Clone the GitHub Repo: https://github.com/Raghavarora09/task-pipeline.git

Run: terraform init

And terraform apply

The following infrastructure will be set up

1. VPC (Virtual Private Cloud)

- **aws_vpc.main**: Creates a VPC with a CIDR block of 10.0.0.0/16, enabling DNS support and hostnames.

2. Internet Gateway

- **aws_internet_gateway.main**: Creates an Internet Gateway and attaches it to the VPC to allow internet access for resources within the VPC.

3. Subnets

- **aws_subnet.public_1 and aws_subnet.public_2**: Creates two public subnets with different CIDR blocks in different availability zones, both configured to assign public IP addresses to instances launched within them.

4. Route Table

- **aws_route_table.public**: Creates a route table that directs all traffic (0.0.0.0/0) to the Internet Gateway.
- **aws_route_table_association.public_1 and aws_route_table_association.public_2**: Associates the public subnets with the route table, allowing them to route traffic through the Internet Gateway.

5. Security Groups

- **aws_security_group.jenkins**: Defines a security group for the Jenkins server, allowing inbound SSH (port 22) and Jenkins (port 8080) traffic from anywhere, and allowing all outbound traffic.
- **aws_security_group.web**: Defines a security group for web servers, allowing inbound HTTP (port 80) and SSH (port 22) traffic from anywhere, and allowing all outbound traffic.

6. IAM Roles and Instance Profiles

- **aws_iam_role.codedeploy_ec2_role**: Creates an IAM role for EC2 instances with permissions for AWS CodeDeploy.
- **aws_iam_role_policy_attachment.codedeploy_ec2_policy**: Attaches the necessary policy to the EC2 role for CodeDeploy.
- **aws_iam_instance_profile.codedeploy_ec2_profile**: Creates an instance profile to attach the IAM role to EC2 instances.

7. Jenkins EC2 Instance

- **aws_instance.jenkins**: Launches an EC2 instance for Jenkins with the specified AMI and instance type (t2.medium). The instance is placed in one of the public subnets and uses the Jenkins security group. A user_data script installs Jenkins, AWS CLI, and other dependencies on the instance.

8. Launch Template for Web Servers

- **aws_launch_template.web**: Defines a launch template for creating web server instances. These instances are configured with an Ubuntu AMI, the web security group, and a user_data script that installs Apache and the CodeDeploy agent.

9. Auto Scaling Group

- **aws_autoscaling_group.web**: Creates an Auto Scaling Group that manages the web servers. It maintains a desired capacity of 2 instances, with a maximum of 5. The instances are distributed across the two public subnets and are associated with the ALB's target group.

10. Application Load Balancer

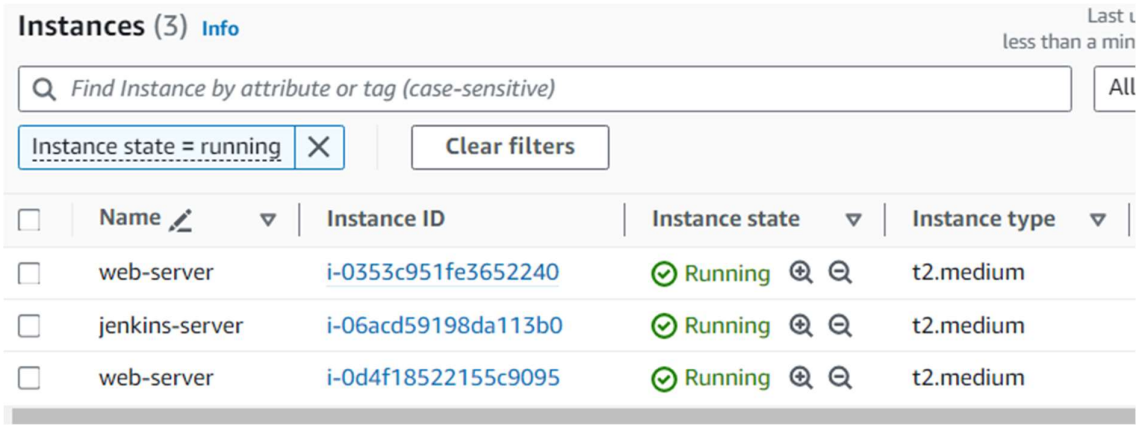
- **aws_lb.web**: Creates an Application Load Balancer (ALB) to distribute traffic across the web servers.
- **aws_lb_target_group.web**: Defines a target group for the ALB, which listens on port 80 and performs health checks on the root path (/) of the web servers.
- **aws_lb_listener.web**: Configures the ALB to listen on port 80 and forward traffic to the target group.

11. CodeDeploy Resources

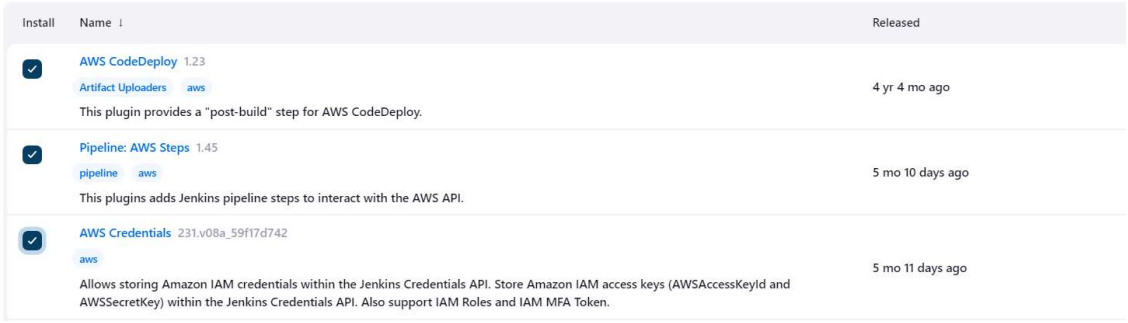
- **aws_codedeploy_app.web**: Defines a CodeDeploy application for deploying the web application.
- **aws_codedeploy_deployment_group.web**: Creates a deployment group within the CodeDeploy application. It uses EC2 tags to identify the instances to deploy to and enables automatic rollback on deployment failure.

12. IAM Role for CodeDeploy

- **aws_iam_role.codedeploy**: Creates an IAM role that allows AWS CodeDeploy to interact with other AWS services.
- **aws_iam_role_policy_attachment.codedeploy_service**: Attaches a managed policy to the CodeDeploy IAM role to grant it the necessary permissions.

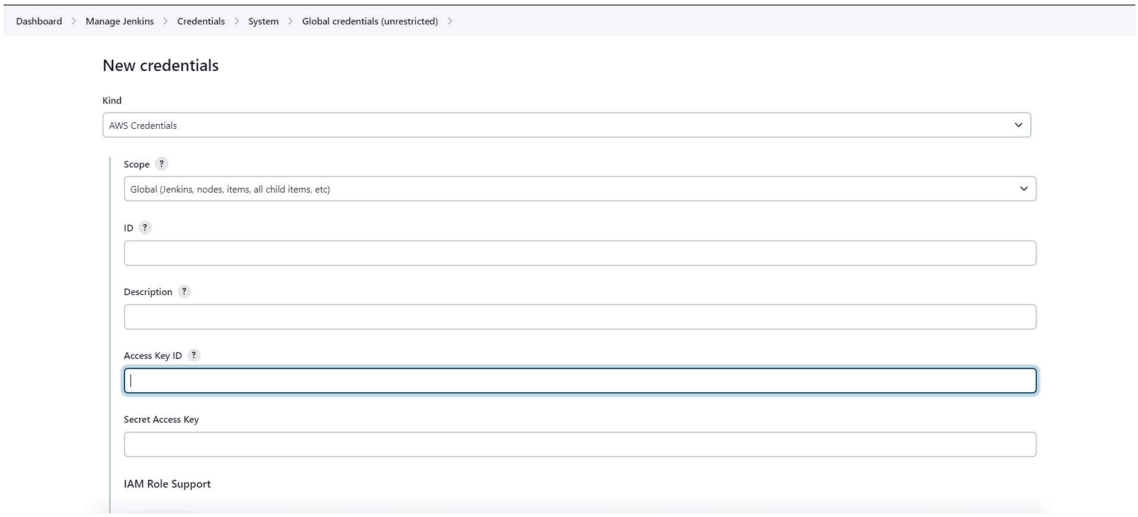


Set up Jenkins using the administrator password and install suggested and the below necessary plugins



And click on restart Jenkins after installation.

Now set up the AWS credentials, the same ones in terraform.tfvars file



Copy the id of the created credentials and edit it in the Jenkinsfile on GitHub

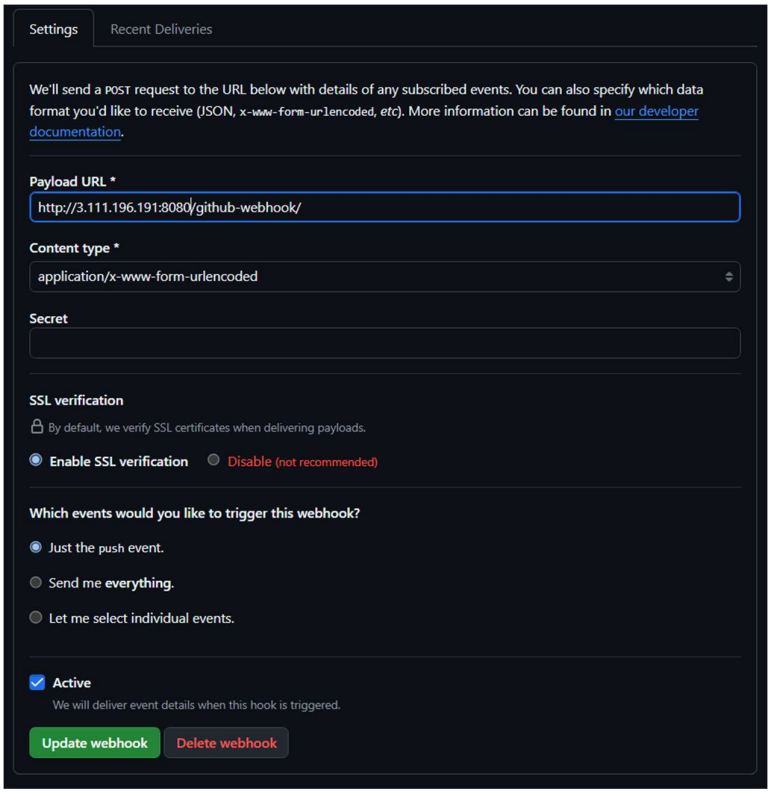
Create a new pipeline style item in Jenkins

Select “GitHub hook trigger for GITScm polling” and in Pipeline select

Pipeline from SCM and give your repository link

Also goto GitHub > Settings > Webhook > New Webhook

<Jenkins_url>/github-webhook/



Now the Pipeline is set up

For every commit on the GitHub a new build will be initiated

And the webpage will be deployed on the webserver instances using AWS CodeBuild