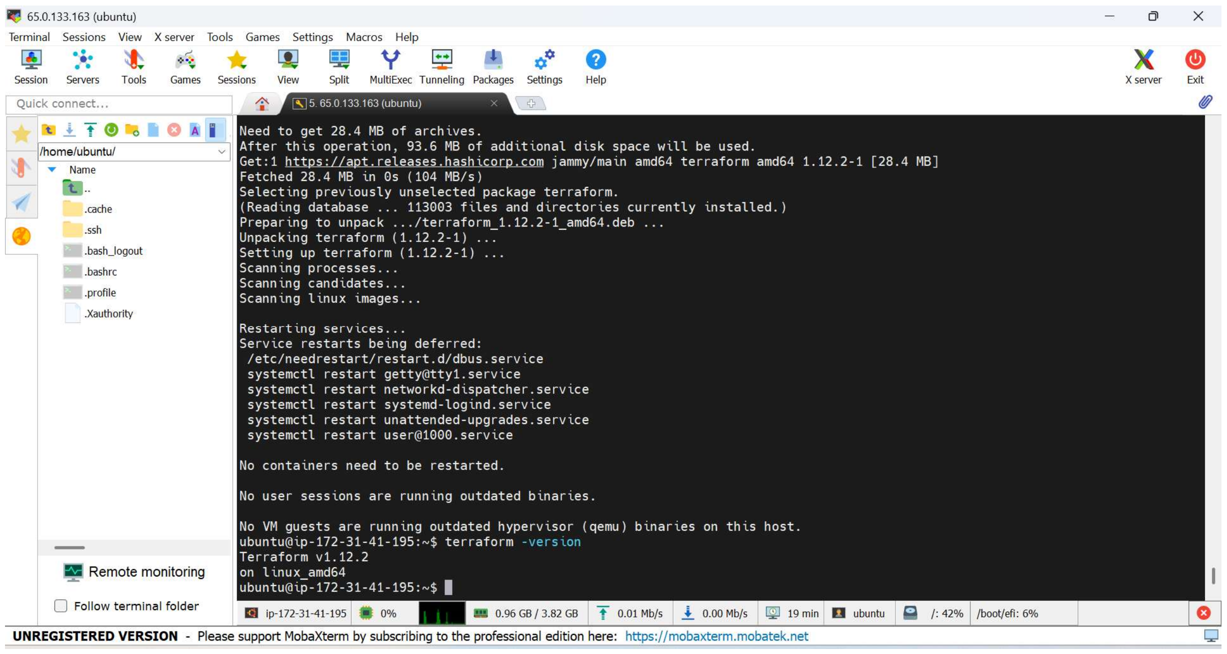
```
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] \
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
sudo tee /etc/apt/sources.list.d/hashicorp.list > /dev/null
```

Step 5: Update and install Terraform

```
sudo apt update
sudo apt install -y terraform
```

Step 6: Verify Terraform installation

```
terraform -version
```


65.0.133.163 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
5. 65.0.133.163 (ubuntu)
/home/ubuntu/
Name
..
.cache
.ssh
.bash_logout
.bashrc
.profile
.Xauthority
Remote monitoring
Follow terminal folder
ip-172-31-41-195 0% 0.96 GB / 3.82 GB 0.01 Mb/s 0.00 Mb/s 19 min ubuntu /: 42% /boot/efi: 6%
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

```
Need to get 28.4 MB of archives.
After this operation, 93.6 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com jammy/main amd64 terraform amd64 1.12.2-1 [28.4 MB]
Fetched 28.4 MB in 0s (104 MB/s)
Selecting previously unselected package terraform.
(Reading database ... 113003 files and directories currently installed.)
Preparing to unpack .../terraform_1.12.2-1_amd64.deb ...
Unpacking terraform (1.12.2-1) ...
Setting up terraform (1.12.2-1) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Restarting services...
Service restarts being deferred:
 /etc/needrestart/restart.d/dbus.service
 systemctl restart getty@tty1.service
 systemctl restart networkd-dispatcher.service
 systemctl restart systemd-logind.service
 systemctl restart unattended-upgrades.service
 systemctl restart user@1000.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-41-195:~$ terraform -version
Terraform v1.12.2
on linux_amd64
ubuntu@ip-172-31-41-195:~$
```

Make sure this IAM user has:

- AmazonEC2FullAccess
- AmazonS3FullAccess
- IAMFullAccess
- CloudWatchLogsFullAccess
- AmazonVPCFullAccess

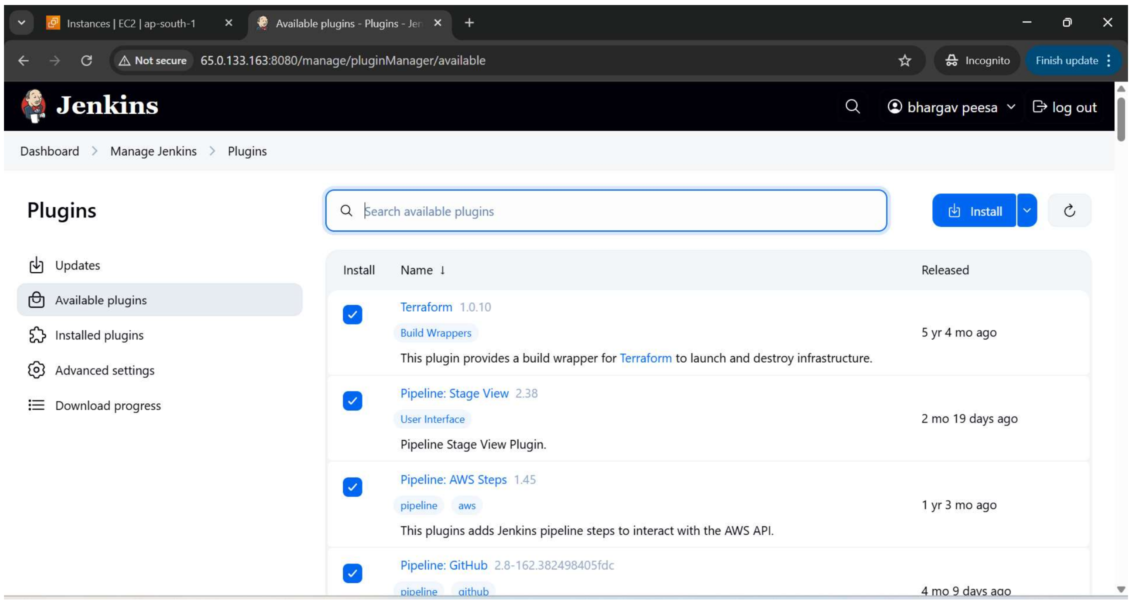
Access Jenkins in Browser:

- http://<your-ip>:8080

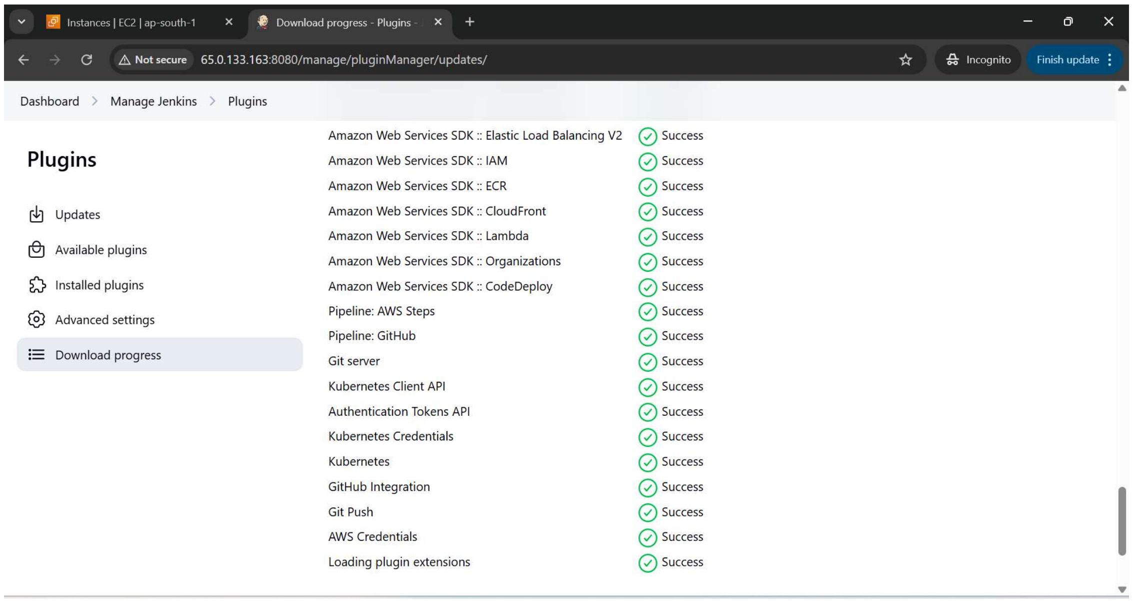
To Execute the Jenkins pipeline that handles Terraform-based infrastructure automation on **AWS**, including **blue/green deployment**, **delta updates**, and **dynamic agents**.

Before creating the Jenkins pipeline, make sure you've set up:

1. Jenkins & Terraform installed on EC2.
2. Jenkins has the following plugins installed:
   - Terraform
   - Git
   - Pipeline
   - AWS Credentials
   - Kubernetes plugin (for dynamic agents, optional)



Install	Name ↓	Released
✓	Terraform 1.0.10 Build Wrappers This plugin provides a build wrapper for Terraform to launch and destroy infrastructure.	5 yr 4 mo ago
✓	Pipeline: Stage View 2.38 User Interface Pipeline Stage View Plugin.	2 mo 19 days ago
✓	Pipeline: AWS Steps 1.45 pipeline aws This plugins adds Jenkins pipeline steps to interact with the AWS API.	1 yr 3 mo ago
✓	Pipeline: GitHub 2.8-162.382498405fdc pipeline github	4 mo 9 days ago

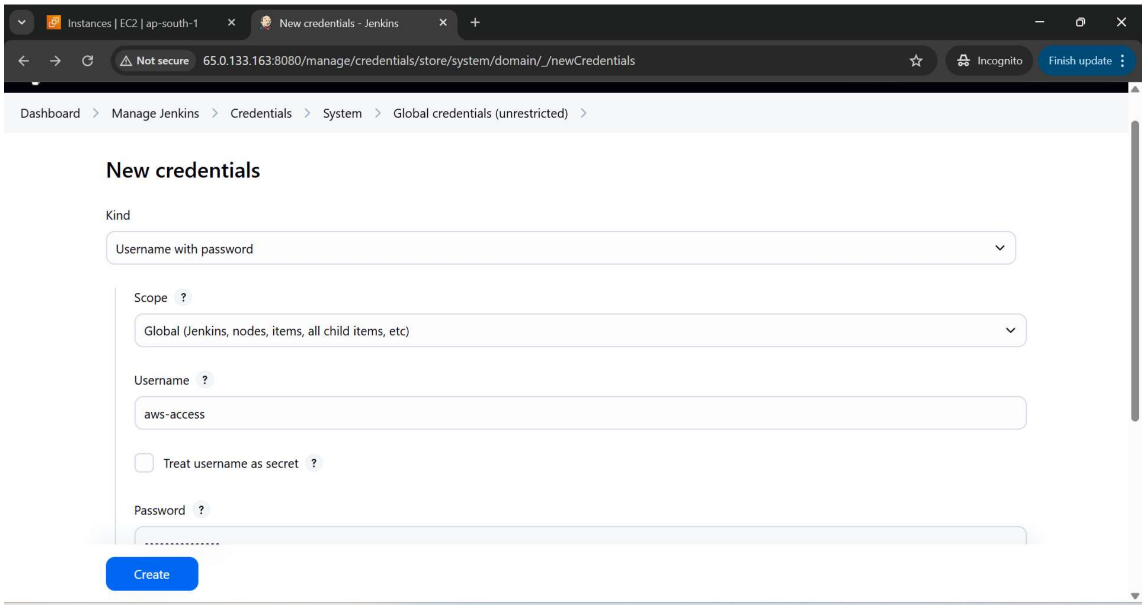

Instances | EC2 | ap-south-1
Download progress - Plugins
Not secure
65.0.133.163:8080/manage/pluginManager/updates/
Incognito
Finish update
Dashboard > Manage Jenkins > Plugins
Plugins
Updates
Available plugins
Installed plugins
Advanced settings
Download progress

Amazon Web Services SDK :: Elastic Load Balancing V2	✓ Success
Amazon Web Services SDK :: IAM	✓ Success
Amazon Web Services SDK :: ECR	✓ Success
Amazon Web Services SDK :: CloudFront	✓ Success
Amazon Web Services SDK :: Lambda	✓ Success
Amazon Web Services SDK :: Organizations	✓ Success
Amazon Web Services SDK :: CodeDeploy	✓ Success
Pipeline: AWS Steps	✓ Success
Pipeline: GitHub	✓ Success
Git server	✓ Success
Kubernetes Client API	✓ Success
Authentication Tokens API	✓ Success
Kubernetes Credentials	✓ Success
Kubernetes	✓ Success
GitHub Integration	✓ Success
Git Push	✓ Success
AWS Credentials	✓ Success
Loading plugin extensions	✓ Success

Adding AWS Credentials to Jenkins

1. Go to Jenkins → **Manage Jenkins → Credentials**
2. Choose **(global)** → **Add Credentials**
3. Type: **AWS Credentials**
4. ID: aws-access-key-id and aws-secret-access-key
5. Jenkinsfile will automatically inject them.

Configure AWS & GitHub Credentials in Jenkins.


Instances | EC2 | ap-south-1
New credentials - Jenkins
Not secure
65.0.133.163:8080/manage/credentials/store/system/domain/_/newCredentials
Incognito
Finish update
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)
Create

New credentials

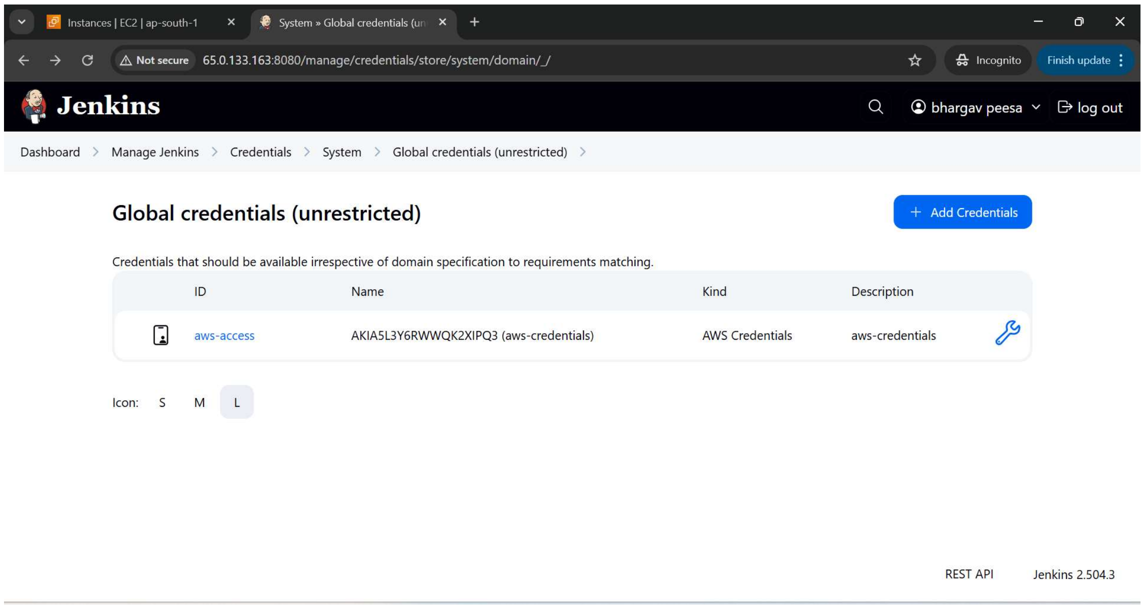
Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: aws-access

Treat username as secret: ☐

Password: ••••••••••••••


Instances | EC2 | ap-south-1
System » Global credentials (un
Not secure
65.0.133.163:8080/manage/credentials/store/system/domain/_/
Incognito
Finish update
Jenkins
bhargav peesa
log out
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)
Add Credentials
Icon: S M L
REST API
Jenkins 2.504.3

Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
aws-access	AKIA5L3Y6RWWQK2XIPQ3 (aws-credentials)	AWS Credentials	aws-credentials

JENKINS PIPELINE CREATION & EXECUTION

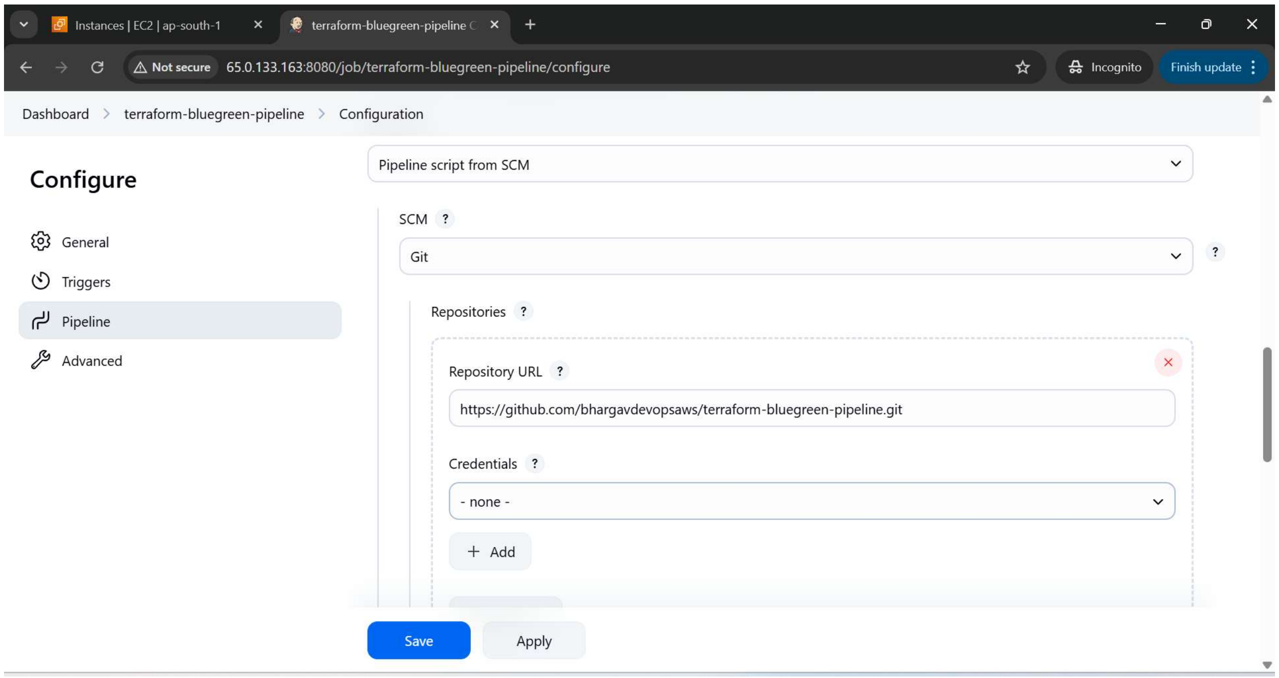
➢ Create a Jenkins Pipeline Job

1. Go to **Jenkins Dashboard → New Item**
2. Name: bluegreen-deployment-pipeline
3. Type: **Pipeline**
4. Click OK

➢ Configure Jenkins Job

In "Pipeline" Section:

- Choose **"Pipeline script from SCM"**
- SCM: Git
- Repo URL: https://github.com/your-org/terraform-bluegreen-pipeline.git
- Branch: */main
- Script Path: Jenkinsfile
- Click **Save**


Instances | EC2 | ap-south-1
terraform-bluegreen-pipeline C
Not secure 65.0.133.163:8080/job/terraform-bluegreen-pipeline/configure
Incognito Finish update
Dashboard > terraform-bluegreen-pipeline > Configuration

Configure

General
Triggers
Pipeline
Advanced

Pipeline script from SCM

SCM ?
Git

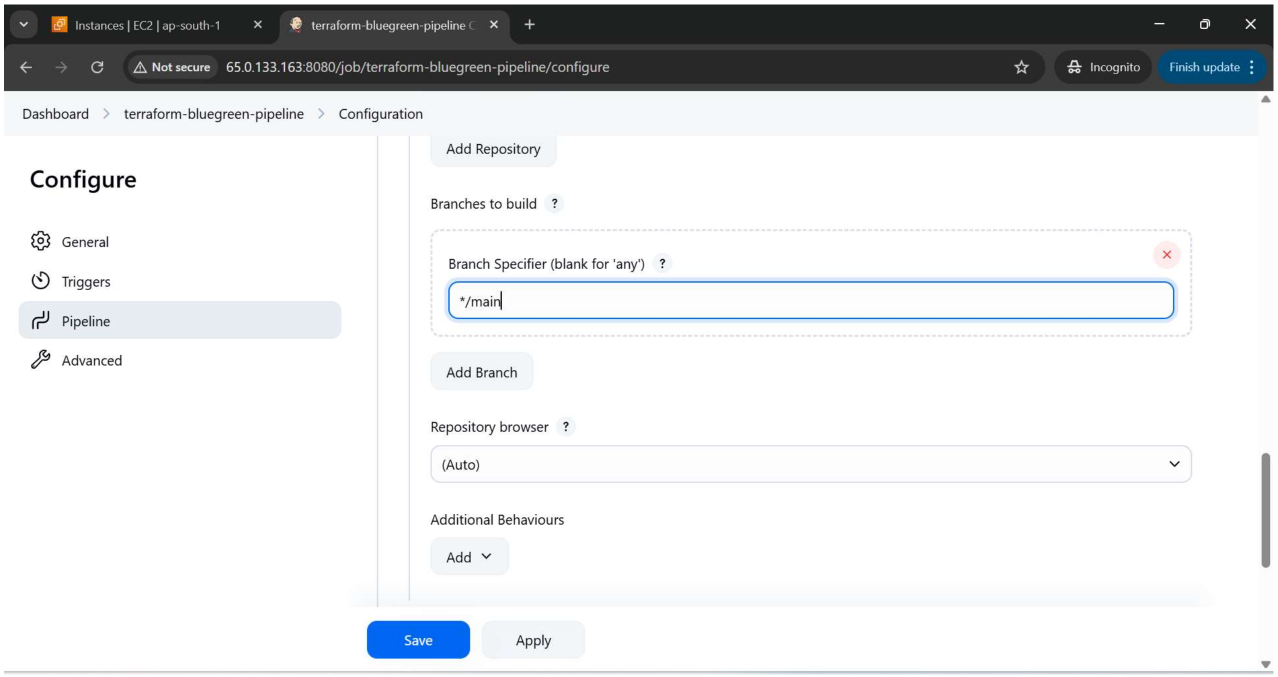
Repositories ?

Repository URL ?
https://github.com/bhargavdevopsaws/terraform-bluegreen-pipeline.git

Credentials ?
- none -

Add

Save Apply


Instances | EC2 | ap-south-1
terraform-bluegreen-pipeline C
Not secure 65.0.133.163:8080/job/terraform-bluegreen-pipeline/configure
Incognito Finish update
Dashboard > terraform-bluegreen-pipeline > Configuration

Configure

General
Triggers
Pipeline
Advanced

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?
*/main

Add Branch

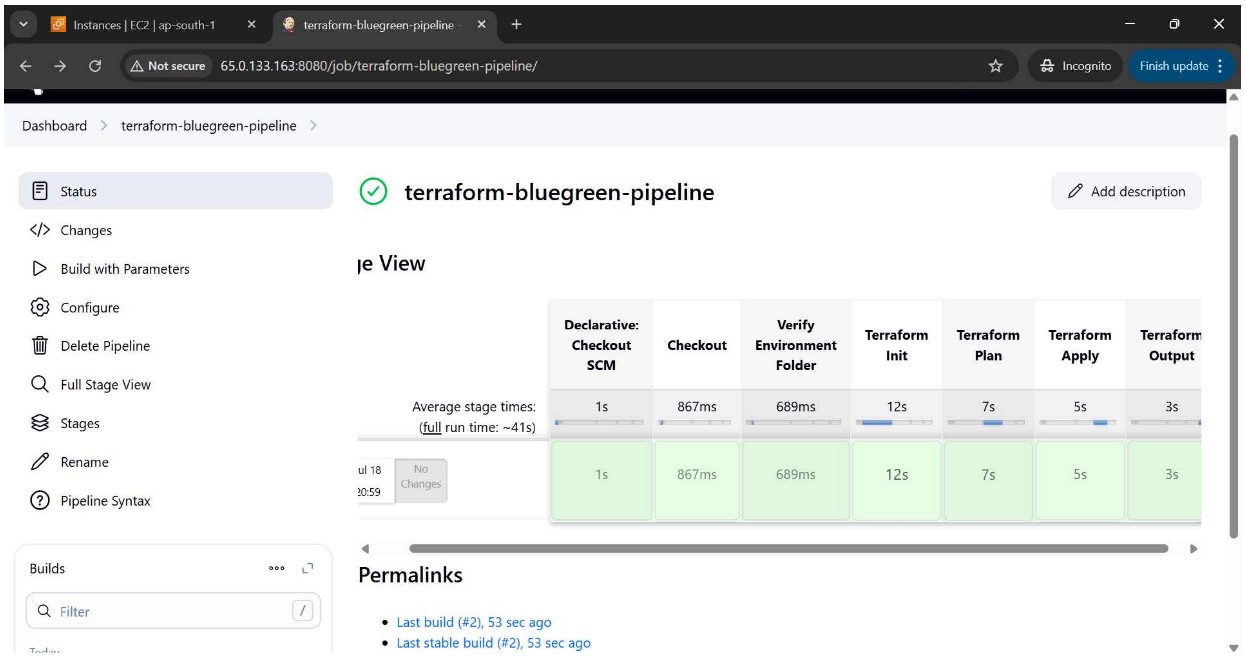
Repository browser ?
(Auto)

Additional Behaviours
Add

Save Apply

➢ Execute Blue Deployment

1. Commit code to GitHub with environment = "blue" in terraform.tfvars & Jenkinsfile.
2. Trigger Jenkins job → Jenkins runs pipeline → Deploys blue infrastructure.

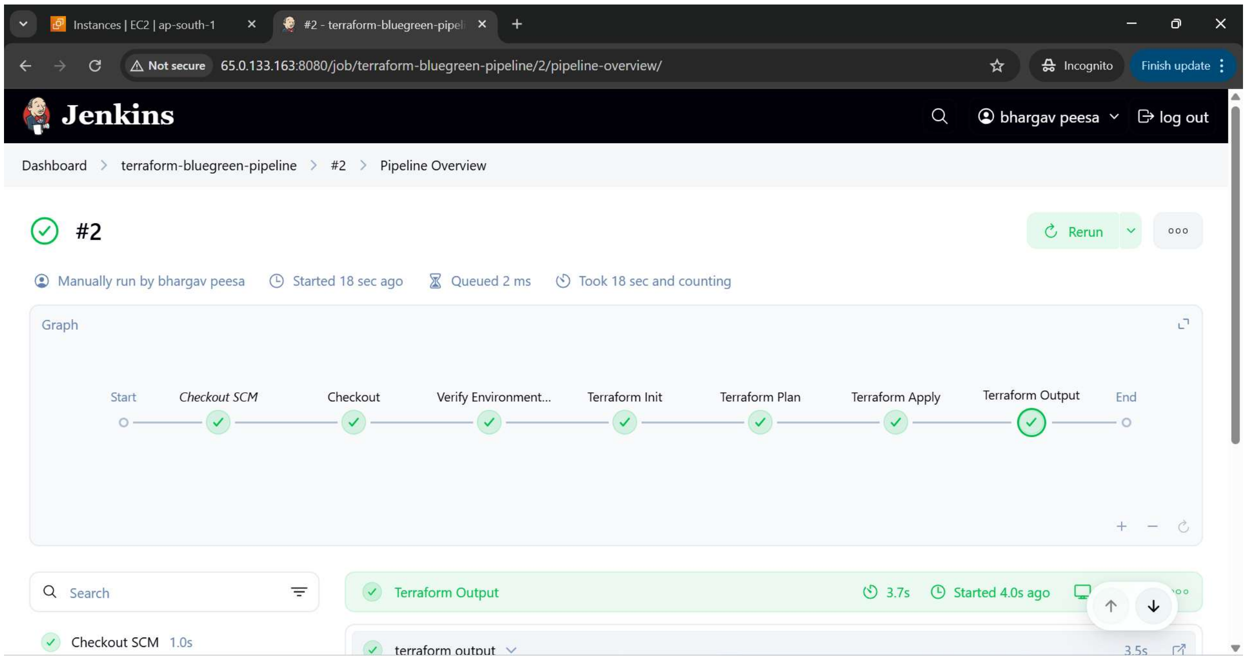


Instances | EC2 | ap-south-1
terraform-bluegreen-pipeline
Not secure
65.0.133.163:8080/job/terraform-bluegreen-pipeline/
Incognito
Finish update
Dashboard > terraform-bluegreen-pipeline >
Status
Changes
Build with Parameters
Configure
Delete Pipeline
Full Stage View
Stages
Rename
Pipeline Syntax
Builds
Filter
terraform-bluegreen-pipeline
Add description
ge View

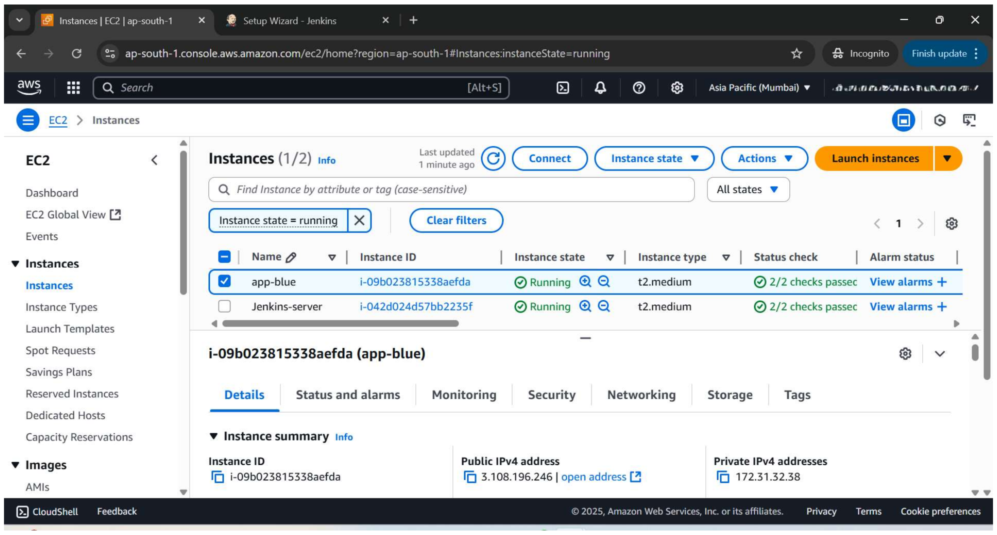
	Declarative: Checkout SCM	Checkout	Verify Environment Folder	Terraform Init	Terraform Plan	Terraform Apply	Terraform Output
Average stage times: (full run time: ~41s)	1s	867ms	689ms	12s	7s	5s	3s
ul 18 20:59 No Changes	1s	867ms	689ms	12s	7s	5s	3s

Permalinks

- Last build (#2), 53 sec ago
- Last stable build (#2), 53 sec ago



The Infrastructure created after the blue deployment.



Instances | EC2 | ap-south-1 — Setup Wizard - Jenkins

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:instanceState=running

EC2 > Instances

Instances (1/2) Info — Last updated 1 minute ago — Connect — Instance state — Actions — Launch instances

Find Instance by attribute or tag (case-sensitive) — All states

Instance state = running — Clear filters

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
app-blue	i-09b023815338aefda	Running	t2.medium	2/2 checks passed	View alarms +
Jenkins-server	i-042d024d57bb2235f	Running	t2.medium	2/2 checks passed	View alarms +

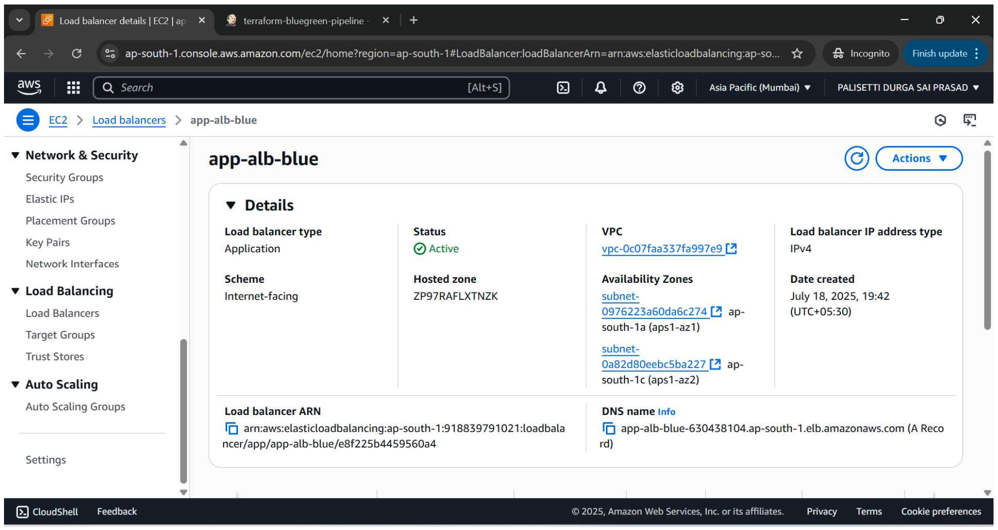
i-09b023815338aefda (app-blue)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-09b023815338aefda	3.108.196.246 | open address	172.31.32.38

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



Load balancer details | EC2 | ap — terraform-bluegreen-pipeline

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:ap-so...

Asia Pacific (Mumbai) — PALISETTI DURGA SAI PRASAD

EC2 > Load balancers > app-alb-blue

app-alb-blue

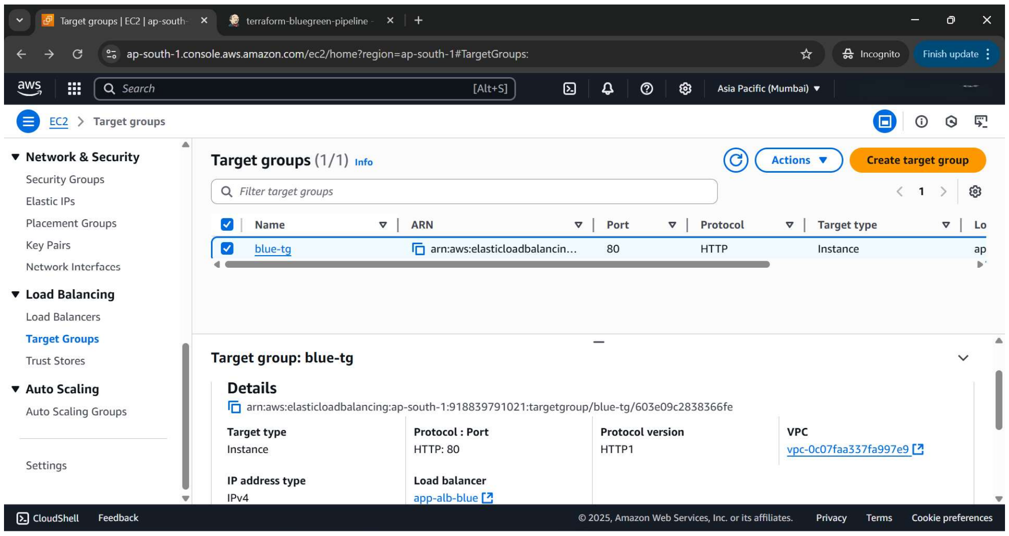
Details

Load balancer type	Status	VPC	Load balancer IP address type
Application	Active	vpc-0c07faa337fa997e9	IPv4

Scheme	Hosted zone	Availability Zones	Date created
Internet-facing	ZP97RAFLXTNZK	subnet-0976223a60da6c274 ap-south-1a (aps1-az1); subnet-0a82d80eebc5ba227 ap-south-1c (aps1-az2)	July 18, 2025, 19:42 (UTC+05:30)

Load balancer ARN: arn:aws:elasticloadbalancing:ap-south-1:918839791021:loadbalancer/app/app-alb-blue/e8f225b4459560a4

DNS name Info: app-alb-blue-630438104.ap-south-1.elb.amazonaws.com (A Record)



Target groups | EC2 | ap-south — terraform-bluegreen-pipeline

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#TargetGroups:

EC2 > Target groups

Target groups (1/1) Info — Actions — Create target group

Filter target groups

Name	ARN	Port	Protocol	Target type	Lo
blue-tg	arn:aws:elasticloadbalancin...	80	HTTP	Instance	ap

Target group: blue-tg

Details

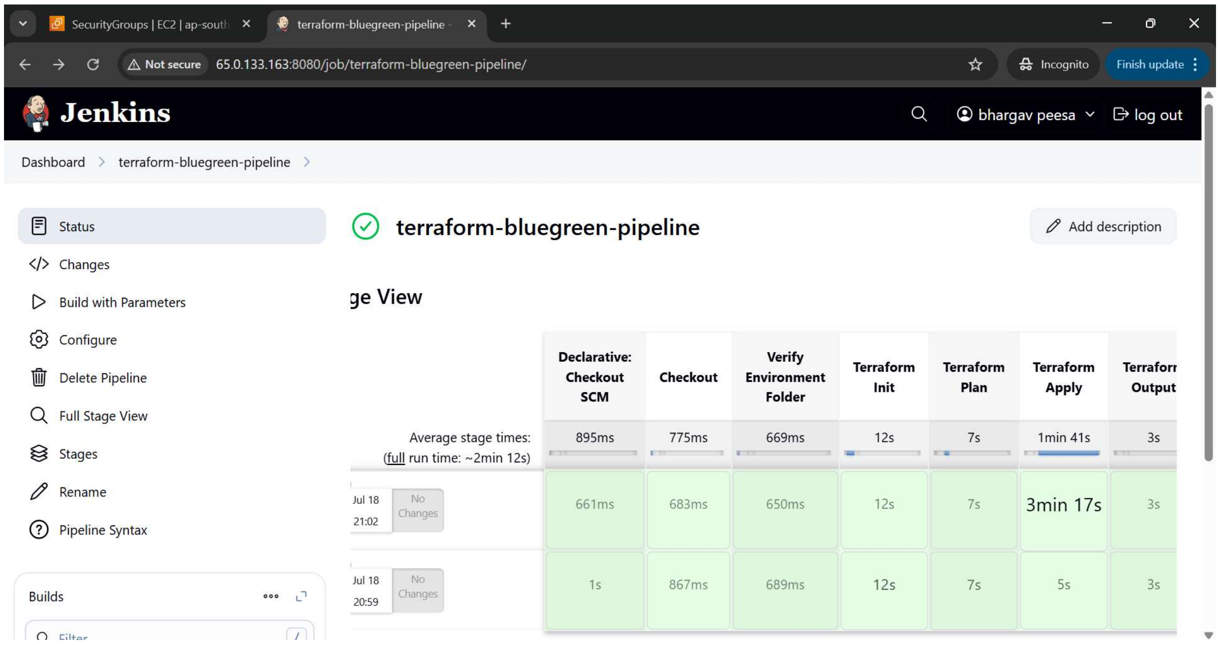
arn:aws:elasticloadbalancing:ap-south-1:918839791021:targetgroup/blue-tg/603e09c2838366fe

Target type	Protocol : Port	Protocol version	VPC
Instance	HTTP: 80	HTTP1	vpc-0c07faa337fa997e9

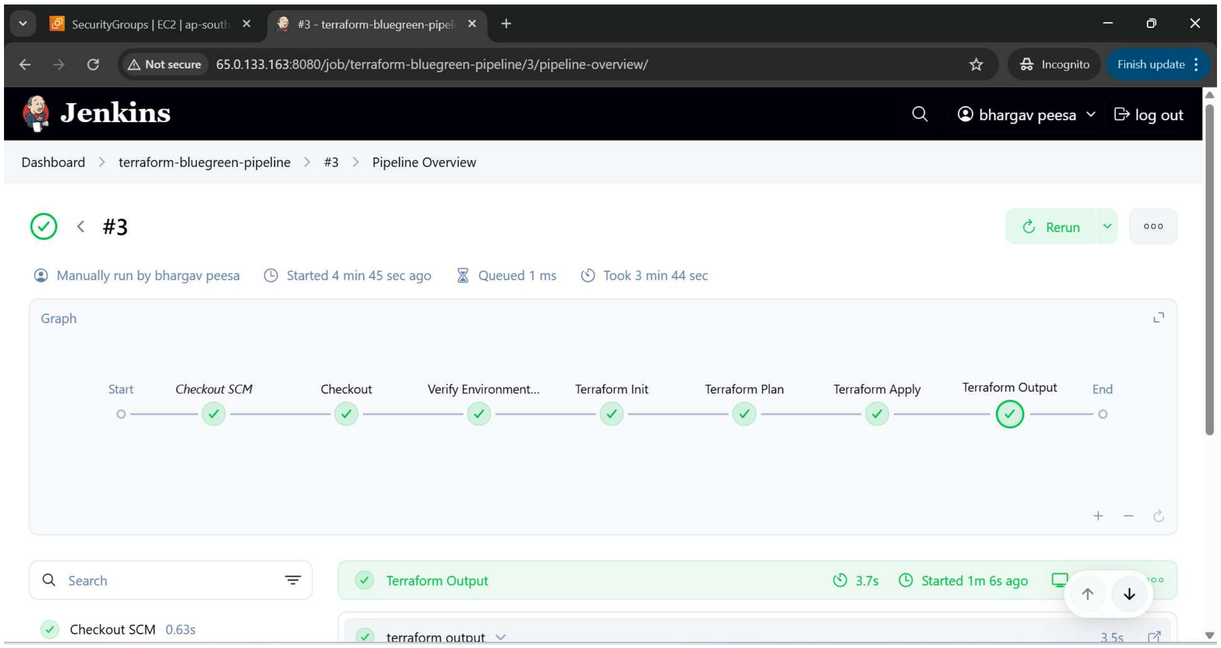
IP address type	Load balancer
IPv4	app-alb-blue

➢ Execute Green Deployment

1. Update terraform.tfvars with environment = "green"
2. Trigger the same Jenkins pipeline job.
3. Jenkins deploys **green** infrastructure (usually with ASG/Launch Template)
4. Listener rules direct traffic based on blue/green weights.



	Declarative: Checkout SCM	Checkout	Verify Environment Folder	Terraform Init	Terraform Plan	Terraform Apply	Terraform Output
Average stage times: (full run time: ~2min 12s)	895ms	775ms	669ms	12s	7s	1min 41s	3s
Jul 18 21:02 No Changes	661ms	683ms	650ms	12s	7s	3min 17s	3s
Jul 18 20:59 No Changes	1s	867ms	689ms	12s	7s	5s	3s



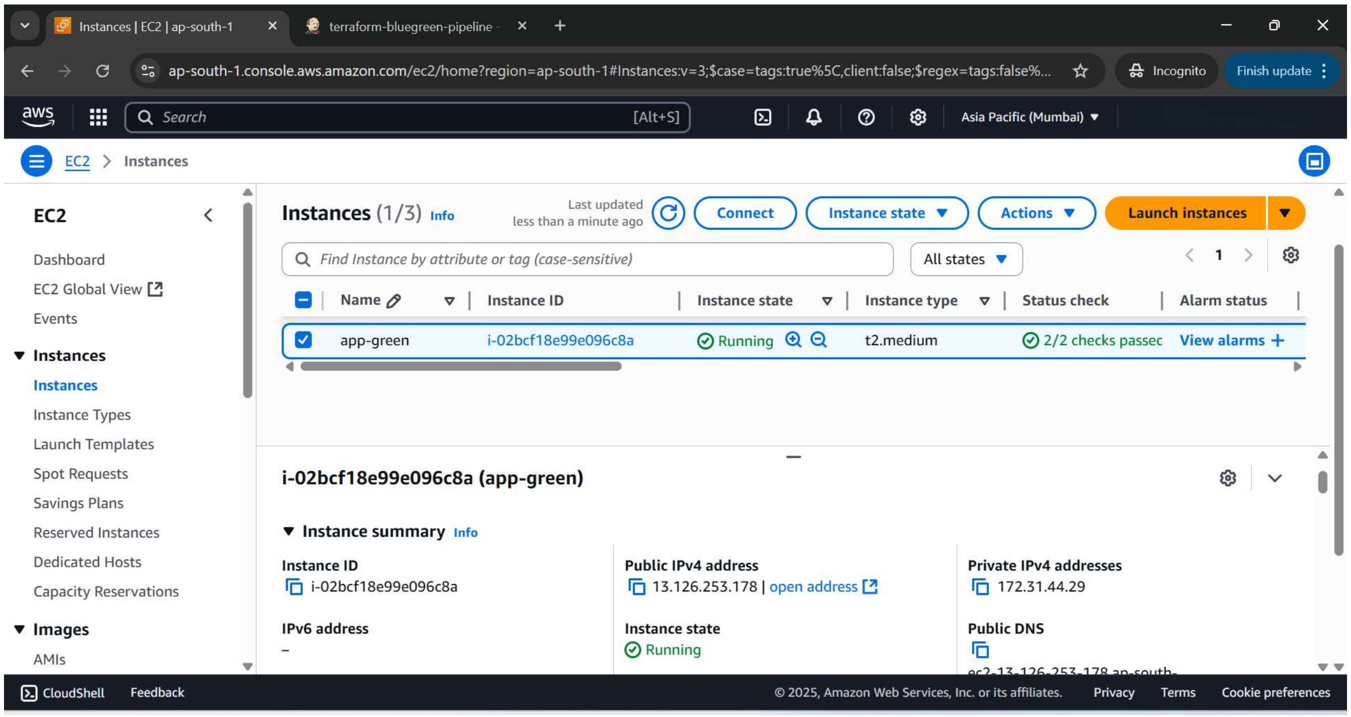
Manually run by bhargav peesa · Started 4 min 45 sec ago · Queued 1 ms · Took 3 min 44 sec

Checkout SCM 0.63s

Terraform Output 3.7s Started 1m 6s ago

terraform output 3.5s

The Infrastructure created after the **green** deployment.



ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%...

EC2 > Instances

Instances (1/3)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
app-green	i-02bcf18e99e096c8a	Running	t2.medium	2/2 checks passec	View alarms +

i-02bcf18e99e096c8a (app-green)

Instance summary

Instance ID: i-02bcf18e99e096c8a

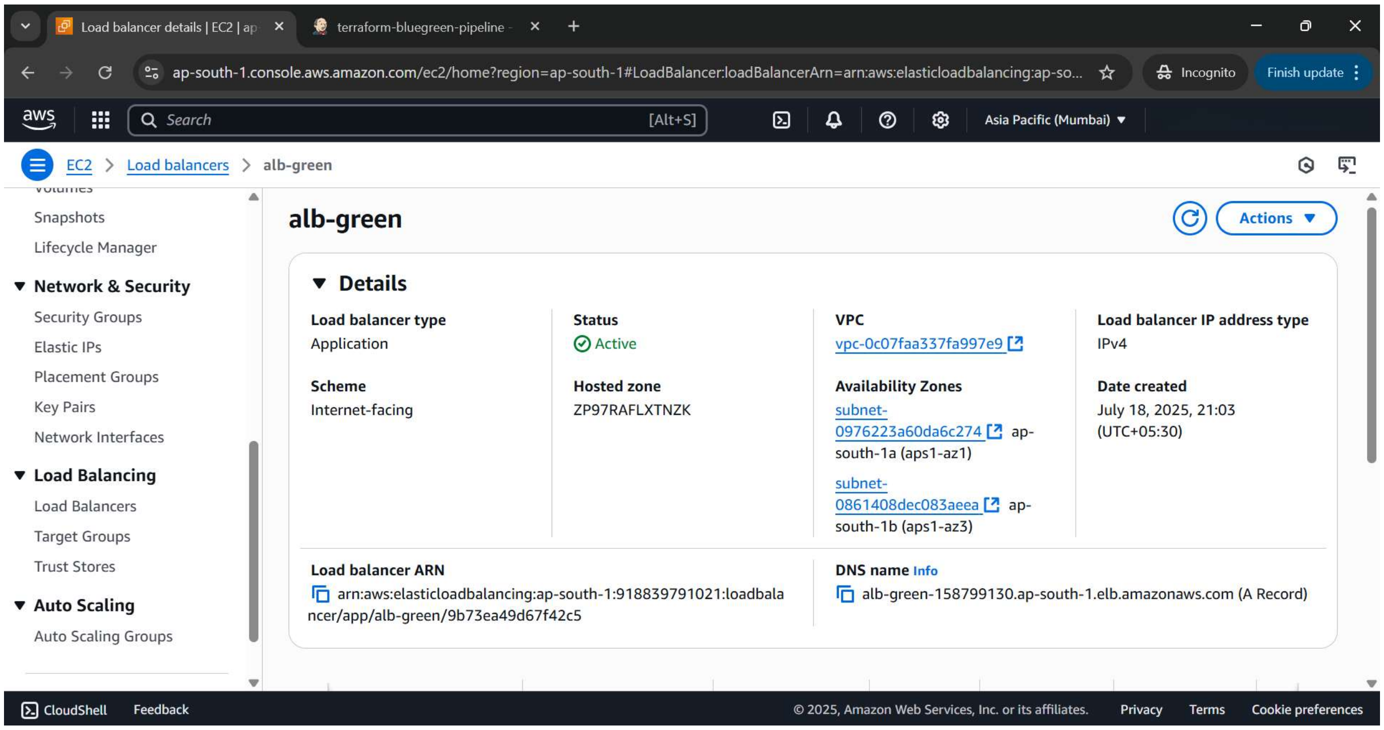
Public IPv4 address: 13.126.253.178 | open address

Private IPv4 addresses: 172.31.44.29

IPv6 address: –

Instance state: Running

Public DNS: ec2-13-126-253-178.ap-south-



ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:ap-so...

EC2 > Load balancers > alb-green

alb-green

Details

Load balancer type: Application

Status: Active

VPC: vpc-0c07faa337fa997e9

Load balancer IP address type: IPv4

Scheme: Internet-facing

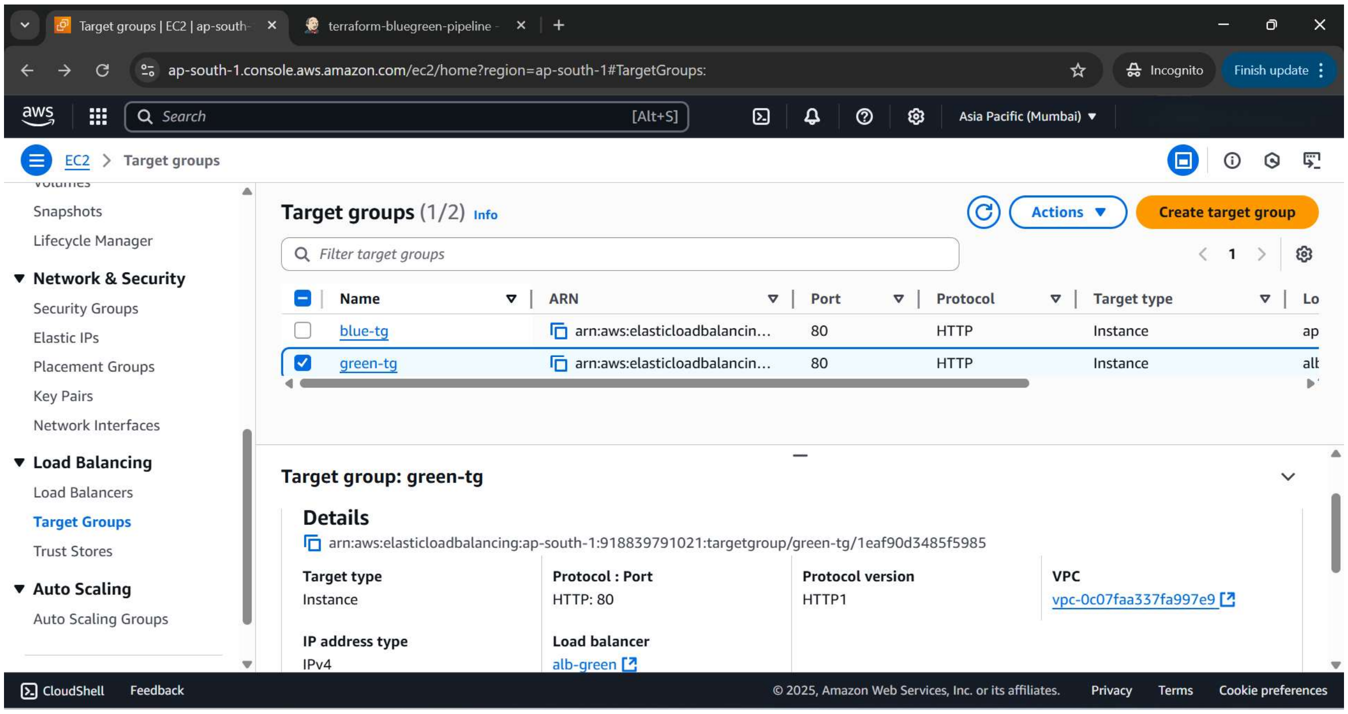
Hosted zone: ZP97RAFLXTNZK

Availability Zones: subnet-0976223a60da6c274 ap-south-1a (aps1-az1); subnet-0861408dec083aeea ap-south-1b (aps1-az3)

Date created: July 18, 2025, 21:03 (UTC+05:30)

Load balancer ARN: arn:aws:elasticloadbalancing:ap-south-1:918839791021:loadbalancer/app/alb-green/9b73ea49d67f42c5

DNS name: alb-green-158799130.ap-south-1.elb.amazonaws.com (A Record)


Target groups | EC2 | ap-south- | terraform-bluegreen-pipeline

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#TargetGroups:

EC2 > Target groups

Target groups (1/2) Info

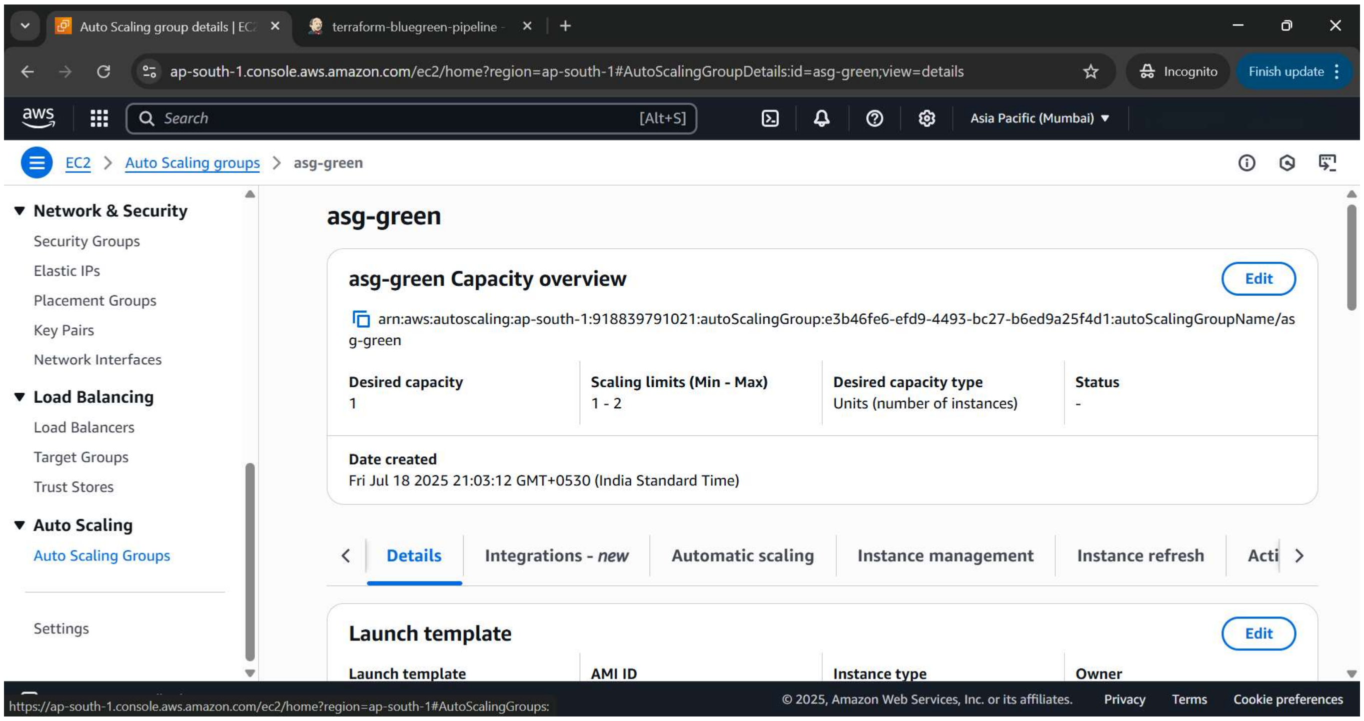
Name	ARN	Port	Protocol	Target type	Lo
blue-tg	arn:aws:elasticloadbalancin…	80	HTTP	Instance	ap
green-tg	arn:aws:elasticloadbalancin…	80	HTTP	Instance	alb

Target group: green-tg

Details

arn:aws:elasticloadbalancing:ap-south-1:918839791021:targetgroup/green-tg/1eaf90d3485f5985

Target type	Protocol : Port	Protocol version	VPC
Instance	HTTP: 80	HTTP1	vpc-0c07faa337fa997e9
IP address type	**Load balancer**		
IPv4	alb-green		


Auto Scaling group details | EC2 | terraform-bluegreen-pipeline

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#AutoScalingGroupDetails:id=asg-green;view=details

EC2 > Auto Scaling groups > asg-green

asg-green

asg-green Capacity overview

arn:aws:autoscaling:ap-south-1:918839791021:autoScalingGroup:e3b46fe6-efd9-4493-bc27-b6ed9a25f4d1:autoScalingGroupName/asg-green

Desired capacity	Scaling limits (Min - Max)	Desired capacity type	Status
1	1 - 2	Units (number of instances)	-

Date created
Fri Jul 18 2025 21:03:12 GMT+0530 (India Standard Time)

Launch template

Launch template | AMI ID | Instance type | Owner

Summary:

- Built a complete CI/CD pipeline for Terraform-based infrastructure.
- Integrated Jenkins and GitHub.
- Parameterized blue/green environments.
- Successfully deployed both environments on AWS.

Benefits:

- Zero-downtime deployment.
- Safe rollback using ALB traffic shifting.
- Infrastructure as Code (IaC) with automation.