Name- Aarif M

Emp- 2355418

Use Case – Provisioning Infrastructure Design for Multi-Tiered Web Application in AWS

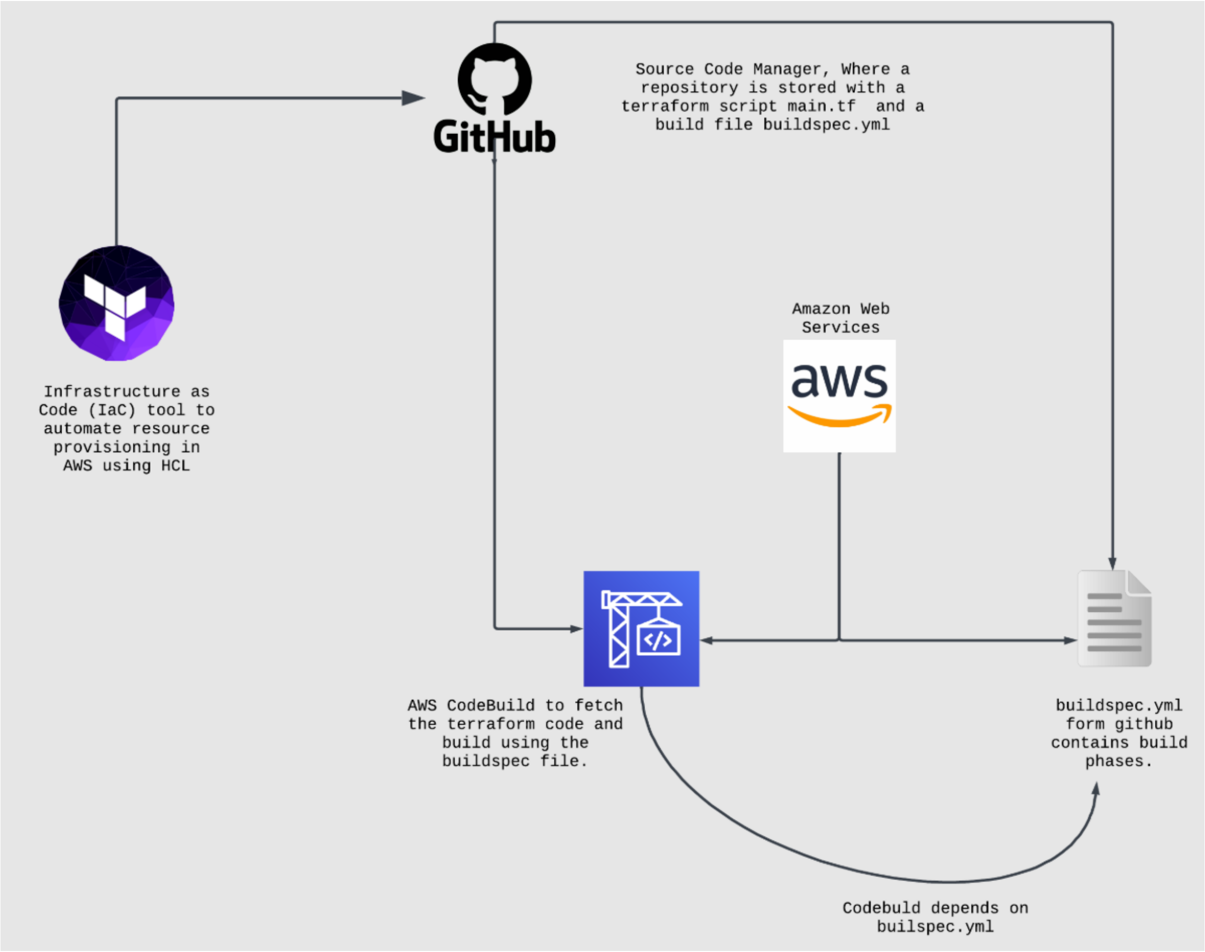
Overview:

The objective is to design a secure, scalable, and highly available infrastructure for a multi-tiered web application in AWS. The solution includes creating the application tiers, security groups, load balancers, an S3 bucket, IAM roles, and provisioning resources using Terraform. Below is the infrastructure design and Terraform configuration breakdown.

Technologies Used:

1) Hashi Corp Terraform
2) GitHub
3) AWS (Amazon Web Services)

Methodology:


GitHub
Source Code Manager, Where a repository is stored with a terraform script main.tf and a build file buildspec.yml
Infrastructure as Code (IaC) tool to automate resource provisioning in AWS using HCL
Amazon Web Services
aws
AWS CodeBuild to fetch the terraform code and build using the buildspec file.
buildspec.yml form github contains build phases.
Codebuld depends on builspec.yml

Name- Aarif M

Emp- 2355418

Terraform template has the following resources,

- Application tiers:
  - ✓ Autoscaling group in a public Subnet with servers running in at least 2 AZ’s.
  - ✓ A single EC2 instance in a private Subnet

- Security group for each subnet
  - ✓ Security group in public subnet should allow only HTTP and ssh traffic.
  - ✓ Attach above SG to the ASG in public Subnet.
  - ✓ Security group in private subnet should allow all traffic from above security group only.
  - ✓ Attach above SG to EC2 instance in Private subnet.

- Load balancer
  - ✓ Application load balancer, Target group – directing traffic to ASG in public subnet.
  - ✓ NLB, Target group – directing traffic to EC2 instance in private subnet.

- S3 bucket
  - ✓ Should not be accessible to public, version enabled.
- IAM role
  - ✓ Role to have full access to the above S3 bucket.
  - ✓ The role should be attached to the Instances in ASG in public subnet.

To set up AWS infrastructure, we start by creating an Autoscaling Group (ASG) in a public subnet, ensuring it spans at least two Availability Zones (AZs) for high availability, and launch a single EC2 instance in a private subnet. Next, creating a security group for the public subnet that allows inbound HTTP (port 80) and SSH (port 22) traffic, and attach it to the ASG. For the private subnet, creating a security group that allows all traffic from the public subnet's security group and attach it to the EC2 instance. Setting up an Application Load Balancer (ALB) with a target group directing traffic to the ASG in the public subnet, and a Network Load Balancer (NLB) with a target group directing traffic to the EC2 instance in the private subnet. Creating an S3 bucket, ensure it is not publicly accessible, and enable versioning. Finally, creating an IAM role with full access to the S3 bucket and attach this role to the instances in the ASG in the public subnet. This setup ensures a secure, scalable, and highly available infrastructure.

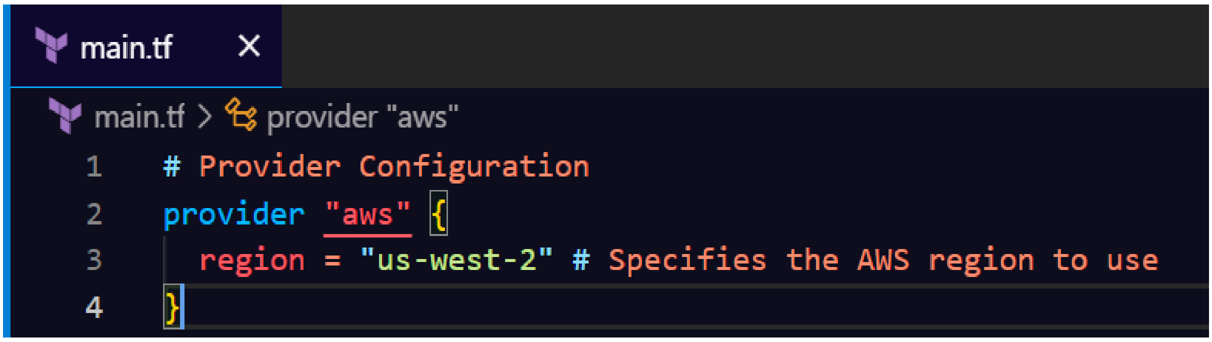
Stage 1 – Terraform:

Name- Aarif M

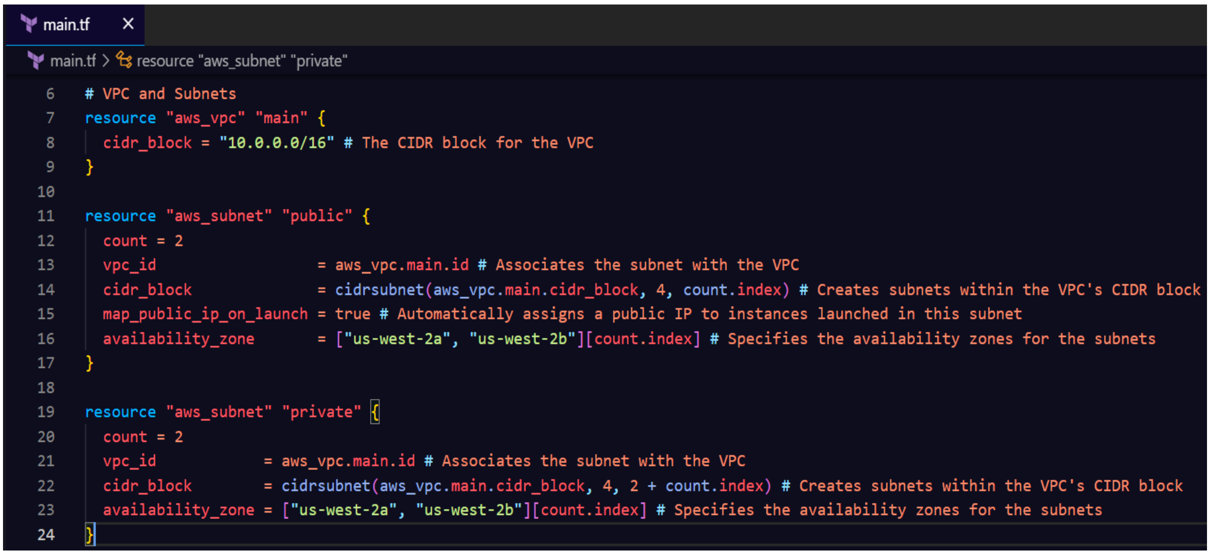
Emp- 2355418

Terraform is an open-source Infrastructure as Code (IaC) tool developed by HashiCorp. It allows you to define and provision infrastructure using a high-level configuration language called HashiCorp Configuration Language (HCL) or JSON.

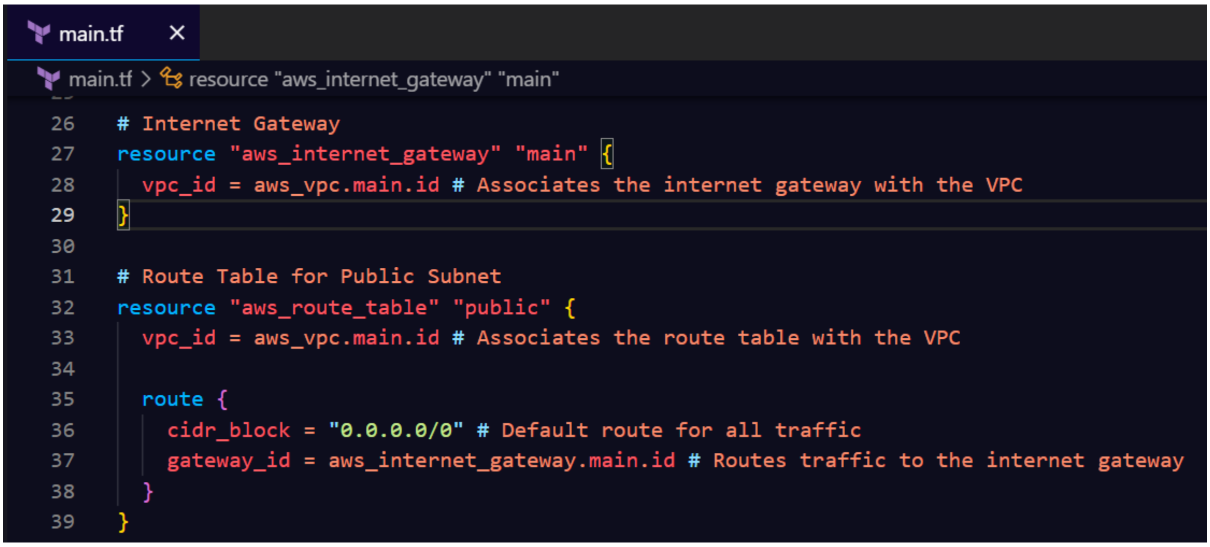
It is stored in GitHub - https://github.com/Aarifmedharsha/Devops1/blob/main/main.tf



```
# Provider Configuration
provider "aws" {
  region = "us-west-2" # Specifies the AWS region to use
}
```



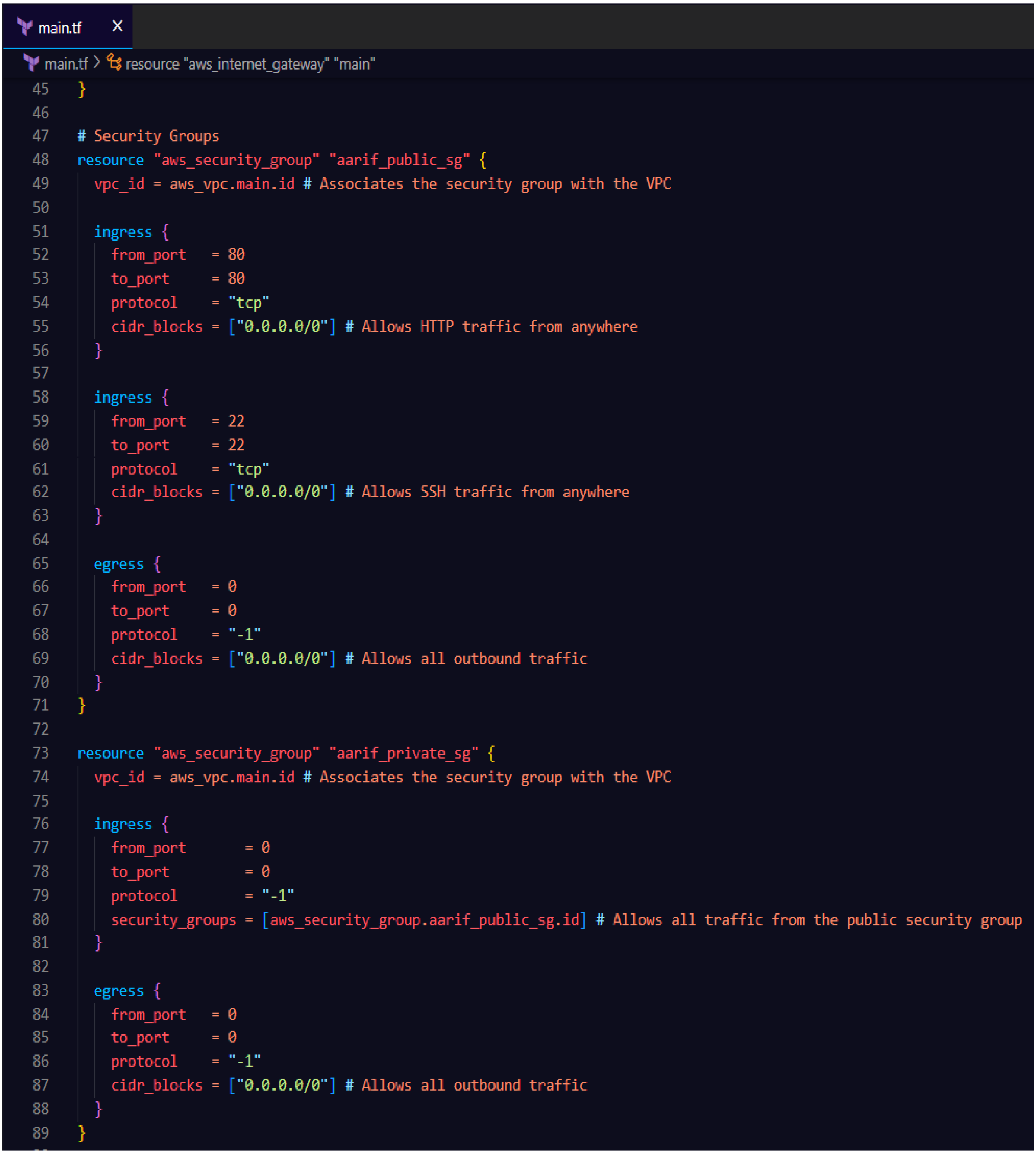
```
# VPC and Subnets
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16" # The CIDR block for the VPC
}

resource "aws_subnet" "public" {
  count = 2
  vpc_id                  = aws_vpc.main.id # Associates the subnet with the VPC
  cidr_block              = cidrsubnet(aws_vpc.main.cidr_block, 4, count.index) # Creates subnets within the VPC's CIDR block
  map_public_ip_on_launch = true # Automatically assigns a public IP to instances launched in this subnet
  availability_zone       = ["us-west-2a", "us-west-2b"][count.index] # Specifies the availability zones for the subnets
}

resource "aws_subnet" "private" {
  count = 2
  vpc_id            = aws_vpc.main.id # Associates the subnet with the VPC
  cidr_block        = cidrsubnet(aws_vpc.main.cidr_block, 4, 2 + count.index) # Creates subnets within the VPC's CIDR block
  availability_zone = ["us-west-2a", "us-west-2b"][count.index] # Specifies the availability zones for the subnets
}
```



```
# Internet Gateway
resource "aws_internet_gateway" "main" {
  vpc_id = aws_vpc.main.id # Associates the internet gateway with the VPC
}

# Route Table for Public Subnet
resource "aws_route_table" "public" {
  vpc_id = aws_vpc.main.id # Associates the route table with the VPC

  route {
    cidr_block = "0.0.0.0/0" # Default route for all traffic
    gateway_id = aws_internet_gateway.main.id # Routes traffic to the internet gateway
  }
}
```

Name- Aarif M

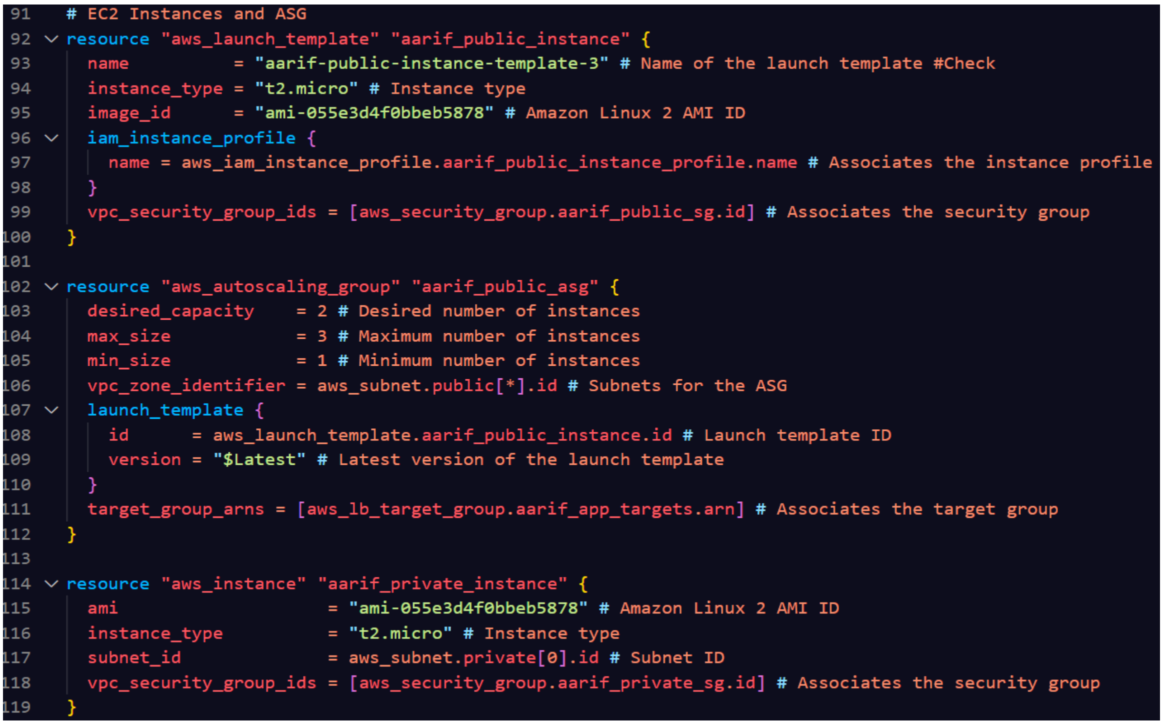
Emp- 2355418



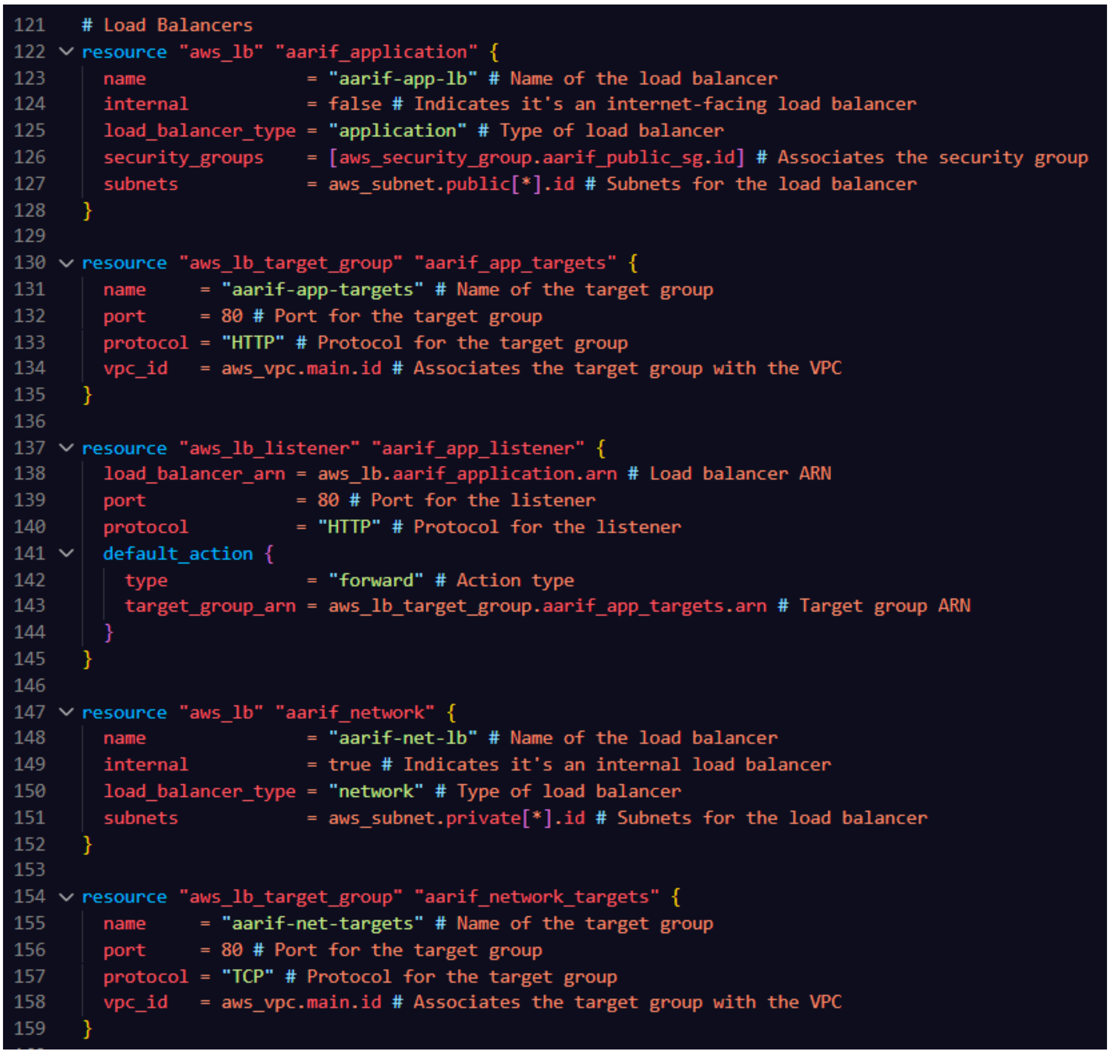
```
}

# Security Groups
resource "aws_security_group" "aarif_public_sg" {
  vpc_id = aws_vpc.main.id # Associates the security group with the VPC

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Allows HTTP traffic from anywhere
  }

  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Allows SSH traffic from anywhere
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"] # Allows all outbound traffic
  }
}

resource "aws_security_group" "aarif_private_sg" {
  vpc_id = aws_vpc.main.id # Associates the security group with the VPC

  ingress {
    from_port       = 0
    to_port         = 0
    protocol        = "-1"
    security_groups = [aws_security_group.aarif_public_sg.id] # Allows all traffic from the public security group
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"] # Allows all outbound traffic
  }
}
```

Name- Aarif M

Emp- 2355418



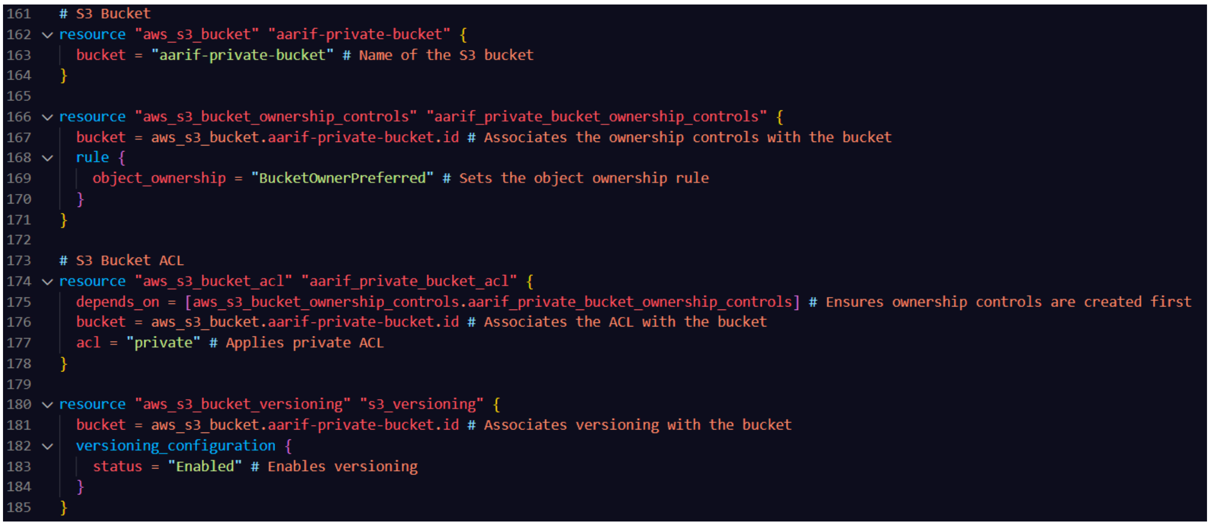
```
# EC2 Instances and ASG
resource "aws_launch_template" "aarif_public_instance" {
  name          = "aarif-public-instance-template-3" # Name of the launch template #Check
  instance_type = "t2.micro" # Instance type
  image_id      = "ami-055e3d4f0bbeb5878" # Amazon Linux 2 AMI ID
  iam_instance_profile {
    name = aws_iam_instance_profile.aarif_public_instance_profile.name # Associates the instance profile
  }
  vpc_security_group_ids = [aws_security_group.aarif_public_sg.id] # Associates the security group
}

resource "aws_autoscaling_group" "aarif_public_asg" {
  desired_capacity    = 2 # Desired number of instances
  max_size            = 3 # Maximum number of instances
  min_size            = 1 # Minimum number of instances
  vpc_zone_identifier = aws_subnet.public[*].id # Subnets for the ASG
  launch_template {
    id      = aws_launch_template.aarif_public_instance.id # Launch template ID
    version = "$Latest" # Latest version of the launch template
  }
  target_group_arns = [aws_lb_target_group.aarif_app_targets.arn] # Associates the target group
}

resource "aws_instance" "aarif_private_instance" {
  ami                    = "ami-055e3d4f0bbeb5878" # Amazon Linux 2 AMI ID
  instance_type          = "t2.micro" # Instance type
  subnet_id              = aws_subnet.private[0].id # Subnet ID
  vpc_security_group_ids = [aws_security_group.aarif_private_sg.id] # Associates the security group
}
```



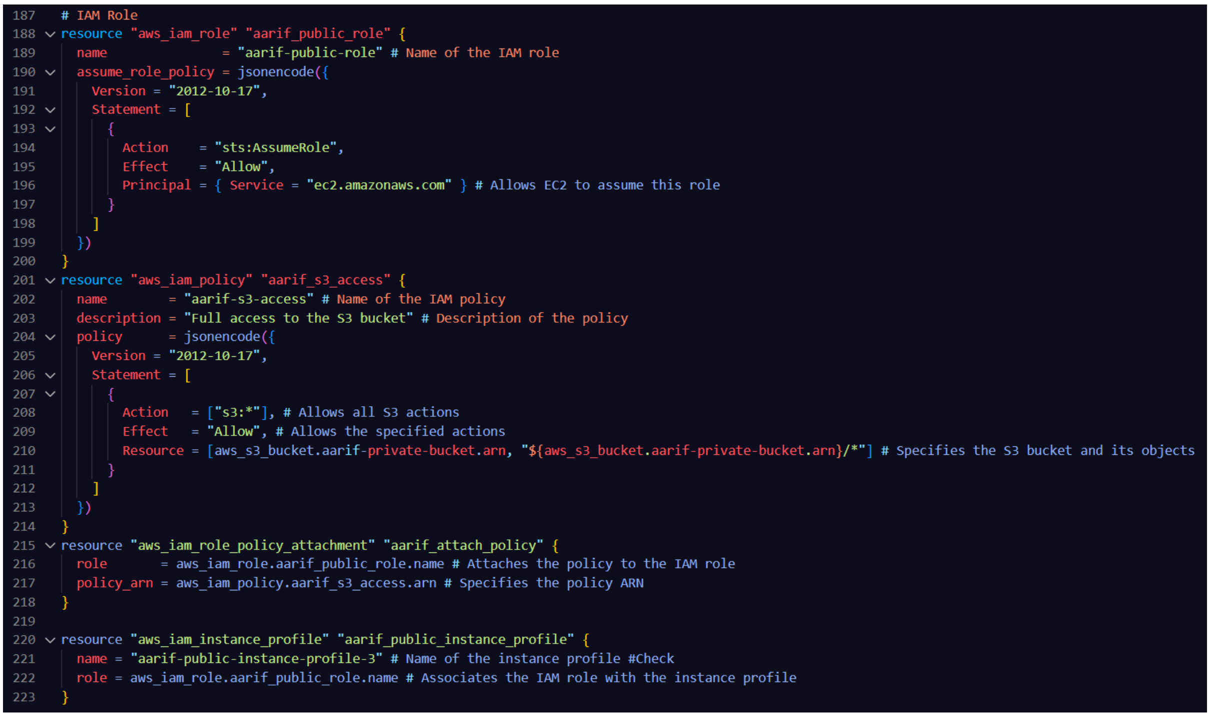
```
# Load Balancers
resource "aws_lb" "aarif_application" {
  name               = "aarif-app-lb" # Name of the load balancer
  internal           = false # Indicates it's an internet-facing load balancer
  load_balancer_type = "application" # Type of load balancer
  security_groups    = [aws_security_group.aarif_public_sg.id] # Associates the security group
  subnets            = aws_subnet.public[*].id # Subnets for the load balancer
}

resource "aws_lb_target_group" "aarif_app_targets" {
  name     = "aarif-app-targets" # Name of the target group
  port     = 80 # Port for the target group
  protocol = "HTTP" # Protocol for the target group
  vpc_id   = aws_vpc.main.id # Associates the target group with the VPC
}

resource "aws_lb_listener" "aarif_app_listener" {
  load_balancer_arn = aws_lb.aarif_application.arn # Load balancer ARN
  port              = 80 # Port for the listener
  protocol          = "HTTP" # Protocol for the listener
  default_action {
    type             = "forward" # Action type
    target_group_arn = aws_lb_target_group.aarif_app_targets.arn # Target group ARN
  }
}

resource "aws_lb" "aarif_network" {
  name               = "aarif-net-lb" # Name of the load balancer
  internal           = true # Indicates it's an internal load balancer
  load_balancer_type = "network" # Type of load balancer
  subnets            = aws_subnet.private[*].id # Subnets for the load balancer
}

resource "aws_lb_target_group" "aarif_network_targets" {
  name     = "aarif-net-targets" # Name of the target group
  port     = 80 # Port for the target group
  protocol = "TCP" # Protocol for the target group
  vpc_id   = aws_vpc.main.id # Associates the target group with the VPC
}
```

Name- Aarif M

Emp- 2355418



```
# S3 Bucket
resource "aws_s3_bucket" "aarif-private-bucket" {
  bucket = "aarif-private-bucket" # Name of the S3 bucket
}

resource "aws_s3_bucket_ownership_controls" "aarif_private_bucket_ownership_controls" {
  bucket = aws_s3_bucket.aarif-private-bucket.id # Associates the ownership controls with the bucket
  rule {
    object_ownership = "BucketOwnerPreferred" # Sets the object ownership rule
  }
}

# S3 Bucket ACL
resource "aws_s3_bucket_acl" "aarif_private_bucket_acl" {
  depends_on = [aws_s3_bucket_ownership_controls.aarif_private_bucket_ownership_controls] # Ensures ownership controls are created first
  bucket = aws_s3_bucket.aarif-private-bucket.id # Associates the ACL with the bucket
  acl = "private" # Applies private ACL
}

resource "aws_s3_bucket_versioning" "s3_versioning" {
  bucket = aws_s3_bucket.aarif-private-bucket.id # Associates versioning with the bucket
  versioning_configuration {
    status = "Enabled" # Enables versioning
  }
}
```



```
# IAM Role
resource "aws_iam_role" "aarif_public_role" {
  name               = "aarif-public-role" # Name of the IAM role
  assume_role_policy = jsonencode({
    Version = "2012-10-17",
    Statement = [
      {
        Action    = "sts:AssumeRole",
        Effect    = "Allow",
        Principal = { Service = "ec2.amazonaws.com" } # Allows EC2 to assume this role
      }
    ]
  })
}
resource "aws_iam_policy" "aarif_s3_access" {
  name        = "aarif-s3-access" # Name of the IAM policy
  description = "Full access to the S3 bucket" # Description of the policy
  policy      = jsonencode({
    Version = "2012-10-17",
    Statement = [
      {
        Action   = ["s3:*"], # Allows all S3 actions
        Effect   = "Allow", # Allows the specified actions
        Resource = [aws_s3_bucket.aarif-private-bucket.arn, "${aws_s3_bucket.aarif-private-bucket.arn}/*"] # Specifies the S3 bucket and its objects
      }
    ]
  })
}
resource "aws_iam_role_policy_attachment" "aarif_attach_policy" {
  role       = aws_iam_role.aarif_public_role.name # Attaches the policy to the IAM role
  policy_arn = aws_iam_policy.aarif_s3_access.arn # Specifies the policy ARN
}

resource "aws_iam_instance_profile" "aarif_public_instance_profile" {
  name = "aarif-public-instance-profile-3" # Name of the instance profile #Check
  role = aws_iam_role.aarif_public_role.name # Associates the IAM role with the instance profile
}
```

This Terraform code sets up an AWS infrastructure with a VPC, subnets, security groups, EC2 instances, load balancers, an S3 bucket, and IAM roles. It begins by configuring the AWS provider for the us-west-2 region and creating a VPC with a CIDR block of 10.0.0.0/16. Two public and two private subnets are created within this VPC, each spanning different availability zones. An internet gateway is attached to the VPC, and a route table is configured to allow internet access for the public subnets.

Security groups are defined for both public and private subnets. The public security group allows HTTP and SSH traffic, while the private security group allows all traffic from the public security group. An autoscaling group (ASG) is set up in the public subnets, using a launch template for EC2 instances. A single EC2 instance is launched in one of the private subnets. The code also sets up an Application Load Balancer (ALB) for the ASG and a Network Load Balancer (NLB) for the private EC2 instance. An S3 bucket is created with private access and versioning enabled. Finally, an IAM role with full access to the S3 bucket is created and attached to the instances in the ASG. This setup ensures a secure, scalable, and highly available infrastructure.

Name- Aarif M

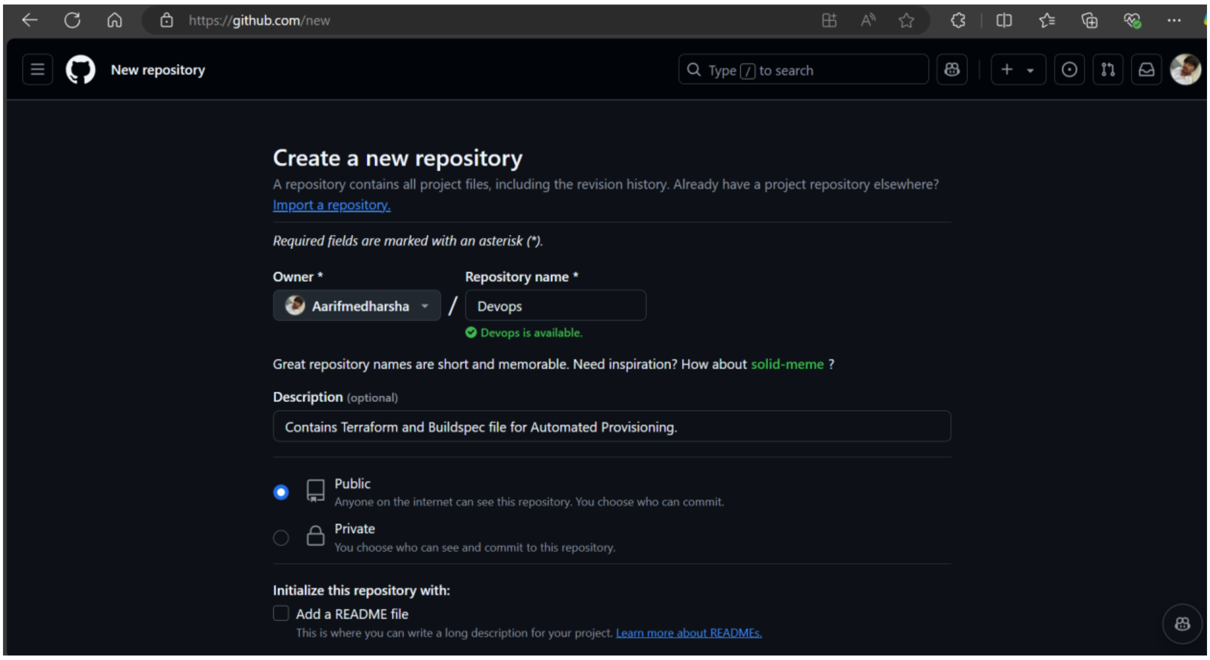
Emp- 2355418

Stage 2 – GitHub:

GitHub is a web-based platform that uses Git for version control, enabling developers to collaborate on projects efficiently. It allows users to host and review code, manage projects, and build software alongside millions of other developers. For DevOps, GitHub is invaluable as it integrates seamlessly with various CI/CD tools, automating the software development lifecycle. It supports continuous integration, continuous deployment, and continuous delivery, ensuring that code changes are automatically tested and deployed. GitHub Actions, a feature of GitHub, allows developers to create custom workflows for their projects, further enhancing automation.

My Repository: https://github.com/Aarifmedharsha/Devops1

Creating a repository on Github:



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * Aarifmedharsha / Repository name * Devops

Devops is available.

Great repository names are short and memorable. Need inspiration? How about solid-meme ?

Description (optional)

Contains Terraform and Buildspec file for Automated Provisioning.

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

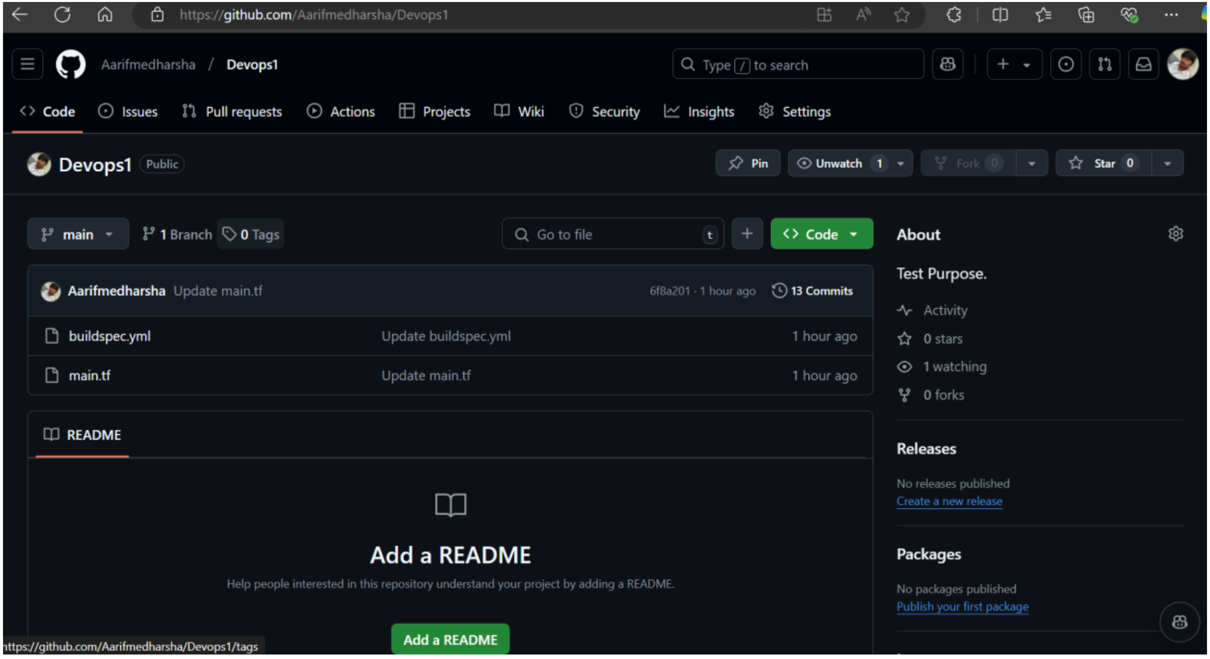
Initialize this repository with:

□ Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Name- Aarif M

Emp- 2355418

Adding files using Git (or) GitHub UI:



https://github.com/Aarifmedharsha/Devops1

Aarifmedharsha / Devops1

Devops1 Public

main · 1 Branch · 0 Tags

Aarifmedharsha Update main.tf · 6f8a201 · 1 hour ago · 13 Commits

buildspec.yml	Update buildspec.yml	1 hour ago
main.tf	Update main.tf	1 hour ago

README

Add a README

Help people interested in this repository understand your project by adding a README.

About

Test Purpose.

Activity · 0 stars · 1 watching · 0 forks

Releases

No releases published

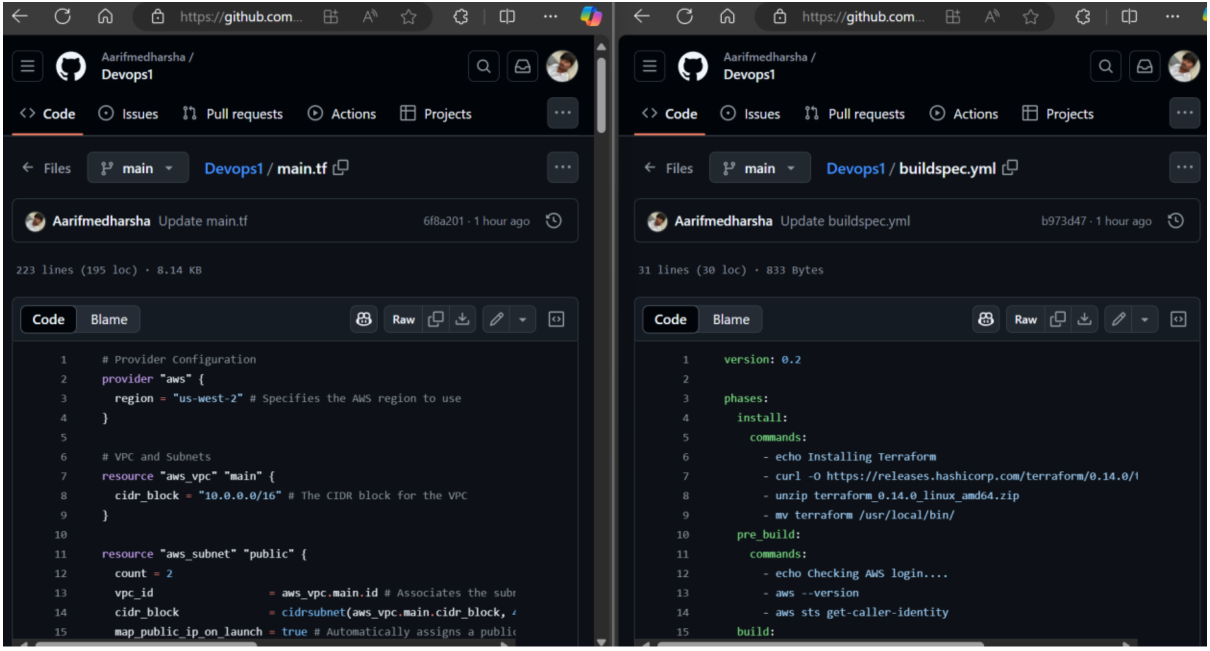
Create a new release

Packages

No packages published

Publish your first package

Uploaded files:



Devops1 / main.tf

Aarifmedharsha Update main.tf · 6f8a201 · 1 hour ago

223 lines (195 loc) · 8.14 KB

```
# Provider Configuration
provider "aws" {
  region = "us-west-2" # Specifies the AWS region to use
}

# VPC and Subnets
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16" # The CIDR block for the VPC
}

resource "aws_subnet" "public" {
  count = 2
  vpc_id                  = aws_vpc.main.id # Associates the subr
  cidr_block              = cidrsubnet(aws_vpc.main.cidr_block, 4
  map_public_ip_on_launch = true # Automatically assigns a public
```

Devops1 / buildspec.yml

Aarifmedharsha Update buildspec.yml · b973d47 · 1 hour ago

31 lines (30 loc) · 833 Bytes

```
version: 0.2

phases:
  install:
    commands:
      - echo Installing Terraform
      - curl -O https://releases.hashicorp.com/terraform/0.14.0/t
      - unzip terraform_0.14.0_linux_amd64.zip
      - mv terraform /usr/local/bin/
  pre_build:
    commands:
      - echo Checking AWS login....
      - aws --version
      - aws sts get-caller-identity
  build:
```

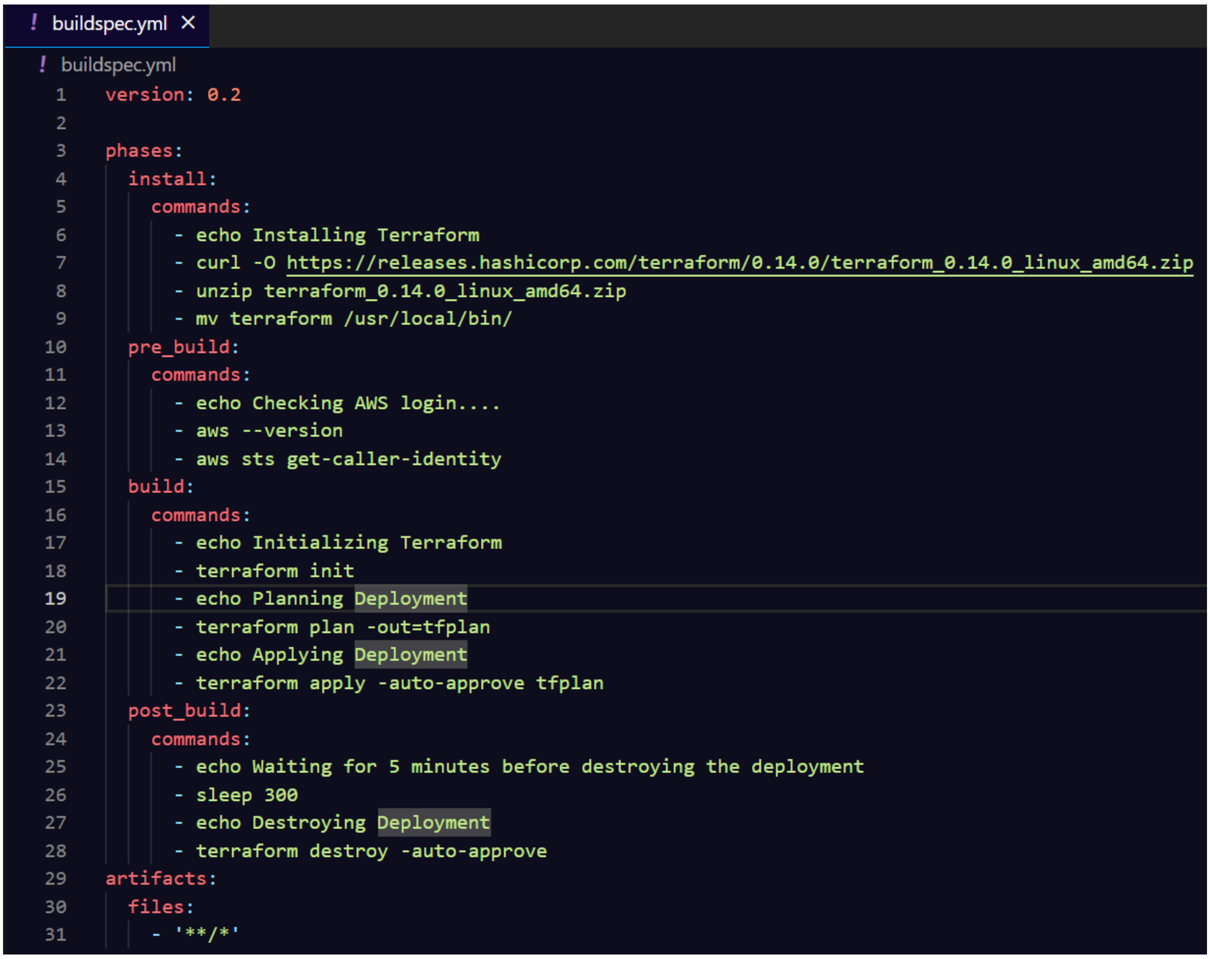
Name- Aarif M

Emp- 2355418

Stage 3 – AWS (AMAZON WEB SERVICES):

AWS CodeBuild is a fully managed build service that compiles source code, runs tests, and produces software packages ready for deployment. It eliminates the need to provision, manage, and scale your own build servers, allowing you to focus on writing code. CodeBuild scales continuously and processes multiple builds concurrently, ensuring that your builds are not left waiting in a queue.

buildspec.yml:



```yaml
version: 0.2

phases:
  install:
    commands:
      - echo Installing Terraform
      - curl -O https://releases.hashicorp.com/terraform/0.14.0/terraform_0.14.0_linux_amd64.zip
      - unzip terraform_0.14.0_linux_amd64.zip
      - mv terraform /usr/local/bin/
  pre_build:
    commands:
      - echo Checking AWS login....
      - aws --version
      - aws sts get-caller-identity
  build:
    commands:
      - echo Initializing Terraform
      - terraform init
      - echo Planning Deployment
      - terraform plan -out=tfplan
      - echo Applying Deployment
      - terraform apply -auto-approve tfplan
  post_build:
    commands:
      - echo Waiting for 5 minutes before destroying the deployment
      - sleep 300
      - echo Destroying Deployment
      - terraform destroy -auto-approve
artifacts:
  files:
    - '**/*'
```

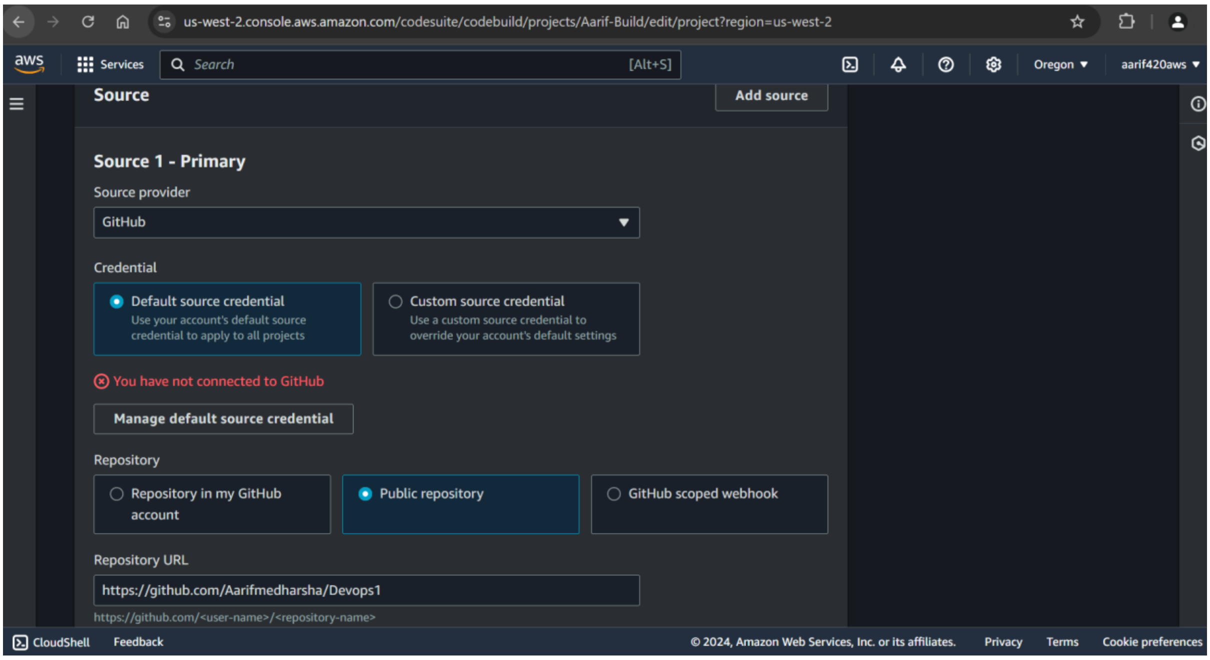
A key component of AWS CodeBuild is the buildspec file, typically named buildspec.yml. This file is written in YAML format and defines the build commands and settings used by CodeBuild to run a build. The **buildspec.yml** file is a configuration file used by AWS CodeBuild to define the build process for your project. It includes several key components**: version**, which specifies the buildspec version; **env,** where you define environment variables and runtime settings; phases, which outline the steps of the build lifecycle including install, pre_build, build, and post_build phases; artifacts, which specify the output files to be stored after the build; cache, which defines paths to cache to speed up subsequent builds; and reports, which generate reports about the build process. Each section allows you to customize and control different aspects of the build, ensuring a consistent and automated build process.

Name- Aarif M

Emp- 2355418

Code Build Setup:

After Creating and add a name to new project, we can use Github as a Primary source from where we can fetch the terraform code. No credentials required as it is a public repository.



us-west-2.console.aws.amazon.com/codesuite/codebuild/projects/Aarif-Build/edit/project?region=us-west-2

Source

Source 1 - Primary

Source provider: GitHub

Credential

● Default source credential
Use your account's default source credential to apply to all projects

○ Custom source credential
Use a custom source credential to override your account's default settings

You have not connected to GitHub

Repository

○ Repository in my GitHub account

● Public repository

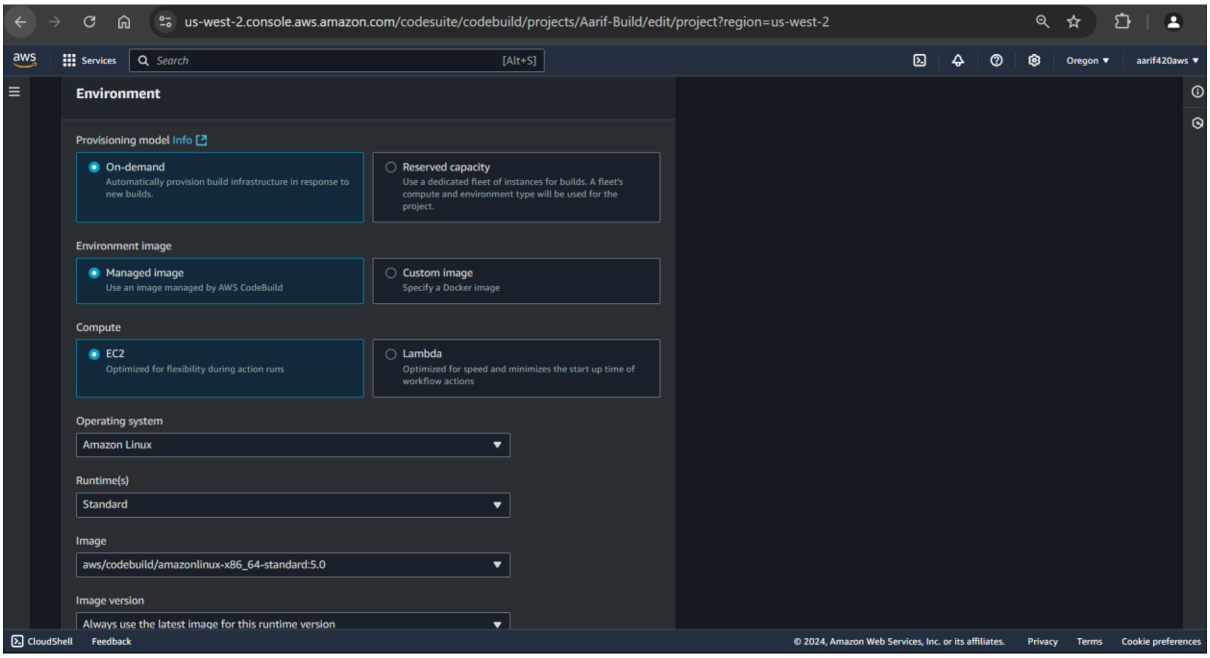
○ GitHub scoped webhook

Repository URL: https://github.com/Aarifmedharsha/Devops1

https://github.com/<user-name>/<repository-name>

© 2024, Amazon Web Services, Inc. or its affiliates.

Then we have to setup the environment for our build



us-west-2.console.aws.amazon.com/codesuite/codebuild/projects/Aarif-Build/edit/project?region=us-west-2

Environment

Provisioning model Info

● On-demand
Automatically provision build infrastructure in response to new builds.

○ Reserved capacity
Use a dedicated fleet of instances for builds. A fleet's compute and environment type will be used for the project.

Environment image

● Managed image
Use an image managed by AWS CodeBuild

○ Custom image
Specify a Docker image

Compute

● EC2
Optimized for flexibility during action runs

○ Lambda
Optimized for speed and minimizes the start up time of workflow actions

Operating system: Amazon Linux

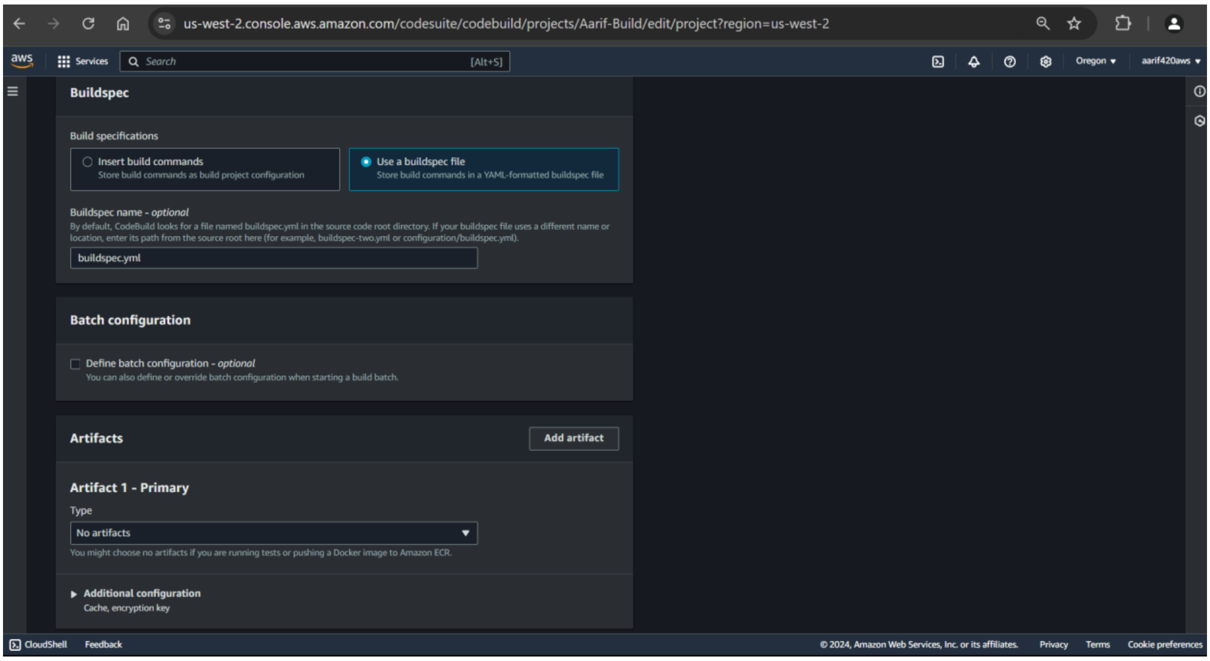
Runtime(s): Standard

Image: aws/codebuild/amazonlinux-x86_64-standard:5.0

Image version: Always use the latest image for this runtime version

© 2024, Amazon Web Services, Inc. or its affiliates.

Then we have to add the **buildspec.yml** file, which is present in the Github Repository for building.



us-west-2.console.aws.amazon.com/codesuite/codebuild/projects/Aarif-Build/edit/project?region=us-west-2

Buildspec

Build specifications

○ Insert build commands
Store build commands as build project configuration

● Use a buildspec file
Store build commands in a YAML-formatted buildspec file

Buildspec name - *optional*
By default, CodeBuild looks for a file named buildspec.yml in the source code root directory. If your buildspec file uses a different name or location, enter its path from the source root here (for example, buildspec-two.yml or configuration/buildspec.yml).

buildspec.yml

Batch configuration

□ Define batch configuration - *optional*
You can also define or override batch configuration when starting a build batch.

Artifacts

Artifact 1 - Primary

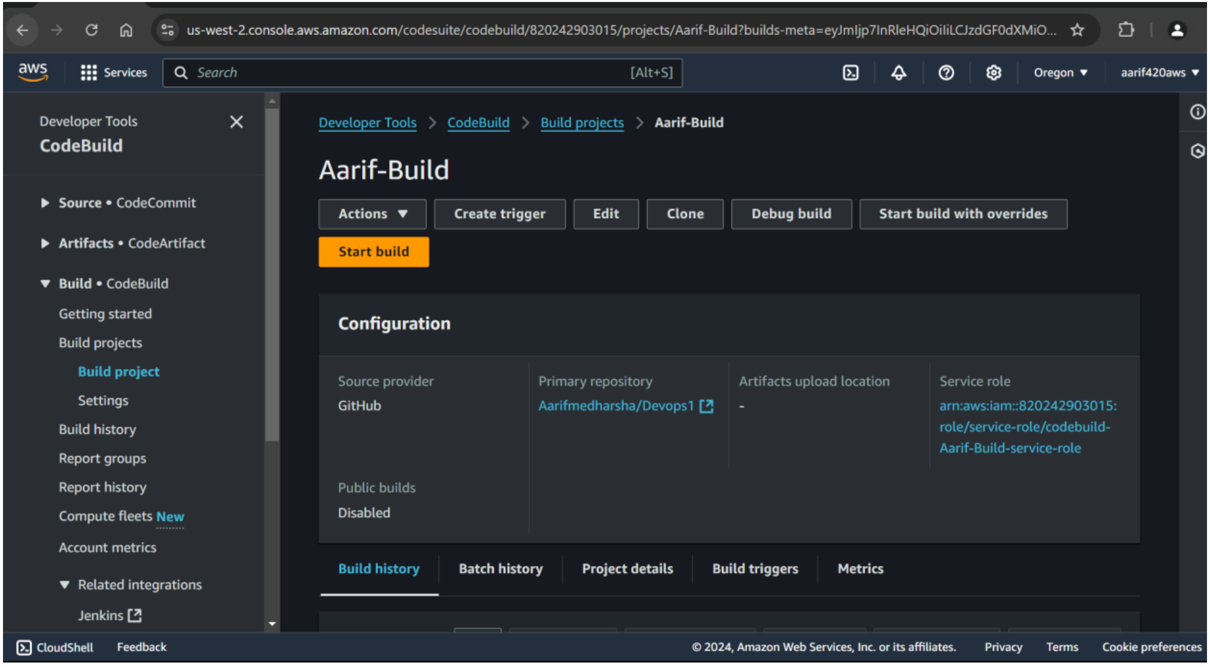
Type

No artifacts

You might choose no artifacts if you are running tests or pushing a Docker image to Amazon ECR.

▶ Additional configuration
Cache, encryption key

Now the Configuration are done, to build the project, we need to click on the **Start build button.**



us-west-2.console.aws.amazon.com/codesuite/codebuild/820242903015/projects/Aarif-Build?builds-meta=eyJmljp7InRleHQiOiIiLCJzdGF0dXMiO...

Developer Tools > CodeBuild > Build projects > Aarif-Build

Aarif-Build

Configuration

Source provider
GitHub

Primary repository
Aarifmedharsha/Devops1

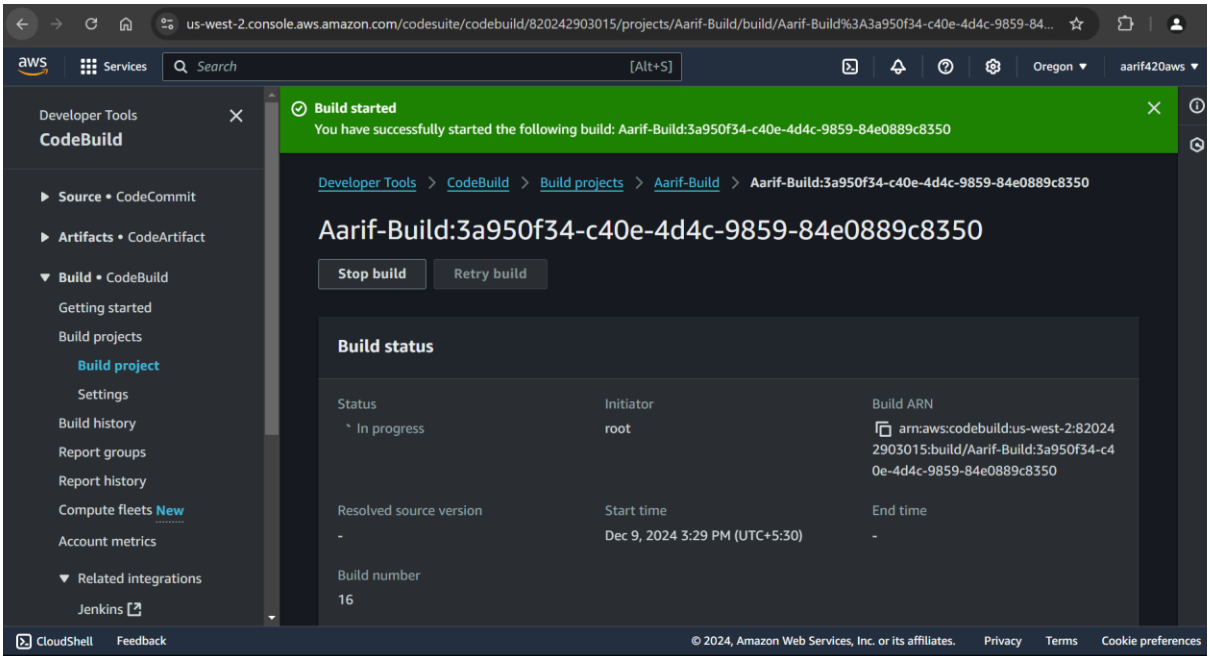
Artifacts upload location
-

Service role
arn:aws:iam::820242903015:role/service-role/codebuild-Aarif-Build-service-role

Public builds
Disabled

Name- Aarif M

Emp- 2355418



Build started
You have successfully started the following build: Aarif-Build:3a950f34-c40e-4d4c-9859-84e0889c8350

Developer Tools > CodeBuild > Build projects > Aarif-Build > Aarif-Build:3a950f34-c40e-4d4c-9859-84e0889c8350

Aarif-Build:3a950f34-c40e-4d4c-9859-84e0889c8350

Build status

Status: In progress

Initiator: root

Build ARN: arn:aws:codebuild:us-west-2:820242903015:build/Aarif-Build:3a950f34-c40e-4d4c-9859-84e0889c8350

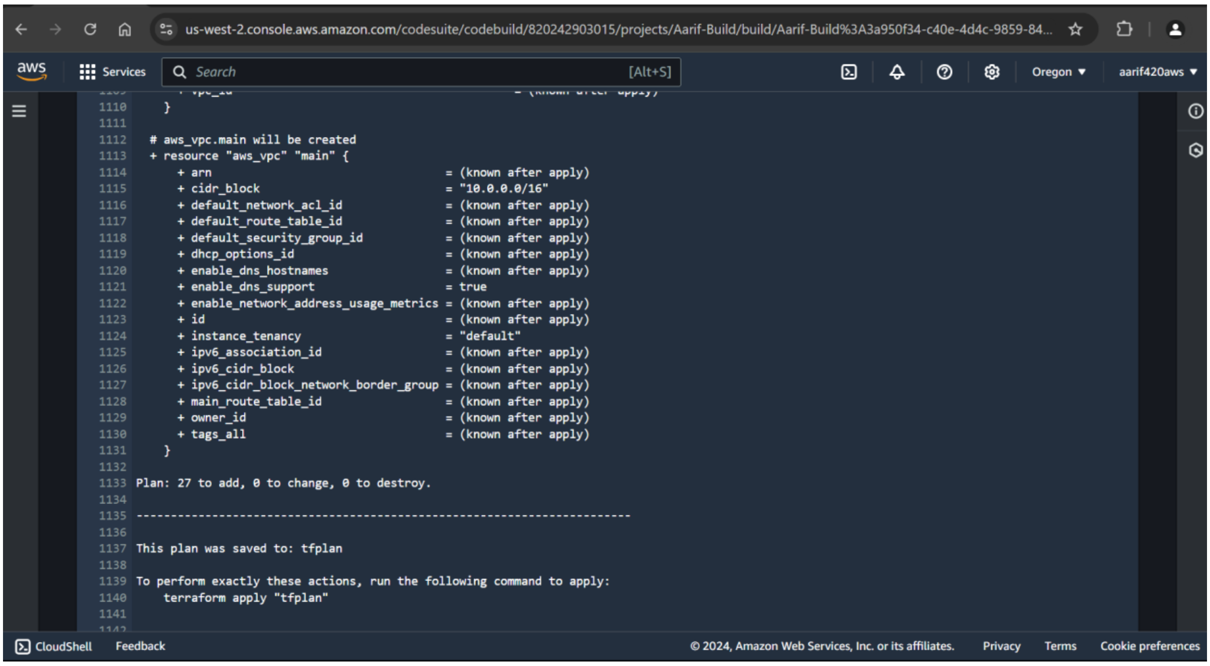
Resolved source version: -

Start time: Dec 9, 2024 3:29 PM (UTC+5:30)

End time: -

Build number: 16

While the build is in progress we can check for the logs, and we can follow along the steps. We see that the terraform is successfully installed and the plan command resulted in success.

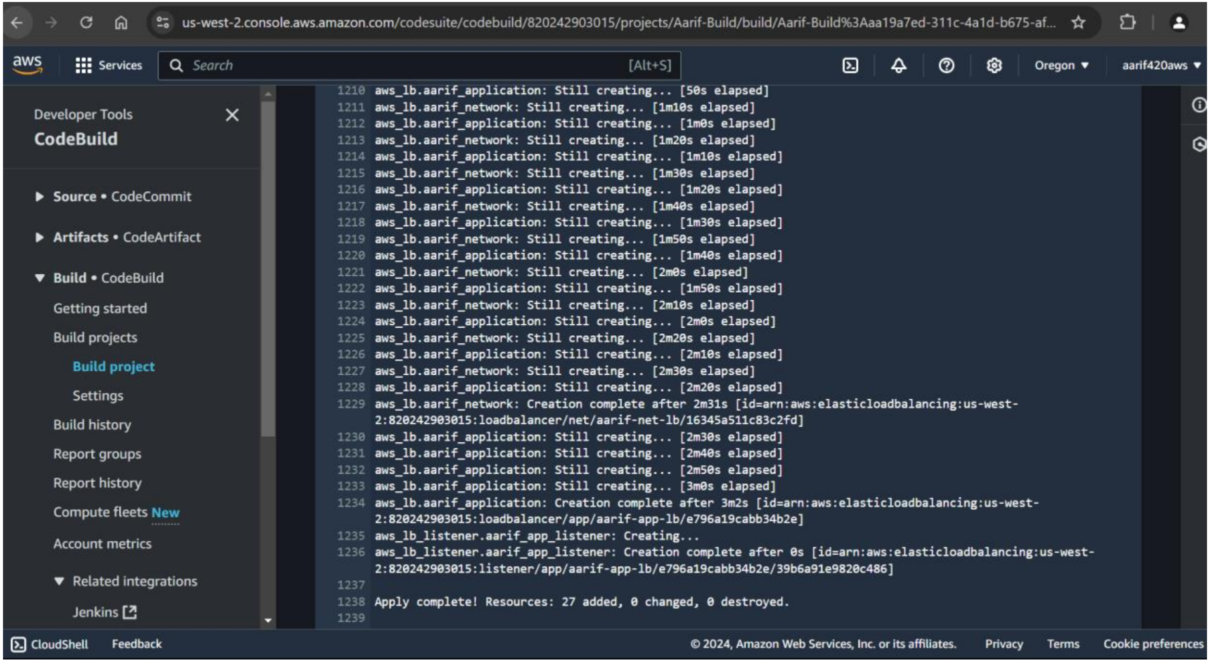

us-west-2.console.aws.amazon.com/codesuite/codebuild/820242903015/projects/Aarif-Build/build/Aarif-Build%3A3a950f34-c40e-4d4c-9859-84...
Services
Search [Alt+S]
Oregon
aarif420aws
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

```
    }

  # aws_vpc.main will be created
  + resource "aws_vpc" "main" {
      + arn                                  = (known after apply)
      + cidr_block                           = "10.0.0.0/16"
      + default_network_acl_id               = (known after apply)
      + default_route_table_id               = (known after apply)
      + default_security_group_id            = (known after apply)
      + dhcp_options_id                      = (known after apply)
      + enable_dns_hostnames                 = (known after apply)
      + enable_dns_support                   = true
      + enable_network_address_usage_metrics = (known after apply)
      + id                                   = (known after apply)
      + instance_tenancy                     = "default"
      + ipv6_association_id                  = (known after apply)
      + ipv6_cidr_block                      = (known after apply)
      + ipv6_cidr_block_network_border_group = (known after apply)
      + main_route_table_id                  = (known after apply)
      + owner_id                             = (known after apply)
      + tags_all                             = (known after apply)
    }

Plan: 27 to add, 0 to change, 0 to destroy.

─────────────────────────────────────────────────────────────────────────

This plan was saved to: tfplan

To perform exactly these actions, run the following command to apply:
    terraform apply "tfplan"
```

Name- Aarif M

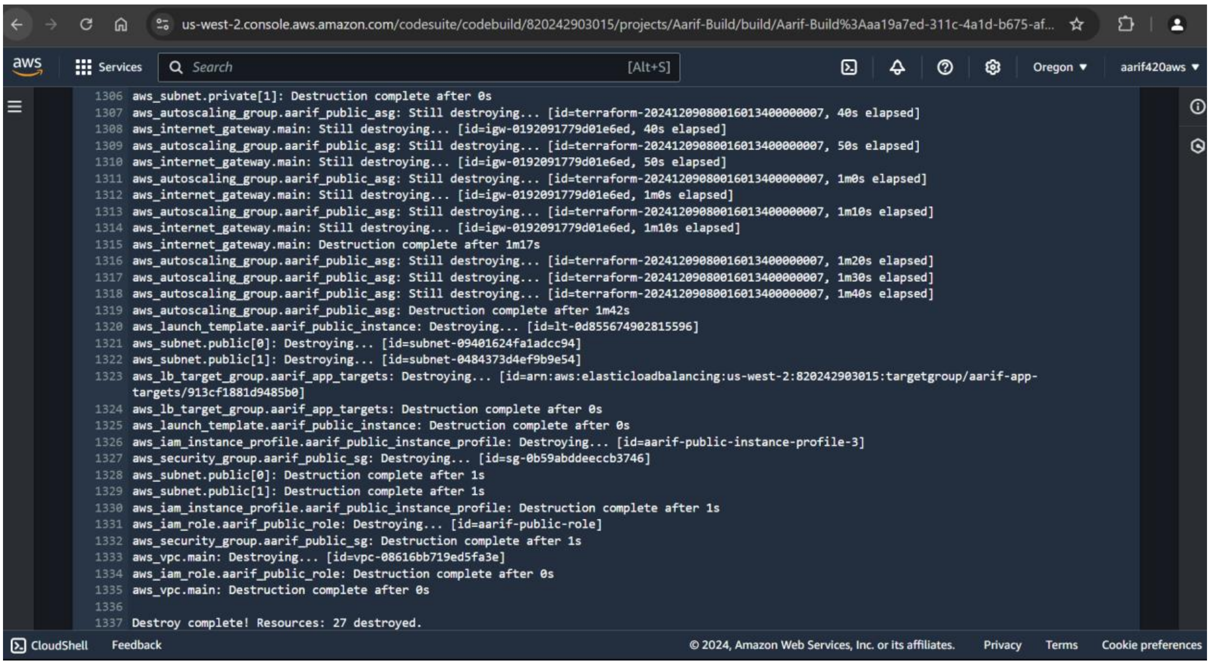
Emp- 2355418

There are 27 resources to add and all are added after apply command.



```
aws_lb.aarif_application: Still creating... [50s elapsed]
aws_lb.aarif_network: Still creating... [1m10s elapsed]
aws_lb.aarif_application: Still creating... [1m0s elapsed]
aws_lb.aarif_network: Still creating... [1m20s elapsed]
aws_lb.aarif_application: Still creating... [1m10s elapsed]
aws_lb.aarif_network: Still creating... [1m30s elapsed]
aws_lb.aarif_application: Still creating... [1m20s elapsed]
aws_lb.aarif_network: Still creating... [1m40s elapsed]
aws_lb.aarif_application: Still creating... [1m30s elapsed]
aws_lb.aarif_network: Still creating... [1m50s elapsed]
aws_lb.aarif_application: Still creating... [1m40s elapsed]
aws_lb.aarif_network: Still creating... [2m0s elapsed]
aws_lb.aarif_application: Still creating... [1m50s elapsed]
aws_lb.aarif_network: Still creating... [2m10s elapsed]
aws_lb.aarif_application: Still creating... [2m0s elapsed]
aws_lb.aarif_network: Still creating... [2m20s elapsed]
aws_lb.aarif_application: Still creating... [2m10s elapsed]
aws_lb.aarif_network: Still creating... [2m30s elapsed]
aws_lb.aarif_application: Still creating... [2m20s elapsed]
aws_lb.aarif_network: Creation complete after 2m31s [id=arn:aws:elasticloadbalancing:us-west-
2:820242903015:loadbalancer/net/aarif-net-lb/16345a511c83c2fd]
aws_lb.aarif_application: Still creating... [2m30s elapsed]
aws_lb.aarif_application: Still creating... [2m40s elapsed]
aws_lb.aarif_application: Still creating... [2m50s elapsed]
aws_lb.aarif_application: Still creating... [3m0s elapsed]
aws_lb.aarif_application: Creation complete after 3m2s [id=arn:aws:elasticloadbalancing:us-west-
2:820242903015:loadbalancer/app/aarif-app-lb/e796a19cabb34b2e]
aws_lb_listener.aarif_app_listener: Creating...
aws_lb_listener.aarif_app_listener: Creation complete after 0s [id=arn:aws:elasticloadbalancing:us-west-
2:820242903015:listener/app/aarif-app-lb/e796a19cabb34b2e/39b6a91e9820c486]

Apply complete! Resources: 27 added, 0 changed, 0 destroyed.

```

Now , after successful creation as specified in the buildspec.yml the script waits for 5 mins then executes terraform destroy to delete all the resources that are created.

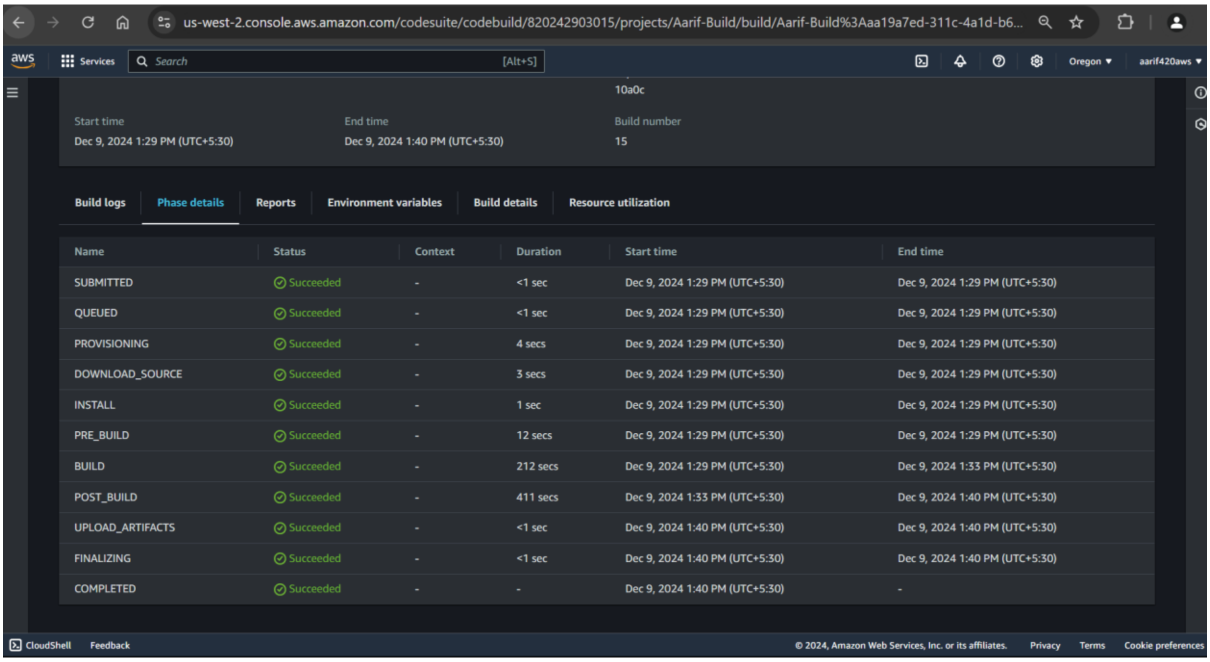

us-west-2.console.aws.amazon.com/codesuite/codebuild/820242903015/projects/Aarif-Build/build/Aarif-Build%3Aaa19a7ed-311c-4a1d-b675-af...
aws
Services
Search
[Alt+S]
Oregon ▾
aarif420aws ▾
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

```
aws_subnet.private[1]: Destruction complete after 0s
aws_autoscaling_group.aarif_public_asg: Still destroying... [id=terraform-20241209080016013400000007, 40s elapsed]
aws_internet_gateway.main: Still destroying... [id=igw-0192091779d01e6ed, 40s elapsed]
aws_autoscaling_group.aarif_public_asg: Still destroying... [id=terraform-20241209080016013400000007, 50s elapsed]
aws_internet_gateway.main: Still destroying... [id=igw-0192091779d01e6ed, 50s elapsed]
aws_autoscaling_group.aarif_public_asg: Still destroying... [id=terraform-20241209080016013400000007, 1m0s elapsed]
aws_internet_gateway.main: Still destroying... [id=igw-0192091779d01e6ed, 1m0s elapsed]
aws_autoscaling_group.aarif_public_asg: Still destroying... [id=terraform-20241209080016013400000007, 1m10s elapsed]
aws_internet_gateway.main: Still destroying... [id=igw-0192091779d01e6ed, 1m10s elapsed]
aws_internet_gateway.main: Destruction complete after 1m17s
aws_autoscaling_group.aarif_public_asg: Still destroying... [id=terraform-20241209080016013400000007, 1m20s elapsed]
aws_autoscaling_group.aarif_public_asg: Still destroying... [id=terraform-20241209080016013400000007, 1m30s elapsed]
aws_autoscaling_group.aarif_public_asg: Still destroying... [id=terraform-20241209080016013400000007, 1m40s elapsed]
aws_autoscaling_group.aarif_public_asg: Destruction complete after 1m42s
aws_launch_template.aarif_public_instance: Destroying... [id=lt-0d855674902815596]
aws_subnet.public[0]: Destroying... [id=subnet-09401624fa1adcc94]
aws_subnet.public[1]: Destroying... [id=subnet-0484373d4ef9b9e54]
aws_lb_target_group.aarif_app_targets: Destroying... [id=arn:aws:elasticloadbalancing:us-west-2:820242903015:targetgroup/aarif-app-
targets/913cf1881d9485b0]
aws_lb_target_group.aarif_app_targets: Destruction complete after 0s
aws_launch_template.aarif_public_instance: Destruction complete after 0s
aws_iam_instance_profile.aarif_public_instance_profile: Destroying... [id=aarif-public-instance-profile-3]
aws_security_group.aarif_public_sg: Destroying... [id=sg-0b59abddeeccb3746]
aws_subnet.public[0]: Destruction complete after 1s
aws_subnet.public[1]: Destruction complete after 1s
aws_iam_instance_profile.aarif_public_instance_profile: Destruction complete after 1s
aws_iam_role.aarif_public_role: Destroying... [id=aarif-public-role]
aws_security_group.aarif_public_sg: Destruction complete after 1s
aws_vpc.main: Destroying... [id=vpc-08616bb719ed5fa3e]
aws_iam_role.aarif_public_role: Destruction complete after 0s
aws_vpc.main: Destruction complete after 0s

Destroy complete! Resources: 27 destroyed.
```

Name- Aarif M

Emp- 2355418

Phase Details:



us-west-2.console.aws.amazon.com/codesuite/codebuild/820242903015/projects/Aarif-Build/build/Aarif-Build%3Aaa19a7ed-311c-4a1d-b6...

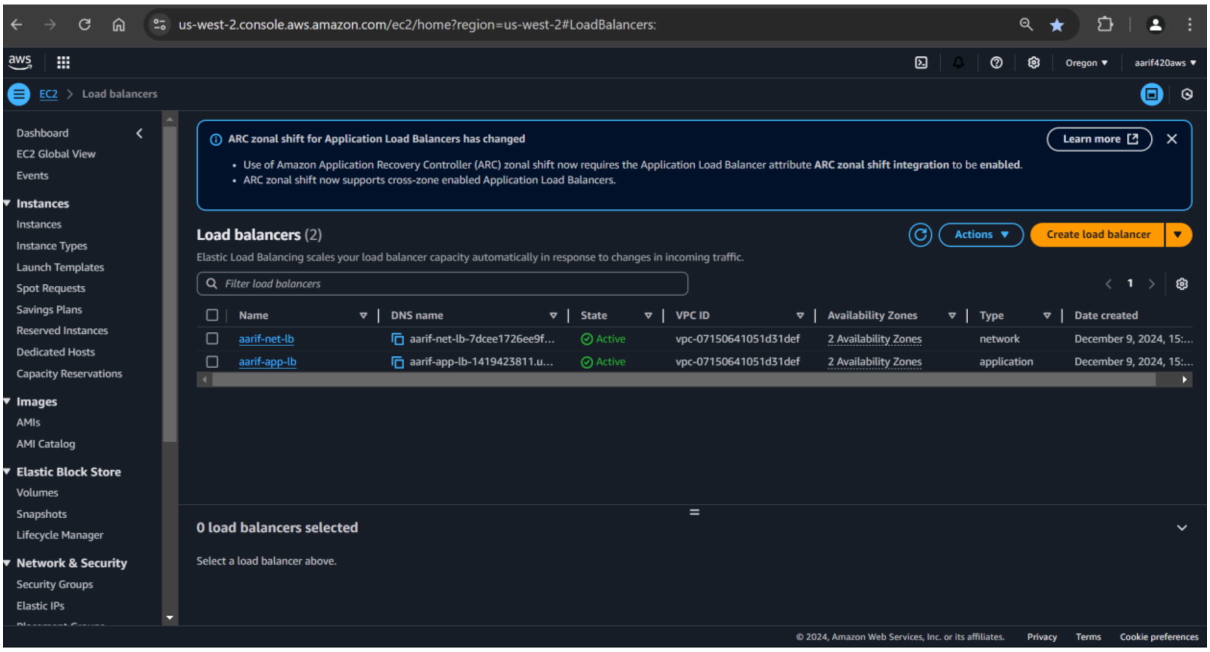
10a0c

Start time	End time	Build number
Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:40 PM (UTC+5:30)	15

Name	Status	Context	Duration	Start time	End time
SUBMITTED	Succeeded	-	<1 sec	Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:29 PM (UTC+5:30)
QUEUED	Succeeded	-	<1 sec	Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:29 PM (UTC+5:30)
PROVISIONING	Succeeded	-	4 secs	Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:29 PM (UTC+5:30)
DOWNLOAD_SOURCE	Succeeded	-	3 secs	Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:29 PM (UTC+5:30)
INSTALL	Succeeded	-	1 sec	Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:29 PM (UTC+5:30)
PRE_BUILD	Succeeded	-	12 secs	Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:29 PM (UTC+5:30)
BUILD	Succeeded	-	212 secs	Dec 9, 2024 1:29 PM (UTC+5:30)	Dec 9, 2024 1:33 PM (UTC+5:30)
POST_BUILD	Succeeded	-	411 secs	Dec 9, 2024 1:33 PM (UTC+5:30)	Dec 9, 2024 1:40 PM (UTC+5:30)
UPLOAD_ARTIFACTS	Succeeded	-	<1 sec	Dec 9, 2024 1:40 PM (UTC+5:30)	Dec 9, 2024 1:40 PM (UTC+5:30)
FINALIZING	Succeeded	-	<1 sec	Dec 9, 2024 1:40 PM (UTC+5:30)	Dec 9, 2024 1:40 PM (UTC+5:30)
COMPLETED	Succeeded	-	-	Dec 9, 2024 1:40 PM (UTC+5:30)	-

Cross verification of the resources created on the UI.

Load Balancers :



us-west-2.console.aws.amazon.com/ec2/home?region=us-west-2#LoadBalancers:

ARC zonal shift for Application Load Balancers has changed

- Use of Amazon Application Recovery Controller (ARC) zonal shift now requires the Application Load Balancer attribute **ARC zonal shift integration** to be **enabled**.
- ARC zonal shift now supports cross-zone enabled Application Load Balancers.

Load balancers (2)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
aarif-net-lb	aarif-net-lb-7dcee1726ee9f...	Active	vpc-07150641051d31def	2 Availability Zones	network	December 9, 2024, 15:...
aarif-app-lb	aarif-app-lb-1419423811.u...	Active	vpc-07150641051d31def	2 Availability Zones	application	December 9, 2024, 15:...

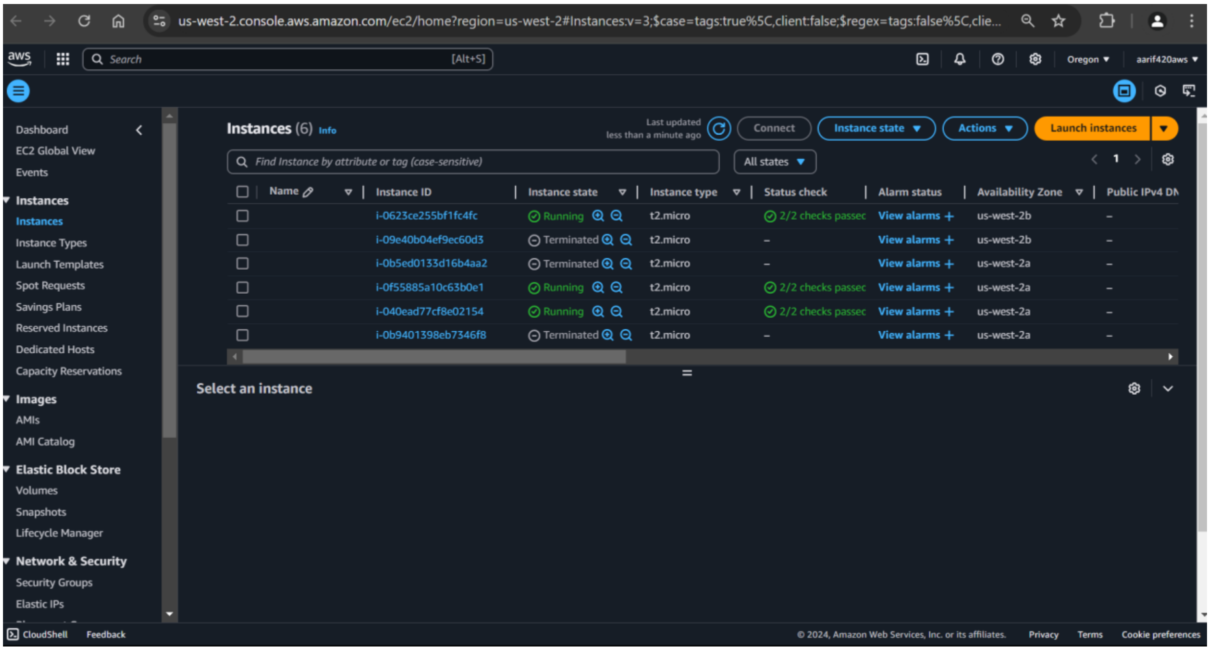
0 load balancers selected

Select a load balancer above.

Name- Aarif M

Emp- 2355418

Instances :



us-west-2.console.aws.amazon.com/ec2/home?region=us-west-2#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,clie...

Instances (6) Info

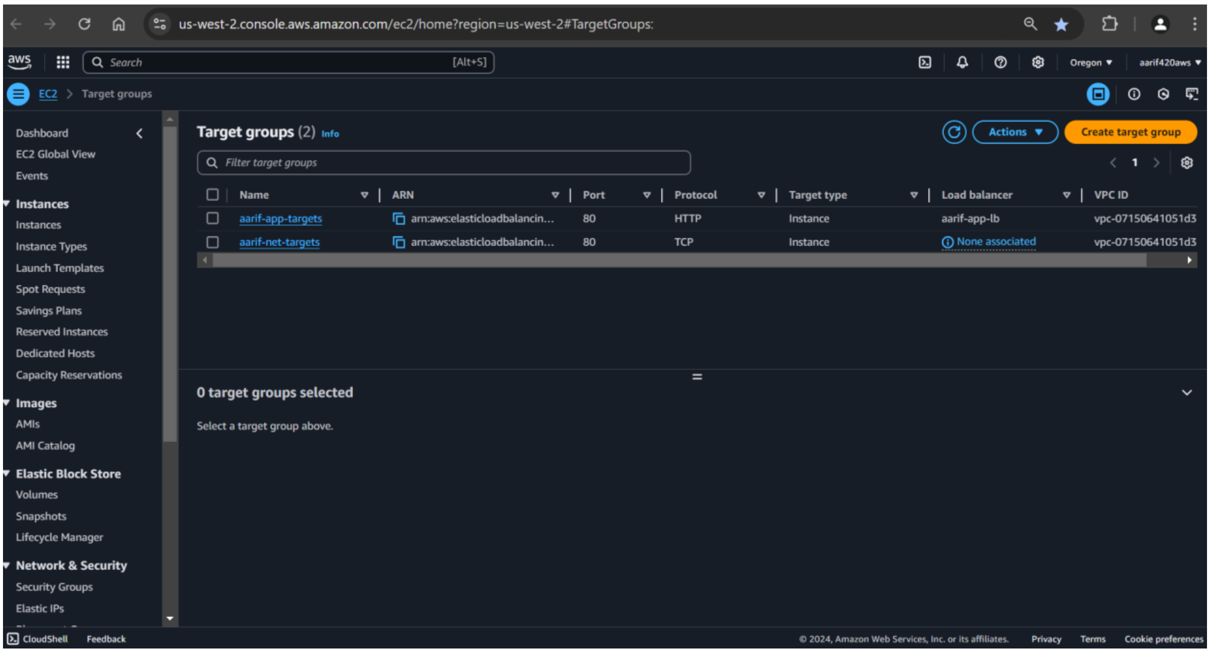
Last updated less than a minute ago

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

□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DN
□		i-0623ce255bf1fc4fc	Running	t2.micro	2/2 checks passec	View alarms +	us-west-2b	–
□		i-09e40b04ef9ec60d3	Terminated	t2.micro	–	View alarms +	us-west-2b	–
□		i-0b5ed0133d16b4aa2	Terminated	t2.micro	–	View alarms +	us-west-2a	–
□		i-0f55885a10c63b0e1	Running	t2.micro	2/2 checks passec	View alarms +	us-west-2a	–
□		i-040ead77cf8e02154	Running	t2.micro	2/2 checks passec	View alarms +	us-west-2a	–
□		i-0b9401398eb7346f8	Terminated	t2.micro	–	View alarms +	us-west-2a	–

Select an instance

Target Groups :



us-west-2.console.aws.amazon.com/ec2/home?region=us-west-2#TargetGroups:

EC2 > Target groups

Target groups (2) Info

Filter target groups

□	Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
□	aarif-app-targets	arn:aws:elasticloadbalancin...	80	HTTP	Instance	aarif-app-lb	vpc-07150641051d3
□	aarif-net-targets	arn:aws:elasticloadbalancin...	80	TCP	Instance	None associated	vpc-07150641051d3

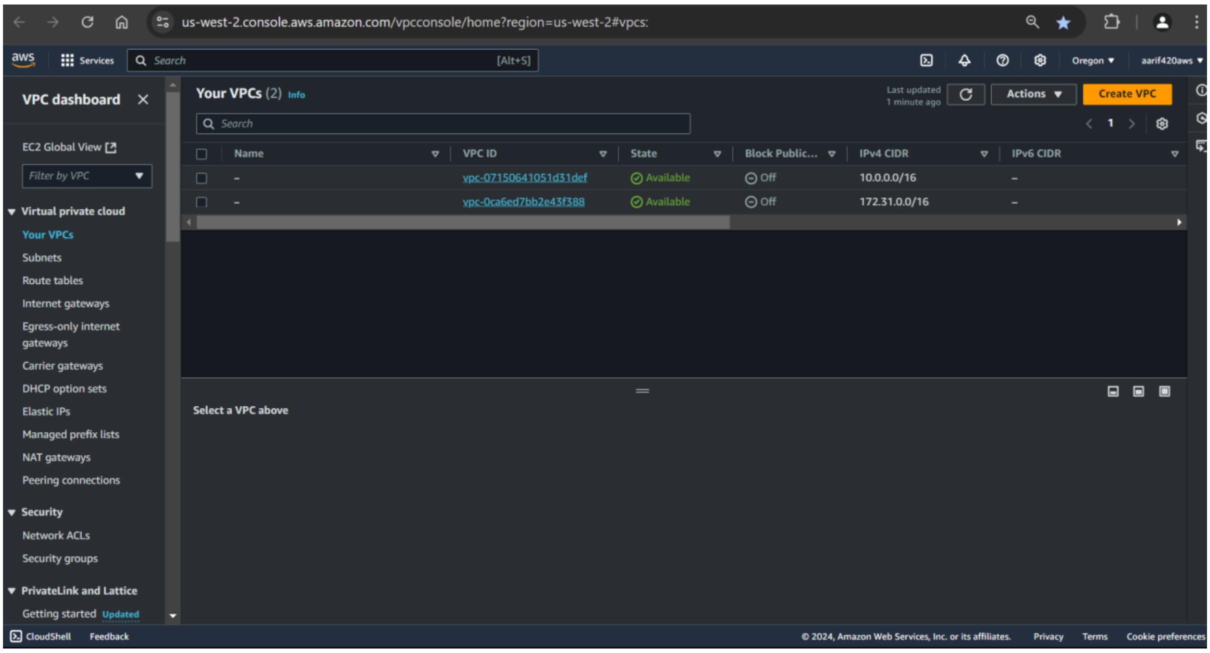
0 target groups selected

Select a target group above.

Name- Aarif M

Emp- 2355418

VPC :



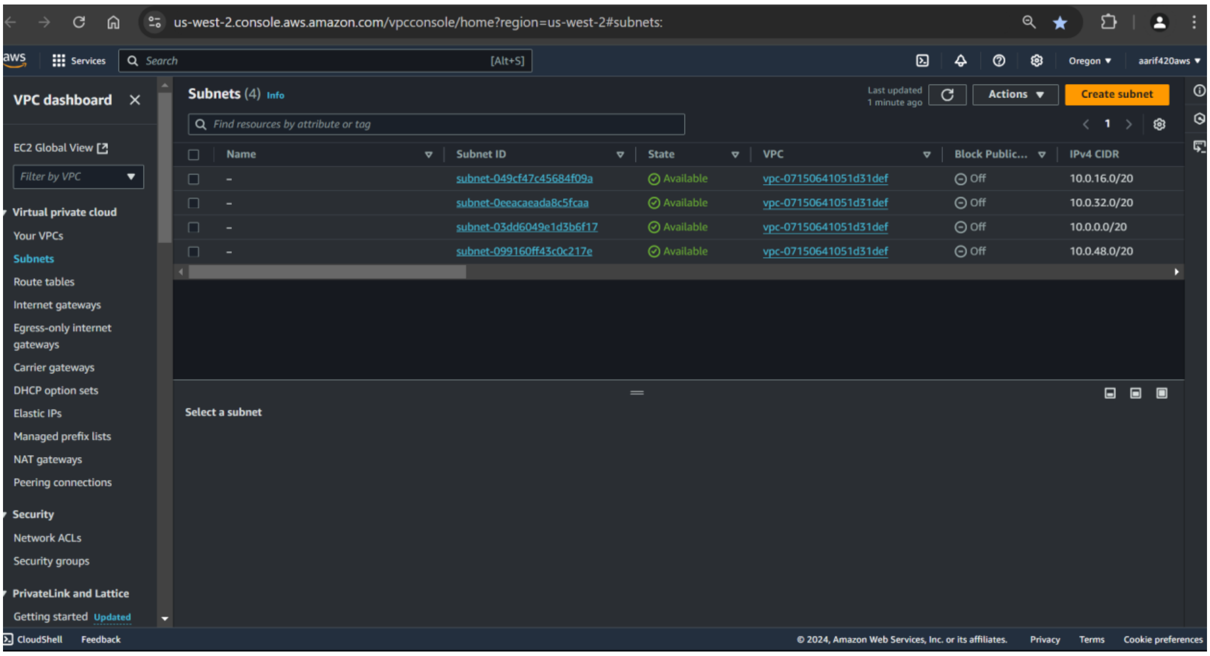
us-west-2.console.aws.amazon.com/vpcconsole/home?region=us-west-2#vpcs:

Your VPCs (2)

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR
–	vpc-07150641051d31def	Available	Off	10.0.0.0/16	–
–	vpc-0ca6ed7bb2e43f388	Available	Off	172.31.0.0/16	–

Select a VPC above

Subnets :



us-west-2.console.aws.amazon.com/vpcconsole/home?region=us-west-2#subnets:

Subnets (4)

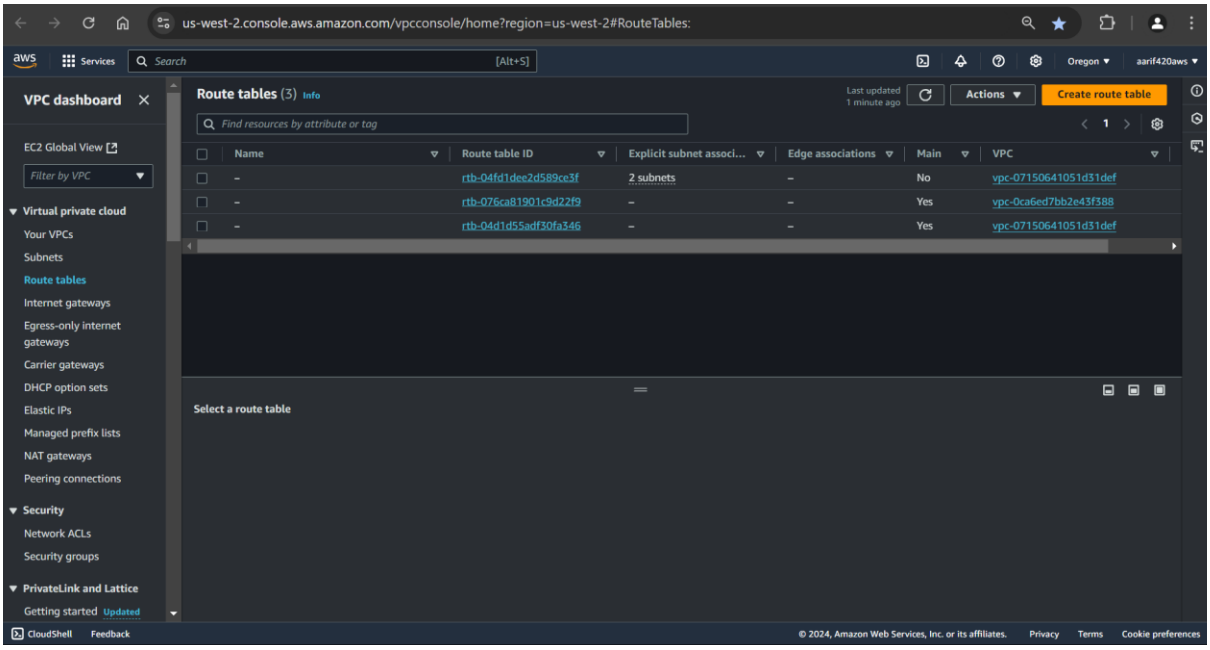
Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
–	subnet-049cf47c45684f09a	Available	vpc-07150641051d31def	Off	10.0.16.0/20
–	subnet-0eeacaeada8c5fcaa	Available	vpc-07150641051d31def	Off	10.0.32.0/20
–	subnet-03dd6049e1d3b6f17	Available	vpc-07150641051d31def	Off	10.0.0.0/20
–	subnet-099160ff43c0c217e	Available	vpc-07150641051d31def	Off	10.0.48.0/20

Select a subnet

Name- Aarif M

Emp- 2355418

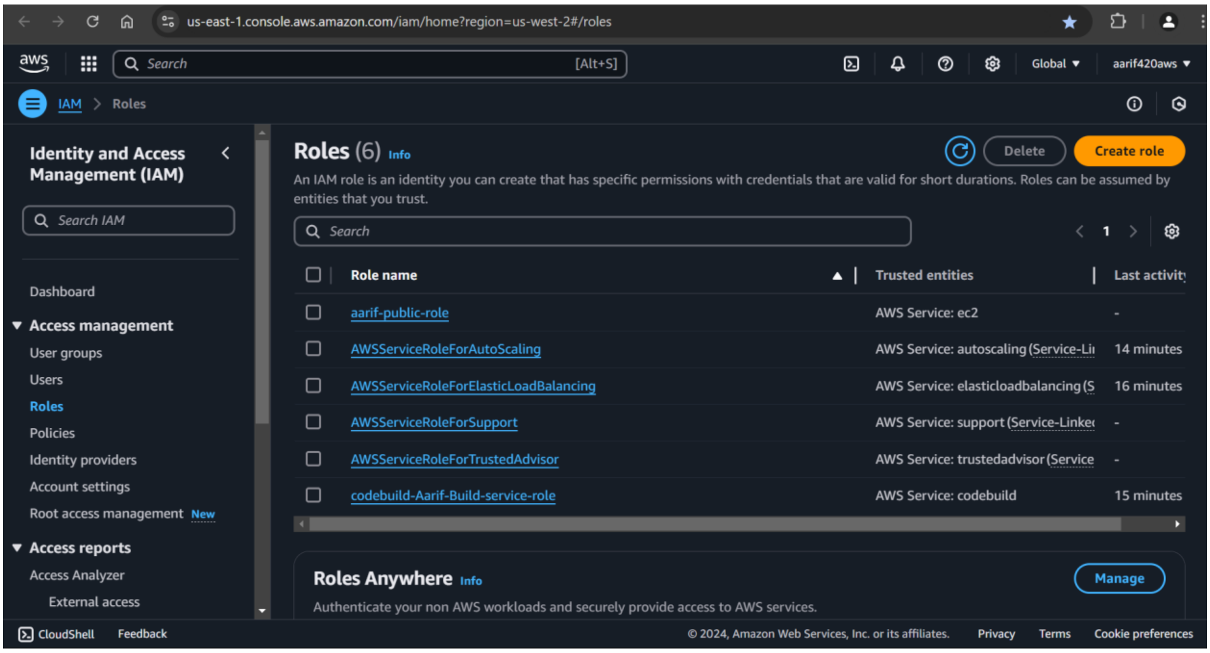
Route Tables :



Route tables (3) Info

	Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
□	–	rtb-04fd1dee2d589ce3f	2 subnets	–	No	vpc-07150641051d31def
□	–	rtb-076ca81901c9d22f9	–	–	Yes	vpc-0ca6ed7bb2e43f388
□	–	rtb-04d1d55adf30fa346	–	–	Yes	vpc-07150641051d31def

IAM Role :



Roles (6) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

□	Role name	Trusted entities	Last activity
□	aarif-public-role	AWS Service: ec2	-
□	AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Li	14 minutes
□	AWSServiceRoleForElasticLoadBalancing	AWS Service: elasticloadbalancing (S	16 minutes
□	AWSServiceRoleForSupport	AWS Service: support (Service-Linke	-
□	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service	-
□	codebuild-Aarif-Build-service-role	AWS Service: codebuild	15 minutes

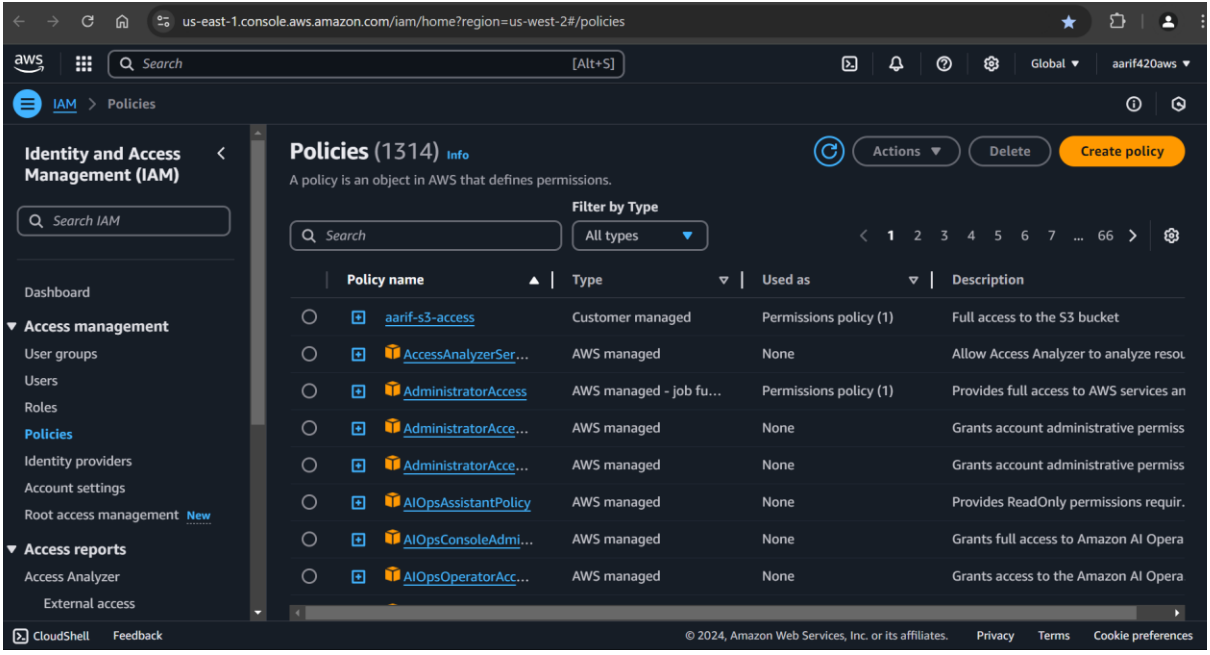
Roles Anywhere Info

Authenticate your non AWS workloads and securely provide access to AWS services.

Name- Aarif M

Emp- 2355418

IAM Policy:



us-east-1.console.aws.amazon.com/iam/home?region=us-west-2#/policies

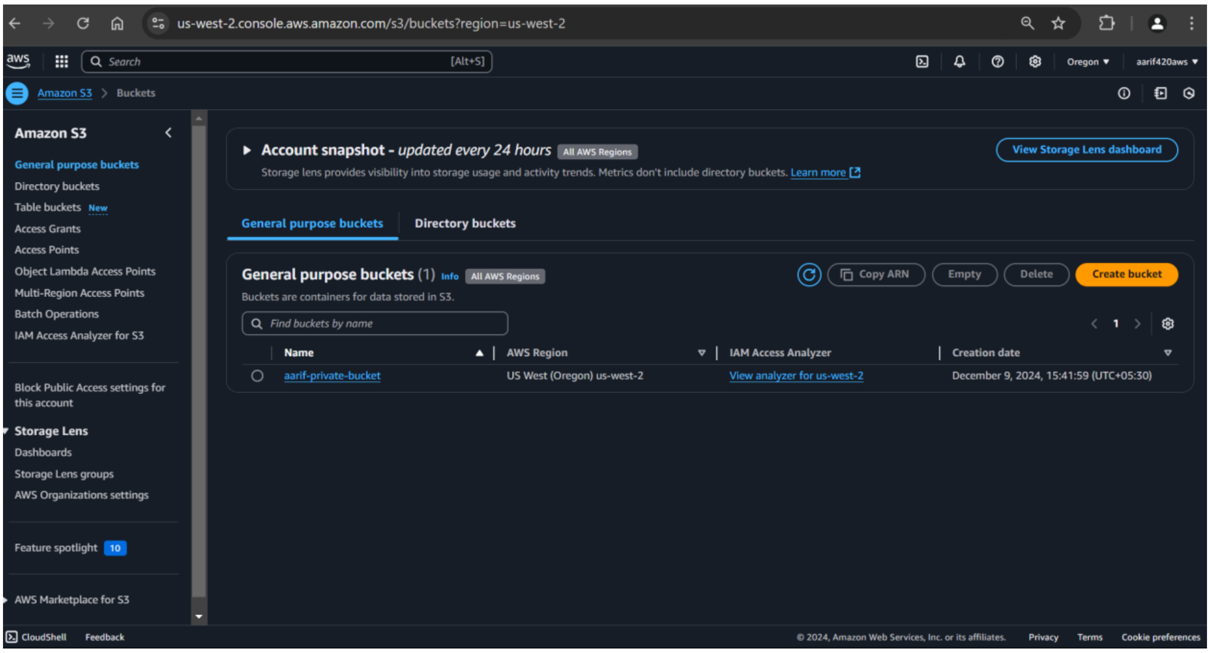
Policies (1314) Info

A policy is an object in AWS that defines permissions.

Filter by Type: All types

	Policy name	Type	Used as	Description
○	aarif-s3-access	Customer managed	Permissions policy (1)	Full access to the S3 bucket
○	AccessAnalyzerSer...	AWS managed	None	Allow Access Analyzer to analyze resou
○	AdministratorAccess	AWS managed - job fu...	Permissions policy (1)	Provides full access to AWS services an
○	AdministratorAcce...	AWS managed	None	Grants account administrative permiss
○	AdministratorAcce...	AWS managed	None	Grants account administrative permiss
○	AIOpsAssistantPolicy	AWS managed	None	Provides ReadOnly permissions requir.
○	AIOpsConsoleAdmi...	AWS managed	None	Grants full access to Amazon AI Opera
○	AIOpsOperatorAcc...	AWS managed	None	Grants access to the Amazon AI Opera

S3 Bucket:



us-west-2.console.aws.amazon.com/s3/buckets?region=us-west-2

Account snapshot - *updated every 24 hours* All AWS Regions

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. Learn more

General purpose buckets (1) Info All AWS Regions

Buckets are containers for data stored in S3.

	Name	AWS Region	IAM Access Analyzer	Creation date
○	aarif-private-bucket	US West (Oregon) us-west-2	View analyzer for us-west-2	December 9, 2024, 15:41:59 (UTC+05:30)

Reference:

1) **AWS Official Documentation –**

   https://docs.aws.amazon.com/

2) **Terraform Official Documentation for AWS –**

   https://registry.terraform.io/providers/hashicorp/aws/latest/docs

3) **Github Documentation –**

   https://docs.github.com/en

4) **Code Build –**

   https://docs.aws.amazon.com/codebuild/

Conclusion:

In this use case, we successfully designed and provisioned a multi-tiered web application environment in AWS using Terraform. The infrastructure stack includes an Autoscaling Group (ASG) in a public subnet with instances running across multiple Availability Zones (AZs) and a single EC2 instance in a private subnet. We implemented security groups to ensure that only necessary traffic is allowed, enhancing the security of our application. The setup includes an Application Load Balancer (ALB) directing traffic to the ASG and a Network Load Balancer (NLB) directing traffic to the EC2 instance in the private subnet. We also created an S3 bucket with versioning enabled and ensured it is not publicly accessible. An IAM role with full access to the S3 bucket was attached to the instances in the ASG. Finally, we stored the Terraform template in an Github repository and created a build using AWS CodeBuild to automate the provisioning of the infrastructure. This comprehensive approach ensures a scalable, secure, and efficient deployment of the web application.