

# CI/CD Pipeline Infrastructure with Terraform, Ansible, Jenkins, and GitHub

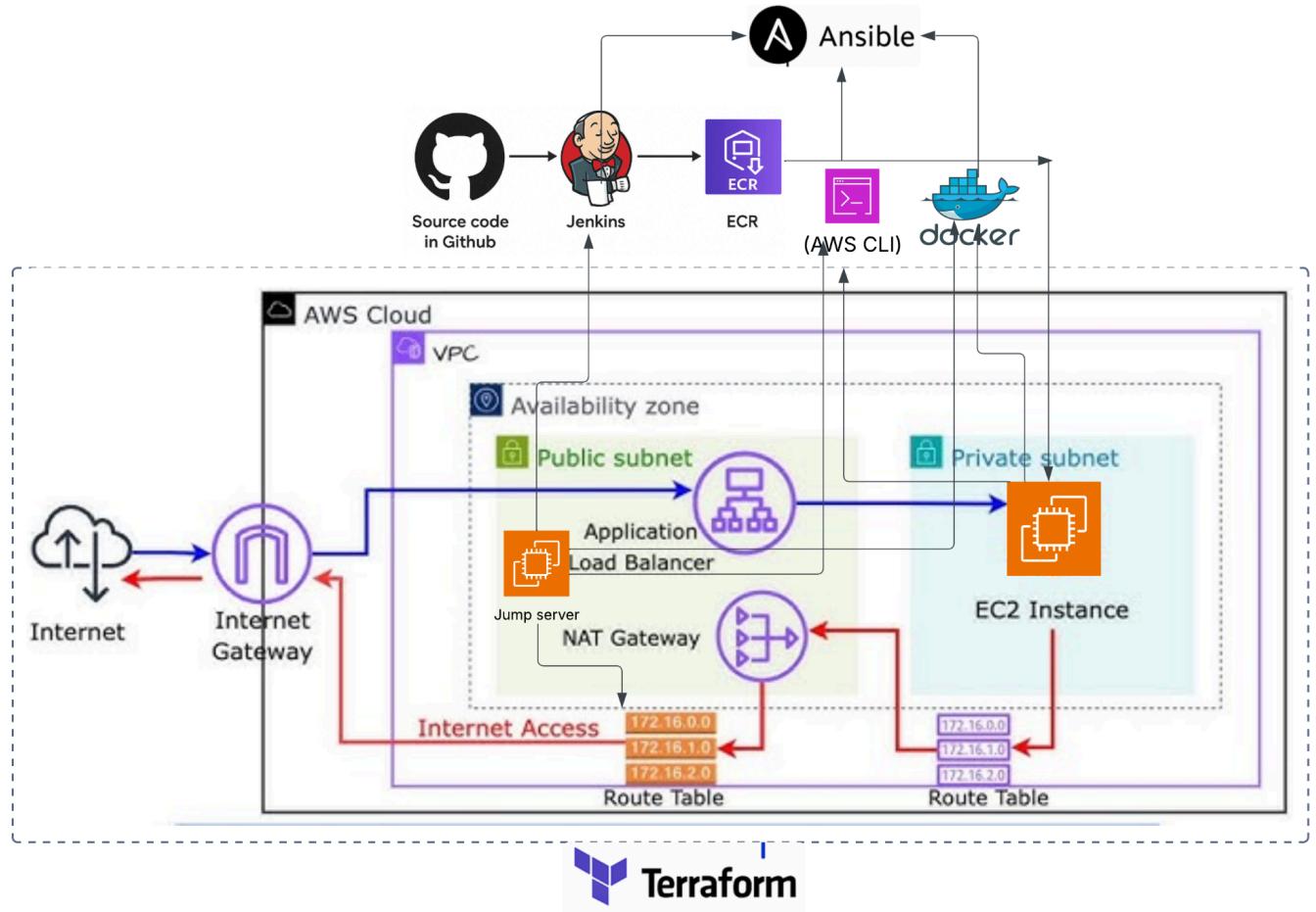
## 1. Infrastructure Provisioning with Terraform

```
terraform/
└── modules/
    ├── alb/
    ├── ec2/
    ├── ecr/
    └── vpc/
    ├── backend.tf
    ├── main.tf
    ├── variables.tf
    └── terraform.tfvars
```

### Key Components:

- **VPC:** Created a VPC with public and private subnets.
- **EC2 Instances:**
  - **Jump Server** (public): Hosts Jenkins and serves as a bastion.
  - **Private Server:** Hosts the application container.
- **NAT Gateway:** Enables the private instance to access the internet.
- **ALB:** Application Load Balancer targets the private server (on port **3005**).
- **ECR:** A Docker image repository created and managed via Terraform.
- **Terraform Backend:**
  - Remote backend configured with an **S3 bucket** to store the **terraform.tfstate**.
  - Enabled **state locking** with a DynamoDB table or S3 versioning.

Screenshot of the Amazon S3 console showing the contents of the 'state' folder in the 'moshitbucket0079' bucket. The folder contains a single file named 'terraform.tfstate' with a size of 71.1 KB and a storage class of Standard. The file was last modified on May 31, 2025, at 21:35:47 (UTC+05:30).



### ● Modules Used:

- `ec2` – for creating jump (public) and private EC2 instances.

- **alb** – for Application Load Balancer targeting the private instance.
- **ecr** – for Elastic Container Registry to store Docker images.
- **vpc** – for networking setup including subnets, NAT, and internet gateway.

```

✓ TERRAFORM-ANSIBLE
  ✓ ansibleterraform
    ! create_user.yml
    ! install_jenkins_docker_withpass.yml
    ! install_jenkins_docker.yml
    ! install_nginx.yml
    ≡ inventory_old.ini
    ≡ inventory.ini
    ! old_install_jenkins_docker_withpass_w-o-x-terra.yml
  ✓ terraform
    > .terraform
    ✓ modules
      > alb
      > ec2
      > ecr
      > vpc
      ⚭ .gitignore
      ≡ .terraform.lock.hcl
      🎨 backend.tf
      ≡ jenkins_output.txt
      🎨 main.tf
      🎨 outputs.tf
      🎨 terraform.tfvars
      🎨 variables.tf
    ✓ terraformbucket
      > .terraform
      ✓ modules/s3_bucket
        🎨 main.tf
        🎨 outputs.tf
        🎨 variables.tf
        ⚭ .gitignore
        ≡ .terraform.lock.hcl
        🎨 main.tf
        🎨 output.tf
        ≡ terraform.tfstate
        ≡ terraform.tfstate.backup
      ⚭ .gitignore
    ⚭ README.md

```

- **Security:**

- SSH key pair is used to access instances securely.
- Instances are appropriately tagged and placed in isolated subnets.

**EC2 > Instances**

**Instances (1/1) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Mo
private-server	i-0c6ba7f12789eed53	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-	-	-	-	dis
jump-server	i-04aeef492a4f918af7	Running	t3.medium	3/3 checks passed	View alarms +	us-east-1a	-	35.175.148.218	-	-	dis

**i-0c6ba7f12789eed53 (private-server)**

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

**Instance summary**

Instance ID	i-0c6ba7f12789eed53
IPV6 address	-
Hostname type	IP name: ip-10-0-2-42.ec2.internal
Answer private resource DNS name	-

**Public IPv4 address**

Private IP4 address	10.0.2.42
Public DNS	-
Elastic IP addresses	-

**Load balancers**

**Load balancers (1/1)**

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
app-alb	app-alb-364511489.us-east...	Active	vpc-0acbffffd1d3b4145	2 Availability Zones	application	May 31, 2025, 21:23 (UTC+05:30)

**Load balancer: app-alb**

**Details** Listeners and rules Network mapping Resource map Security Monitoring Integrations Attributes Capacity Tags

**Details**

Load balancer type	Application	Status	Active
Scheme	Internet-facing	Hosted zone	Z35SXDOTRQ7X7K
Load balancer ARN	arn:aws:elasticloadbalancing:us-east-1:058264275893:loadbalancer/app/app-alb/6c66e04a3b1b0e28	VPC	vpc-0acbffffd1d3b4145
		Availability Zones	subnet-025216360d911a9e4 us-east-1a (use1-az2) subnet-0041da67de8b9ba20 us-east-1b (use1-az4)
		Load balancer IP address type	IPv4
		Date created	May 31, 2025, 21:23 (UTC+05:30)
		DNS name info	app-alb-364511489.us-east-1.elb.amazonaws.com (A Record)

**Amazon ECR > Private registry > Repositories > myapplication**

**Images (1)**

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Last recorded pull time
1	Image	31 May 2025, 22:26:04 (UTC+05:5)	400.14	<a href="#">Copy URI</a>	<a href="#">sha256:212bbe11fb64bcb862bd77d14826...</a>	31 May 2025, 22:26:10 (UTC+05:5)

## 2. Configuration Management with Ansible

Ansible was used to automate the provisioning and configuration of the two EC2 instances:

### ✨ Tasks performed:

- **Jump Server (Public):**

- Installed Docker
- Installed Jenkins
- Installed AWS CLI
- Configured Jenkins to run with Docker privileges

- **Private Server:**

- Installed Docker
- Installed AWS CLI
- Added `ubuntu` user to Docker group

Both servers were configured without any manual SSH logins, maintaining true infrastructure-as-code and configuration-as-code principles.

```
null_resource.run_ansible_playbook (local-exec): TASK [Add ubuntu user to Docker group] ****
null_resource.run_ansible_playbook: Still creating... [11m20s elapsed]
null_resource.run_ansible_playbook (local-exec): changed: [private]

null_resource.run_ansible_playbook (local-exec): TASK [Restart Docker to apply group change] ****
null_resource.run_ansible_playbook: Still creating... [11m30s elapsed]
null_resource.run_ansible_playbook (local-exec): changed: [private]

null_resource.run_ansible_playbook (local-exec): PLAY RECAP ****
null_resource.run_ansible_playbook (local-exec): jump                  : ok=20   changed=14   unreachable=0    failed=0    skipped=0   rescued=0   ignored=0
null_resource.run_ansible_playbook (local-exec): private              : ok=10   changed=6    unreachable=0    failed=0    skipped=0   rescued=0   ignored=0

null_resource.run_ansible_playbook: Creation complete after 11m36s [id=4151363412226839792]
data.local_file.jenkins_password: Reading...
data.local_file.jenkins_password: Read complete after 0s [id=095519d6c0f3ccdcc4220fcdea0831031cf2bdff]
Releasing state lock. This may take a few moments...

Apply complete! Resources: 30 added, 0 changed, 0 destroyed.

Outputs:
app.alb_dns = "app-alb-364511489.us-east-1.elb.amazonaws.com"
jenkins_admin_password = <<EOT
e3fa2c2dac0846a7b6184439dbe914a4

EOT
jump_server_public_ip = "35.175.148.218"
private_server_private_ip = "10.0.2.42"
ac190-mohit@ac190-mohit-Latitude-5410:~/Documents/terraform$
```

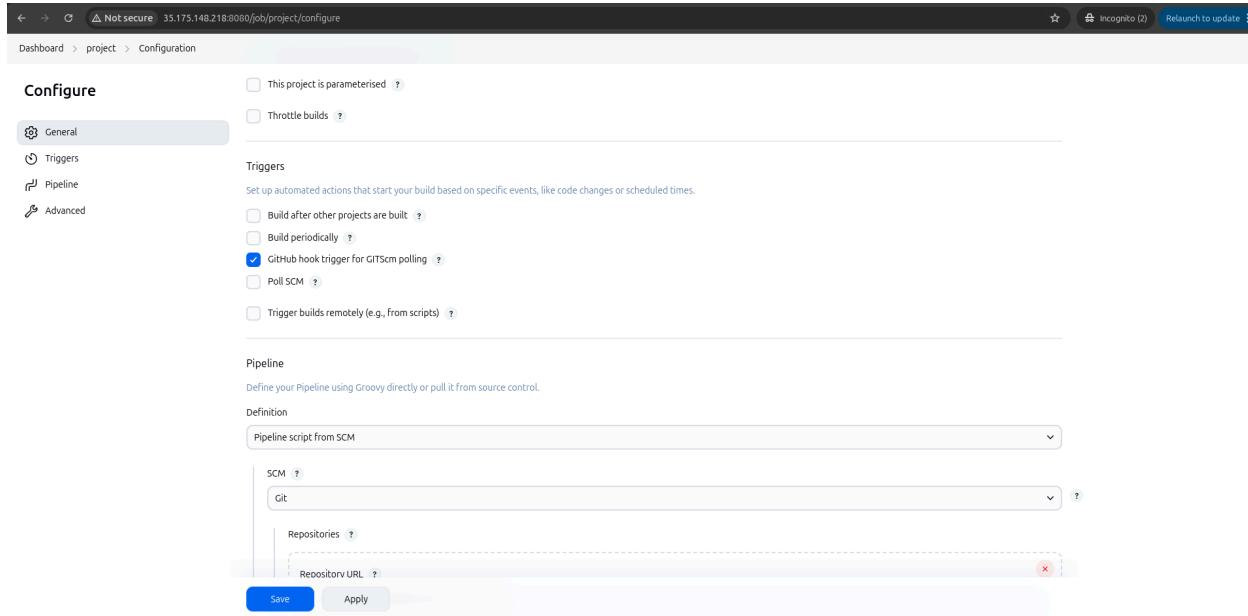
## 3. CI/CD Pipeline with Jenkins and GitHub

## Jenkins Setup:

- Jenkins installed and managed via Ansible on the jump server.
- Plugins installed:
  - GitHub integration plugin
  - Docker plugin
  - ECR plugin
  - AWS CLI plugin (or system AWS CLI manually configured)

## Jenkins Pipeline Configuration:

- Created a new **Pipeline project**.
- Used "**Pipeline script from SCM**" method.
- Connected to a **GitHub repository** (configured with webhook for auto-build).



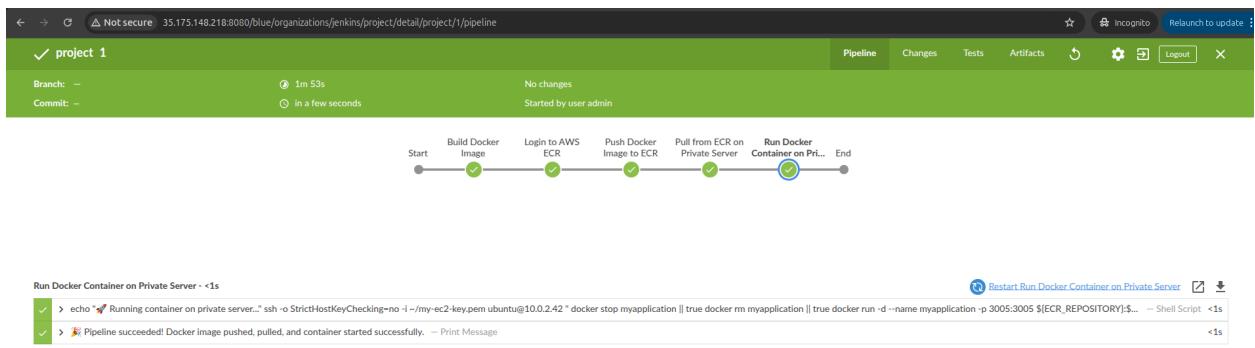
## Credentials Used:

- GitHub credentials for pulling code

- AWS Access Key and Secret Access Key (stored securely in Jenkins Credentials Manager)

## Workflow:

1. Code is pushed to GitHub.
2. Webhook triggers Jenkins job.
3. Jenkins builds Docker image.
4. Image is pushed to ECR.
5. Jenkins deploys the container on the private EC2 instance.



## Networking and Security

- ALB listens on port **3005** and forwards requests to the private EC2.
- Health checks and listener rules configured via Terraform.
- Proper IAM permissions and roles were assigned.
- EC2 instances were launched in **us-east-1** region, complying with SCP restrictions.

- **Credentials:**

- Git credentials for repository access.
- AWS credentials (`aws-ecr-creds`) for ECR push/pull.
- `Secret text` used for:
  - `aws-region-id`
  - `aws-account-id`
  - `private-server-ip`

