NodeJS Application on EC2 Instances Using GitHub Workflows Pipeline and Terraform IaC (Task 14)

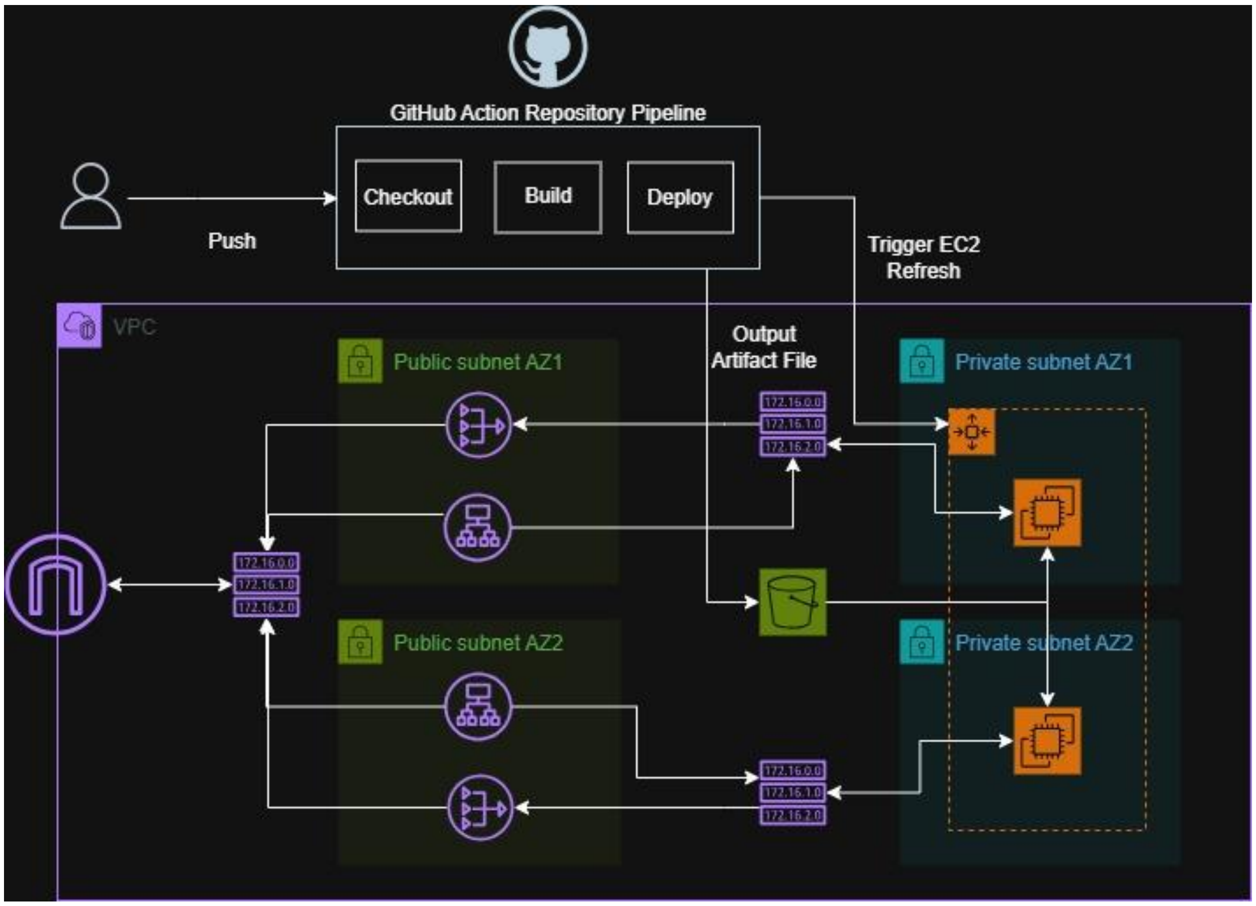
Zaeem Attique Ashar

Cloud Intern

Task Description:

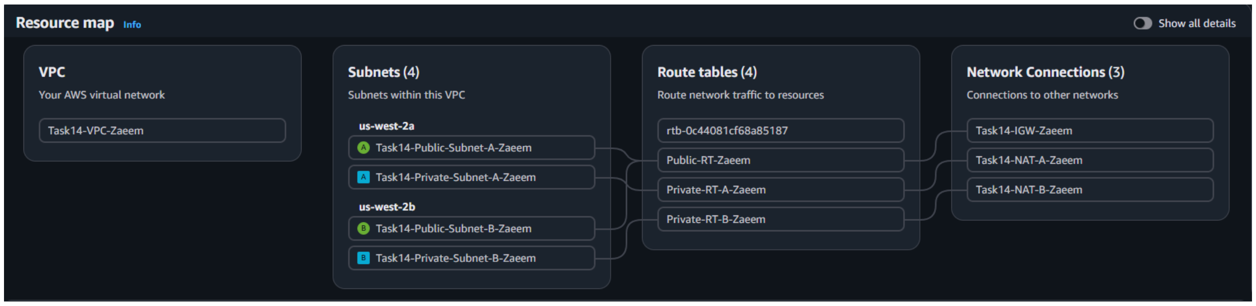
This project involves deploying a simple Node.js application using AWS EC2 Instances that are being scaled by an Auto Scaling Group. The pipeline builds the application and outputs it into a zip artifact stored in an S3 bucket. It then triggers the ASG to renew the instances with 50% always being available. The application deployment is handled by the PM2 on the instances. Traffic will be routed through the ALB deployed in front of the EC2 instances. This helps us achieve HA and rolling updates.

Architecture Diagram:



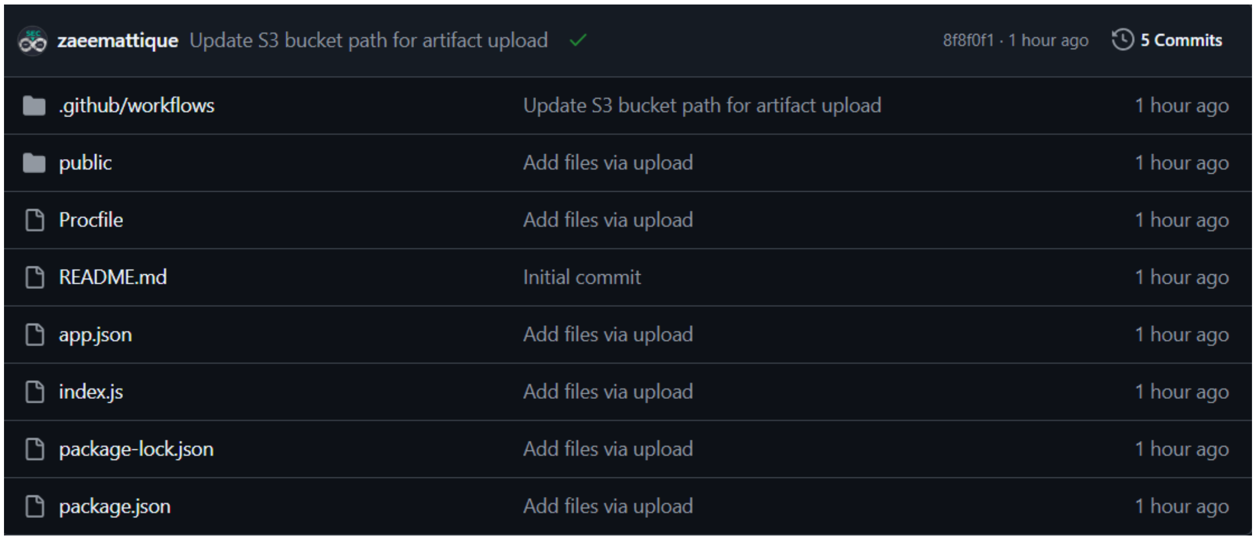
Task14.1: Create basic networking infrastructure

- Create and configure a VPC
  - CIDR Block: 10.0.0.0/16
- Create and configure Subnets
  - Public Subnet A (us-west-2a), CIDR: 10.0.1.0/24
  - Private Subnet A (us-west-2a), CIDR: 10.0.2.0/24
  - Public Subnet B (us-west-2b), CIDR: 10.0.3.0/24
  - Private Subnet A (us-west-2a), CIDR: 10.0.4.0/24
- Create and configure NAT Gateways
  - NAT Gateway A in Public Subnet A
  - NAT Gateway B in Public Subnet B
- Create and configure Internet Gateway
  - Create and attach to the project's VPC
- Create and configure Route Tables
  - Public Route Table, Outbound rule: 0.0.0.0/0 -> IGW, attach to Public SN A&B
  - Private Route Table A, Outbound Rule: 0.0.0.0/0 -> NGW attach to Private SN A
  - Private Route Table B, Outbound Rule: 0.0.0.0/0 -> NGW attach to Private SN B



Task14.2: Upload the Application Source Code to Repo

- This source code will be used by the pipeline to install dependencies, build the application, and package it into an artifact file.

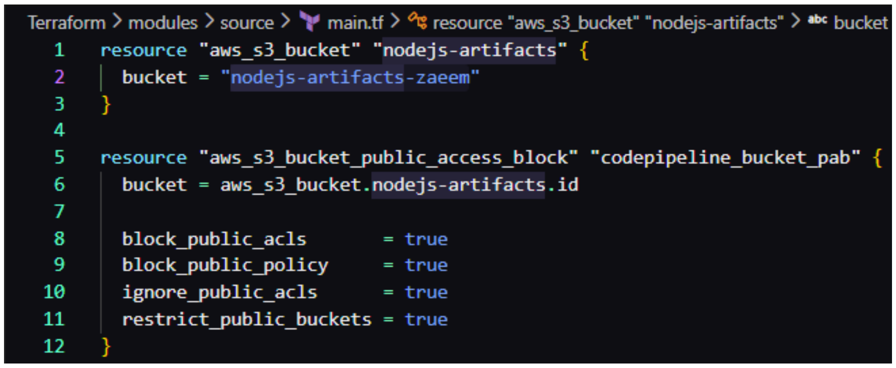


zaeemattique Update S3 bucket path for artifact upload ✓ 8f8f0f1 · 1 hour ago 5 Commits

.github/workflows	Update S3 bucket path for artifact upload	1 hour ago
public	Add files via upload	1 hour ago
Procfile	Add files via upload	1 hour ago
README.md	Initial commit	1 hour ago
app.json	Add files via upload	1 hour ago
index.js	Add files via upload	1 hour ago
package-lock.json	Add files via upload	1 hour ago
package.json	Add files via upload	1 hour ago

Task14.3: Create an S3 Bucket for the Source Artifact

- Configure the S3 bucket with the following configuration:
  - Name: nodejs-artifact-bucket
  - Block public ACLs
  - Block Public Policy
  - Ignore public ACL
  - Restrict public buckets

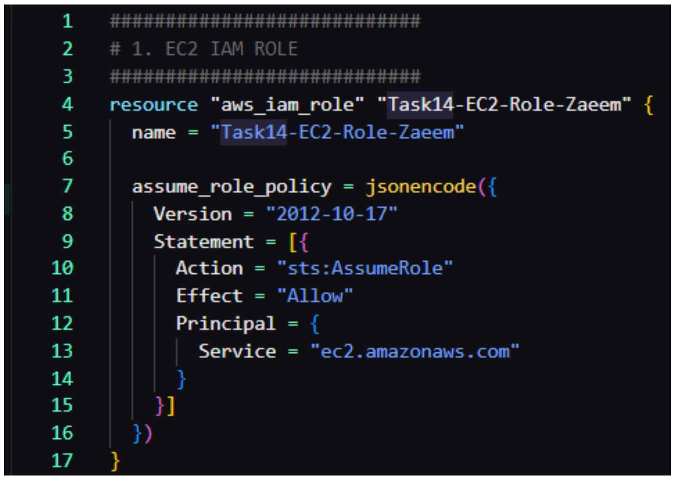


Terraform > modules > source > main.tf > resource "aws_s3_bucket" "nodejs-artifacts" > bucket

```
resource "aws_s3_bucket" "nodejs-artifacts" {
  bucket = "nodejs-artifacts-zaeem"
}

resource "aws_s3_bucket_public_access_block" "codepipeline_bucket_pab" {
  bucket = aws_s3_bucket.nodejs-artifacts.id

  block_public_acls       = true
  block_public_policy     = true
  ignore_public_acls      = true
  restrict_public_buckets = true
}
```

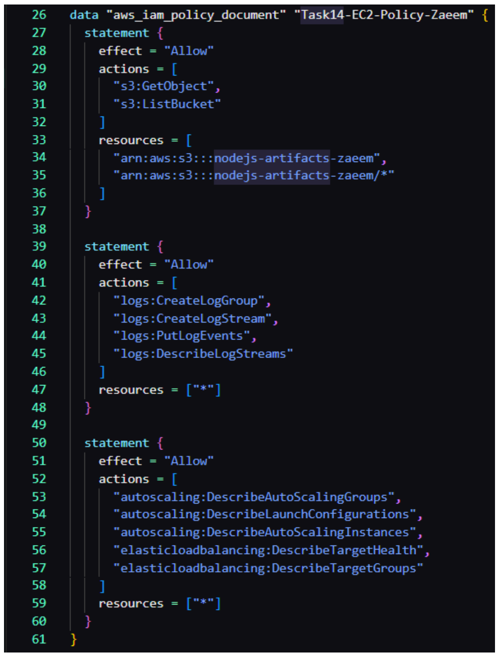
Task14.4: Create IAM Policies for the Resources

- Create and configure an IAM role for the EC2 instance that:
  - Allows EC2 to assume the role



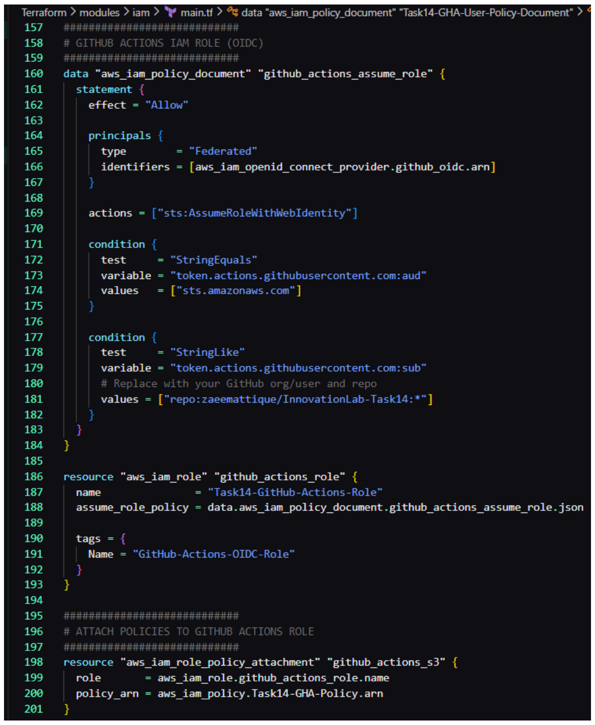
```
############################
# 1. EC2 IAM ROLE
############################
resource "aws_iam_role" "Task14-EC2-Role-Zaeem" {
  name = "Task14-EC2-Role-Zaeem"

  assume_role_policy = jsonencode({
    Version = "2012-10-17"
    Statement = [{
      Action = "sts:AssumeRole"
      Effect = "Allow"
      Principal = {
        Service = "ec2.amazonaws.com"
      }
    }]
  })
}
```

- Create and configure the policy document to attach to the role that:
    - Allows access to the nodejs-artifact-zaeem bucket for source code
    - Allows access to create/describe log groups, streams, events
    - Allows access to describe ASG, Launch config, Target Health, TG



```
data "aws_iam_policy_document" "Task14-EC2-Policy-Zaeem" {
  statement {
    effect = "Allow"
    actions = [
      "s3:GetObject",
      "s3:ListBucket"
    ]
    resources = [
      "arn:aws:s3:::nodejs-artifacts-zaeem",
      "arn:aws:s3:::nodejs-artifacts-zaeem/*"
    ]
  }

  statement {
    effect = "Allow"
    actions = [
      "logs:CreateLogGroup",
      "logs:CreateLogStream",
      "logs:PutLogEvents",
      "logs:DescribeLogStreams"
    ]
    resources = ["*"]
  }

  statement {
    effect = "Allow"
    actions = [
      "autoscaling:DescribeAutoScalingGroups",
      "autoscaling:DescribeLaunchConfigurations",
      "autoscaling:DescribeAutoScalingInstances",
      "elasticloadbalancing:DescribeTargetHealth",
      "elasticloadbalancing:DescribeTargetGroups"
    ]
    resources = ["*"]
  }
}
```

- Create and configure the OIDC Role and allow it to be assumed by GitHub Actions

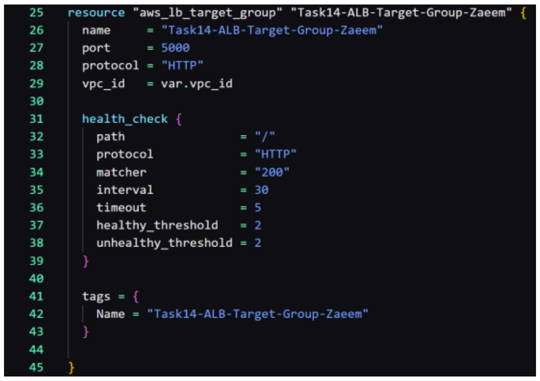


Terraform > modules > iam > main.tf > data "aws_iam_policy_document" "Task14-GHA-User-Policy-Document" >

```
############################
# GITHUB ACTIONS IAM ROLE (OIDC)
############################
data "aws_iam_policy_document" "github_actions_assume_role" {
  statement {
    effect = "Allow"

    principals {
      type        = "Federated"
      identifiers = [aws_iam_openid_connect_provider.github_oidc.arn]
    }

    actions = ["sts:AssumeRoleWithWebIdentity"]

    condition {
      test     = "StringEquals"
      variable = "token.actions.githubusercontent.com:aud"
      values   = ["sts.amazonaws.com"]
    }

    condition {
      test     = "StringLike"
      variable = "token.actions.githubusercontent.com:sub"
      # Replace with your GitHub org/user and repo
      values = ["repo:zaeemattique/InnovationLab-Task14:*"]
    }
  }
}

resource "aws_iam_role" "github_actions_role" {
  name               = "Task14-GitHub-Actions-Role"
  assume_role_policy = data.aws_iam_policy_document.github_actions_assume_role.json

  tags = {
    Name = "GitHub-Actions-OIDC-Role"
  }
}

############################
# ATTACH POLICIES TO GITHUB ACTIONS ROLE
############################
resource "aws_iam_role_policy_attachment" "github_actions_s3" {
  role       = aws_iam_role.github_actions_role.name
  policy_arn = aws_iam_policy.Task14-GHA-Policy.arn
}
```

Task14.5: Create ALB, Target Group and Listener

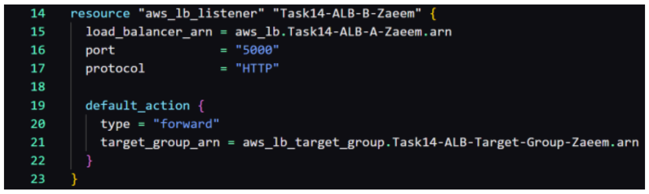
- Create Target group with the following configuration:
  - Name: Task14-ALB-Target-Group-Zaeem
  - Port: 5000
  - Protocol: HTTP
  - VPC ID: of the Task 14 VPC
  - Configure default health check



```
resource "aws_lb_target_group" "Task14-ALB-Target-Group-Zaeem" {
  name     = "Task14-ALB-Target-Group-Zaeem"
  port     = 5000
  protocol = "HTTP"
  vpc_id   = var.vpc_id

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

  tags = {
    Name = "Task14-ALB-Target-Group-Zaeem"
  }

}
```

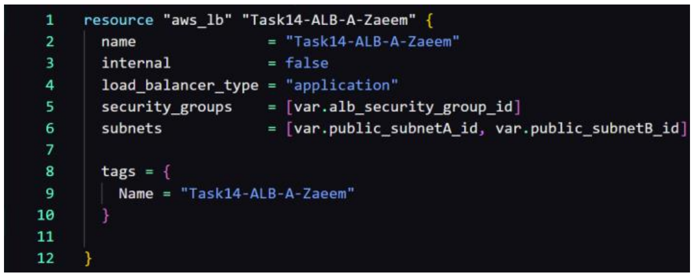
- Create Listener with the following configurations:
  - Port: 5000
  - Protocol: HTTP
  - Default Action: Forward request

- To: Target group



```
resource "aws_lb_listener" "Task14-ALB-B-Zaeem" {
  load_balancer_arn = aws_lb.Task14-ALB-A-Zaeem.arn
  port              = "5000"
  protocol          = "HTTP"

  default_action {
    type = "forward"
    target_group_arn = aws_lb_target_group.Task14-ALB-Target-Group-Zaeem.arn
  }
}
```

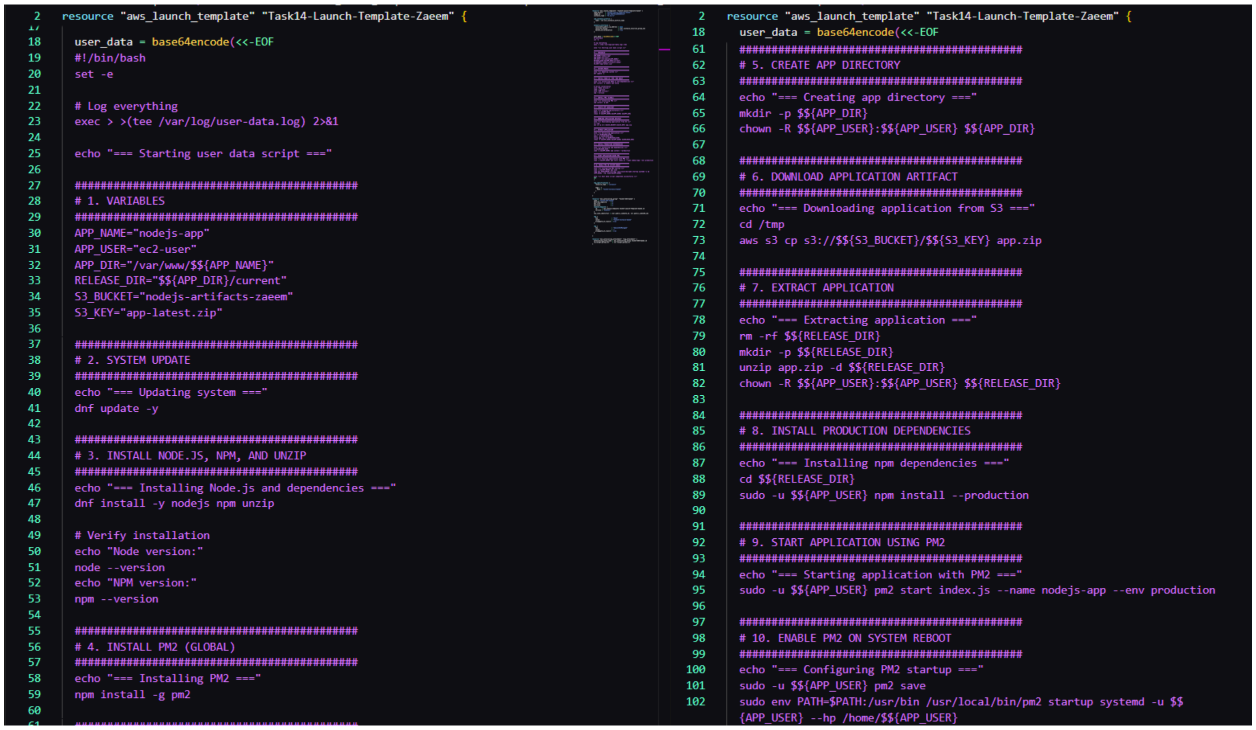
- Create ALB with the following configuration:
  - Name: Task14-ALB-Zaeem
  - Internet facing
  - Default Health Check configuration



```
resource "aws_lb" "Task14-ALB-A-Zaeem" {
  name               = "Task14-ALB-A-Zaeem"
  internal           = false
  load_balancer_type = "application"
  security_groups    = [var.alb_security_group_id]
  subnets            = [var.public_subnetA_id, var.public_subnetB_id]

  tags = {
    Name = "Task14-ALB-A-Zaeem"
  }

}
```

Task14.6: Create Launch Template and ASG

- Create and configure the launch template with the following:
    - Name prefix: Task14-Launch-Template-Zaeem
    - Image ID: Amazon Linux 2023
    - Instance Type: t3.micro
    - Attach the EC2 Instance role previously created
    - Attach the EC2 security group
    - Deny Public IP Allocation
    - Use a user data script that will:
        - Update the repo using yum
        - Install nodejs, npm, unzip
        - Install PM2 using NPM
        - Create and own App directory
        - Download the application from S3 to EC2
        - Extract the ZIP artifact
        - Install app dependencies using npm
        - Start the application using PM2
        - Enable PM2 on system reboot to start automatically

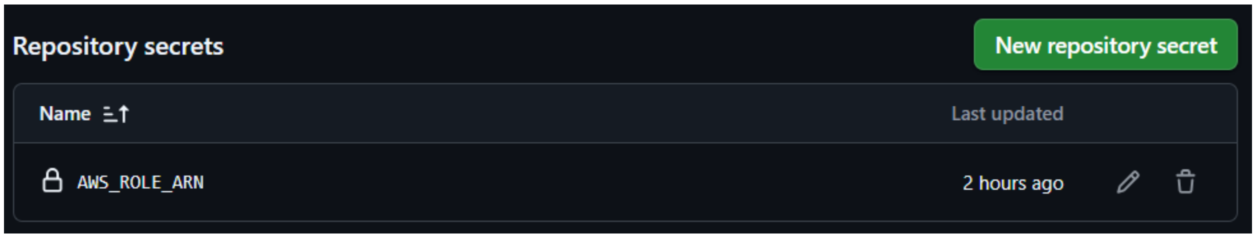


```
resource "aws_launch_template" "Task14-Launch-Template-Zaeem" {
  user_data = base64encode(<<-EOF
  #!/bin/bash
  set -e

  # Log everything
  exec > >(tee /var/log/user-data.log) 2>&1

  echo "=== Starting user data script ==="

  ############################################
  # 1. VARIABLES
  ############################################
  APP_NAME="nodejs-app"
  APP_USER="ec2-user"
  APP_DIR="/var/www/$${APP_NAME}"
  RELEASE_DIR="$${APP_DIR}/current"
  S3_BUCKET="nodejs-artifacts-zaeem"
  S3_KEY="app-latest.zip"

  ############################################
  # 2. SYSTEM UPDATE
  ############################################
  echo "=== Updating system ==="
  dnf update -y

  ############################################
  # 3. INSTALL NODE.JS, NPM, AND UNZIP
  ############################################
  echo "=== Installing Node.js and dependencies ==="
  dnf install -y nodejs npm unzip

  # Verify installation
  echo "Node version:"
  node --version
  echo "NPM version:"
  npm --version

  ############################################
  # 4. INSTALL PM2 (GLOBAL)
  ############################################
  echo "=== Installing PM2 ==="
  npm install -g pm2
```

```
resource "aws_launch_template" "Task14-Launch-Template-Zaeem" {
  user_data = base64encode(<<-EOF
  ############################################
  # 5. CREATE APP DIRECTORY
  ############################################
  echo "=== Creating app directory ==="
  mkdir -p $${APP_DIR}
  chown -R $${APP_USER}:$${APP_USER} $${APP_DIR}

  ############################################
  # 6. DOWNLOAD APPLICATION ARTIFACT
  ############################################
  echo "=== Downloading application from S3 ==="
  cd /tmp
  aws s3 cp s3://$${S3_BUCKET}/$${S3_KEY} app.zip

  ############################################
  # 7. EXTRACT APPLICATION
  ############################################
  echo "=== Extracting application ==="
  rm -rf $${RELEASE_DIR}
  mkdir -p $${RELEASE_DIR}
  unzip app.zip -d $${RELEASE_DIR}
  chown -R $${APP_USER}:$${APP_USER} $${RELEASE_DIR}

  ############################################
  # 8. INSTALL PRODUCTION DEPENDENCIES
  ############################################
  echo "=== Installing npm dependencies ==="
  cd $${RELEASE_DIR}
  sudo -u $${APP_USER} npm install --production

  ############################################
  # 9. START APPLICATION USING PM2
  ############################################
  echo "=== Starting application with PM2 ==="
  sudo -u $${APP_USER} pm2 start index.js --name nodejs-app --env production

  ############################################
  # 10. ENABLE PM2 ON SYSTEM REBOOT
  ############################################
  echo "=== Configuring PM2 startup ==="
  sudo -u $${APP_USER} pm2 save
  sudo env PATH=$PATH:/usr/bin /usr/local/bin/pm2 startup systemd -u $$
  {APP_USER} --hp /home/$${APP_USER}
```

Task14.6: GitHub Workflow Configuration

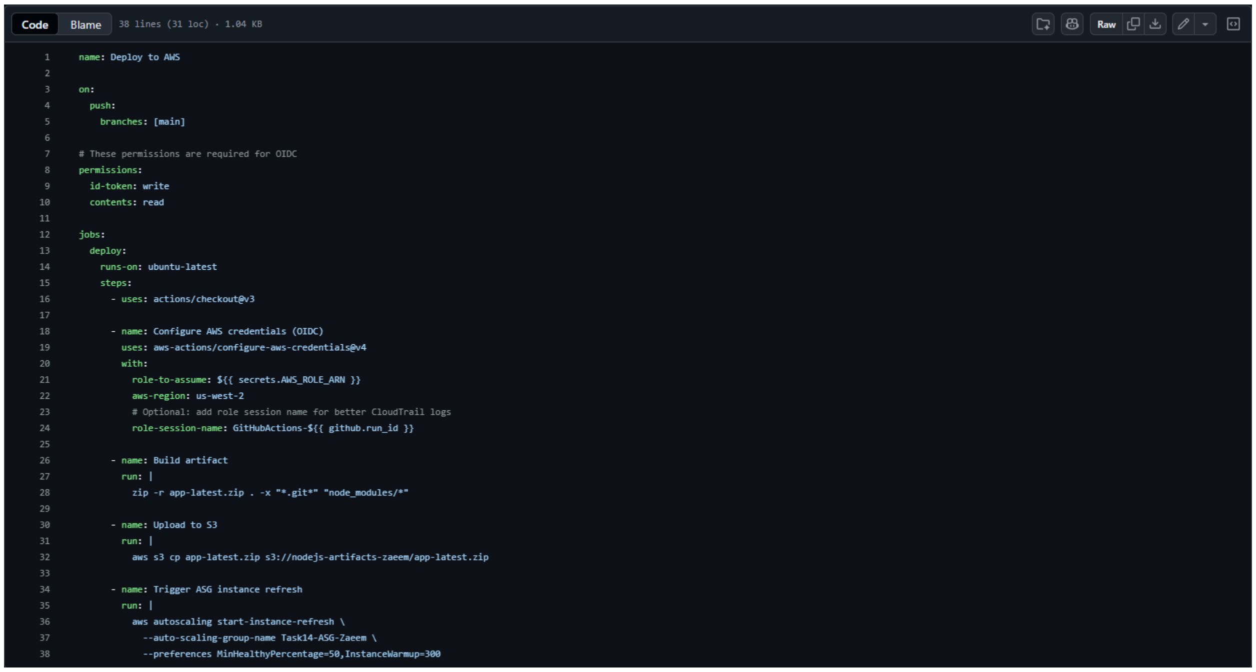
- Head over to github repository settings and add a secret which is the arn of the github actions role which will be used to run the pipeline.



Repository secrets

Name	Last updated
AWS_ROLE_ARN	2 hours ago

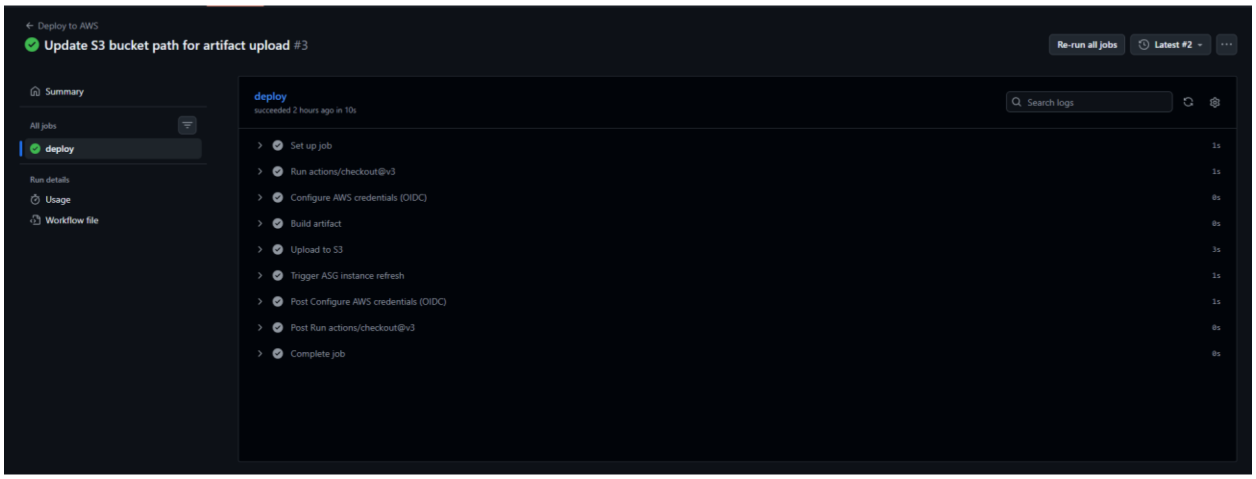
- Configure the workflow pipeline with the following steps:
  - Name: .github/workflows/deploy.yml
  - Deploy on push to main branch
  - Read and write permissions for OIDC
  - Runs on the ubuntu latest runner
  - Checks out the code to copy it to the runner
  - Assume OIDC role we created
  - Build artifacts: Zip the complete application code to artifact
  - Upload the artifact to S3 bucket
  - Trigger ASG instance refresh



```yaml
name: Deploy to AWS

on:
  push:
    branches: [main]

# These permissions are required for OIDC
permissions:
  id-token: write
  contents: read

jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3

      - name: Configure AWS credentials (OIDC)
        uses: aws-actions/configure-aws-credentials@v4
        with:
          role-to-assume: ${{ secrets.AWS_ROLE_ARN }}
          aws-region: us-west-2
          # Optional: add role session name for better CloudTrail logs
          role-session-name: GitHubActions-${{ github.run_id }}

      - name: Build artifact
        run: |
          zip -r app-latest.zip . -x "*.git*" "node_modules/*"

      - name: Upload to S3
        run: |
          aws s3 cp app-latest.zip s3://nodejs-artifacts-zaeem/app-latest.zip

      - name: Trigger ASG instance refresh
        run: |
          aws autoscaling start-instance-refresh \
            --auto-scaling-group-name Task14-ASG-Zaeem \
            --preferences MinHealthyPercentage=50,InstanceWarmup=300
```

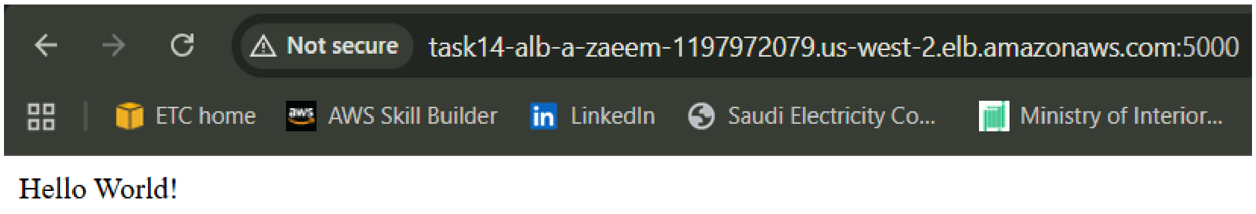
Task14.7: Testing and Checking Deployment

- Firstly, we can check the Workflow that it ran successfully:



```
← Deploy to AWS
Update S3 bucket path for artifact upload #3

deploy
succeeded 2 hours ago in 10s

> Set up job                                   1s
> Run actions/checkout@v3                      1s
> Configure AWS credentials (OIDC)             0s
> Build artifact                               0s
> Upload to S3                                 3s
> Trigger ASG instance refresh                 1s
> Post Configure AWS credentials (OIDC)        1s
> Post Run actions/checkout@v3                 0s
> Complete job                                 0s
```

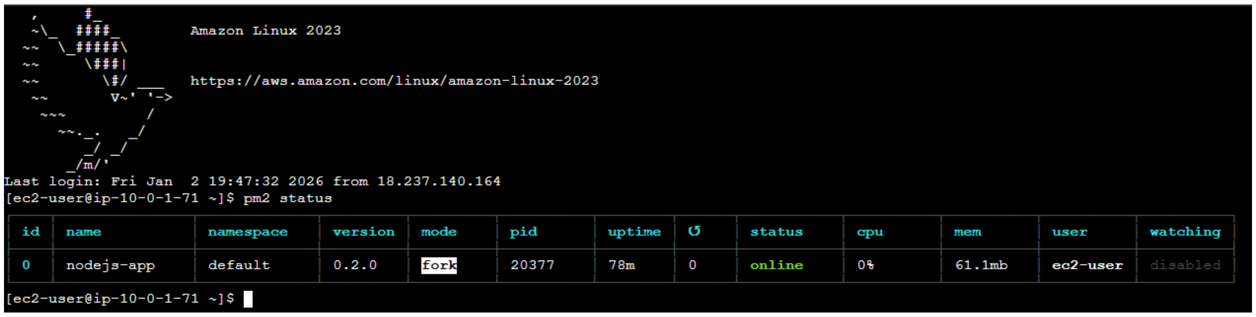
- Secondly, we can access the application via ALB Domain:



Not secure task14-alb-a-zaeem-1197972079.us-west-2.elb.amazonaws.com:5000

Hello World!

- To get deeper we can log into the EC2 instance and check PM2



```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Fri Jan  2 19:47:32 2026 from 18.237.140.164
[ec2-user@ip-10-0-1-71 ~]$ pm2 status
┌────┬──────────────┬─────────────┬─────────┬─────────┬──────────┬────────┬──────┬───────────┬──────────┬──────────┬──────────┬──────────┐
│ id │ name         │ namespace   │ version │ mode    │ pid      │ uptime │ ↺    │ status    │ cpu      │ mem      │ user     │ watching │
├────┼──────────────┼─────────────┼─────────┼─────────┼──────────┼────────┼──────┼───────────┼──────────┼──────────┼──────────┼──────────┤
│ 0  │ nodejs-app   │ default     │ 0.2.0   │ fork    │ 20377    │ 78m    │ 0    │ online    │ 0%       │ 61.1mb   │ ec2-user │ disabled │
└────┴──────────────┴─────────────┴─────────┴─────────┴──────────┴────────┴──────┴───────────┴──────────┴──────────┴──────────┴──────────┘
[ec2-user@ip-10-0-1-71 ~]$ █
```