

DevOps Assignment Project Documentation

Project Overview

This project demonstrates a full-stack web application deployed using modern DevOps practices. The stack includes a **FastAPI** backend and a **Next.js** frontend, both containerized with Docker, tested automatically, and deployed on **AWS ECS (Fargate)** using **Terraform** and **GitHub Actions CI/CD pipelines**.

Project Structure

```
DevOps-Assignment/
├── backend/                # FastAPI backend service
│   ├── app/                # Main application code
│   │   └── main.py          # API endpoints
│   ├── tests/              # Unit tests
│   │   └── test_main.py
│   ├── Dockerfile          # Multi-stage Docker build
│   └── requirements.txt     # Python dependencies
├── frontend/              # Next.js frontend application
│   ├── pages/              # UI pages
│   │   └── index.js
│   ├── __tests__/          # E2E tests
│   │   └── index.test.js
│   ├── Dockerfile          # Frontend Docker build
│   ├── package.json        # Node.js dependencies
│   └── .env.local           # Environment config
├── terraform/             # Infrastructure as Code
│   ├── main.tf             # Main Terraform config
│   ├── variables.tf        # Input variables
│   ├── outputs.tf          # Output values
│   ├── terraform.tfvars    # Actual values for variables
│   └── provider.tf         # AWS provider config
└── .github/
    └── workflows/          # CI/CD pipelines
        ├── ci.yml          # CI: tests & build on develop
        └── cd.yml           # CD: deploy on main
```

Technologies Used

- **Frontend:** Next.js (React), JavaScript
- **Backend:** FastAPI, Python
- **Containerization:** Docker
- **CI/CD:** GitHub Actions
- **Cloud:** AWS ECS (Fargate), ECR, ALB, CloudWatch
- **IaC:** Terraform
- **Monitoring:** CloudWatch Container Insights

Testing

- **Backend:** Unit tests with `pytest`
- **Frontend:** E2E tests using `@testing-library/react`
- All tests run on each push via GitHub Actions

CI/CD Pipelines

- **CI (on `develop`):**
 - Checkout code
 - Run backend & frontend tests
 - Build Docker images
 - Push to AWS ECR with `GIT_SHA` and `latest` tags

- **CD (on `main`):**
 - Deploy new Docker images to ECS services
-

Infrastructure (Terraform)

- **Network:**
 - VPC with public subnets
 - Internet gateway and route tables
 - Security groups for ALB and ECS tasks
 - **Compute:**
 - ECS Cluster (Fargate launch type)
 - Services for frontend and backend
 - **Load Balancing:**
 - Application Load Balancer with path-based routing
 - `/api/*` to backend, `/` to frontend
 - **IAM Roles:**
 - Least privilege task execution roles
-

Monitoring & Alerting

- Enabled **CloudWatch Container Insights**
 - Dashboard shows CPU, memory, request metrics
 - Alarm: Send email if CPU > 70% for 5 mins
-



Security Practices

- IAM roles with minimum required policies
 - Security groups to allow inbound HTTP only on necessary ports
 - Secrets stored in `.env.local` (local dev only)
-



Failover Demonstration

- Deployed 2+ tasks for frontend and backend
 - Manually stopped 1 backend task
 - ALB automatically rerouted traffic to healthy task
 - Verified app remained functional
-



Live URLs

- **Frontend:** <http://devops-alb-1451775341.us-east-1.elb.amazonaws.com>
 - **Backend:** <http://devops-alb-1451775341.us-east-1.elb.amazonaws.com/api/message>
-



Final Notes

This project showcases end-to-end DevOps skills, including containerization, cloud infrastructure automation, CI/CD, health monitoring, and failover. Ready for production-grade enhancements like HTTPS, auto-scaling, and domain management.



Demo Video

<https://drive.google.com/file/d/1lftb5vejOebxcZw6vrq2UhKX70SKiFN/view?usp=sharing>



Submitted by

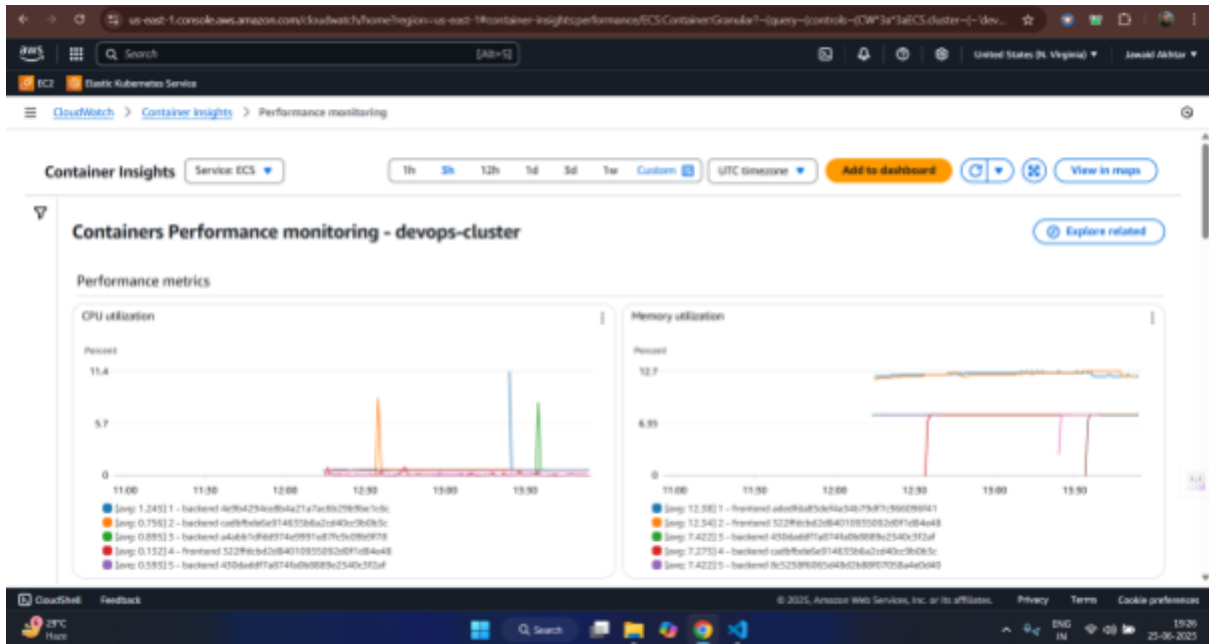
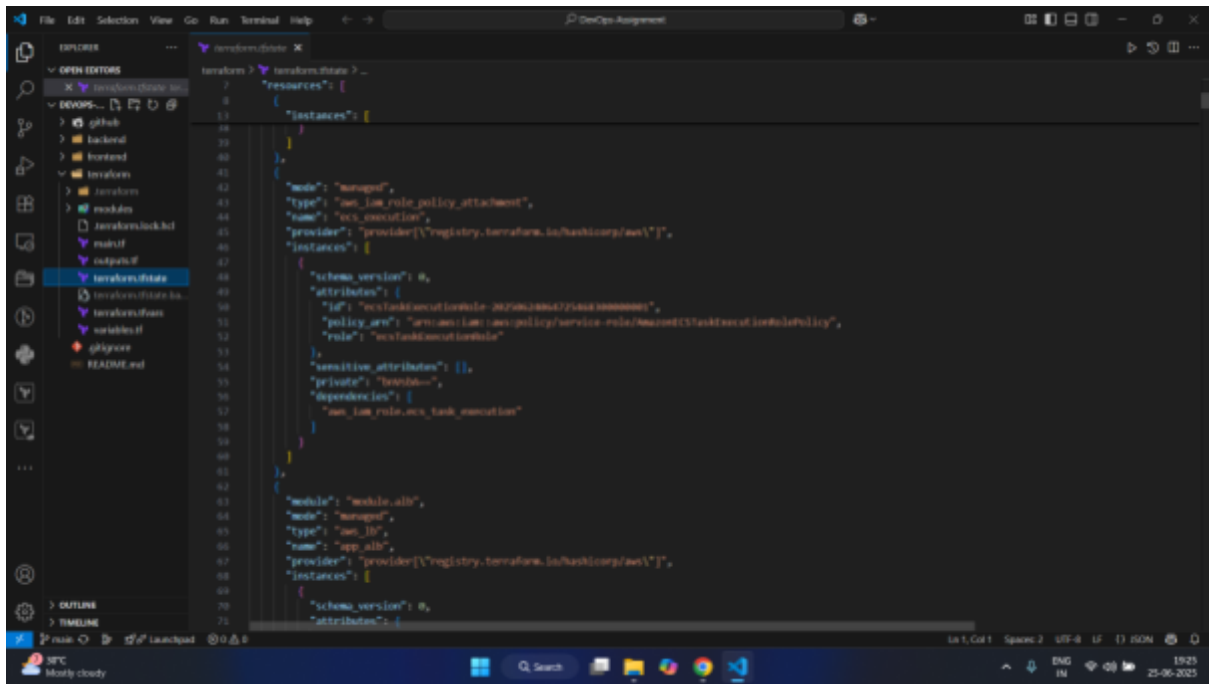
Jawaid Akhtar

DevOps Engineer

The screenshot displays the Amazon ECS console for the 'devops-cluster' in the 'us-east-1' region. The cluster overview shows it is 'Active' with 2 services, 2 pending tasks, and 4 running tasks. The 'Services' tab is selected, showing a list of services: 'backend-service' and 'frontend-service', both in 'Active' status with 2/2 tasks running.

Service name	ARN	Status	Service...	Created at	Deployments and tasks
backend-service	arn:aws:ecs:us-east-1:116981802852:cluster/devops-cluster:service/backend-service	Active	REPLICA	yesterday	2/2 Tasks running
frontend-service	arn:aws:ecs:us-east-1:116981802852:cluster/devops-cluster:service/frontend-service	Active	REPLICA	yesterday	2/2 Tasks running

```
1 terraform {
2   version = "1.10.0"
3   provider "aws" {
4     profile = "default"
5     region = "us-east-1"
6   }
7 }
8
9 resource "aws_ecs_task_definition" "task_definition" {
10   family = "ecs-task-definition"
11   task_role_arn = "arn:aws:iam::116981802852:role/ecsTaskDefinitionRole"
12   network_mode = "awsvpc"
13   execution_role_arn = "arn:aws:iam::116981802852:role/ecsTaskDefinitionRole"
14   container_definitions = jsonencode(
15     [
16       {
17         name = "backend-service",
18         image = "amazon/aws-ecs-ami:latest",
19         portMappings = [
20           { containerPort = 8080, hostPort = 8080 }
21         ],
22         environment = [
23           { name = "ENVIRONMENT", value = "dev" }
24         ],
25         logConfiguration = {
26           logDriver = "awslogs",
27           options = {
28             "awslogs-group" = "/ecs/backend-service",
29             "awslogs-region" = "us-east-1",
30             "awslogs-stream-prefix" = "task"
31           }
32         }
33       },
34       {
35         name = "frontend-service",
36         image = "amazon/aws-ecs-ami:latest",
37         portMappings = [
38           { containerPort = 8080, hostPort = 8080 }
39         ],
40         environment = [
41           { name = "ENVIRONMENT", value = "dev" }
42         ],
43         logConfiguration = {
44           logDriver = "awslogs",
45           options = {
46             "awslogs-group" = "/ecs/frontend-service",
47             "awslogs-region" = "us-east-1",
48             "awslogs-stream-prefix" = "task"
49           }
50         }
51       }
52     ]
53   )
54 }
```



us-east-1.console.aws.amazon.com/ecr/private-registry/repositories/?region=us-east-1

Amazon ECR > Private registry > Repositories

Amazon Elastic Container Registry

- Private registry
 - Repositories
 - Features & Settings
- Public registry
 - Repositories
 - Settings

ECR public gallery [🔗](#)
Amazon ECS [🔗](#)
Amazon EKS [🔗](#)

Getting started [🔗](#)
Documentation [🔗](#)

Private repositories (2)

Search by repository substring

Repository name	URI	Created at	Tag immutability	Encryption type
devops-backend	116981802852.dkr.ecr.us-east-1.amazonaws.com/devops-backend	June 24, 2025, 16:15:15 (UTC+05.5)	Mutable	AES-256
devops-frontend	116981802852.dkr.ecr.us-east-1.amazonaws.com/devops-frontend	June 24, 2025, 16:15:40 (UTC+05.5)	Mutable	AES-256

View push commands Delete Actions Create repository

CloudShell Feedback

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

25°C Mostly cloudy

github.com/lowekMkita/DevOps-Assignment/actions/runs/1567482549/job/4475884025

Summary

- Jobs
 - build-and-test
- Run details
- Usage
- Workflow file

build-and-test

succeeded 2 hours ago in 2m 17s

Search logs

- Run backend tests 1m
- Run frontend tests 17s
- Configure AWS credentials 1m
 - Run aws-actions/configure-aws-credentials@v1
 - 1: echo
 - 2: echo
 - 3: aws-access-key-id: ***
 - 4: aws-secret-access-key: ***
 - 5: aws-region: ***
 - 6: aws-profile: sts.amazonaws.com
 - 7: echo
 - 8: AWS_PROFILE: ***
 - 9: GET_SSM: /aws-ssm/secret1/secret2/secret3
 - 10: python3 -c 'import boto3; client = boto3.client("ssm"); client.get_parameter(Name="/aws-ssm/secret1/secret2/secret3", WithDecryption=True); print(client.get_parameter_response["Parameter"]["Value"]);'
 - 11: echo
 - 12: echo
 - 13: echo
 - 14: echo
 - 15: echo
 - 16: echo
 - 17: echo
 - 18: echo
 - 19: echo
 - 20: echo
 - 21: echo
 - 22: echo
 - 23: echo
 - 24: echo
 - 25: echo
 - 26: echo
 - 27: echo
 - 28: echo
 - 29: echo
 - 30: echo
 - 31: echo
 - 32: echo
 - 33: echo
 - 34: echo
 - 35: echo
 - 36: echo
 - 37: echo
 - 38: echo
 - 39: echo
 - 40: echo
 - 41: echo
 - 42: echo
 - 43: echo
 - 44: echo
 - 45: echo
 - 46: echo
 - 47: echo
 - 48: echo
 - 49: echo
 - 50: echo
 - 51: echo
 - 52: echo
 - 53: echo
 - 54: echo
 - 55: echo
 - 56: echo
 - 57: echo
 - 58: echo
 - 59: echo
 - 60: echo
 - 61: echo
 - 62: echo
 - 63: echo
 - 64: echo
 - 65: echo
 - 66: echo
 - 67: echo
 - 68: echo
 - 69: echo
 - 70: echo
 - 71: echo
 - 72: echo
 - 73: echo
 - 74: echo
 - 75: echo
 - 76: echo
 - 77: echo
 - 78: echo
 - 79: echo
 - 80: echo
 - 81: echo
 - 82: echo
 - 83: echo
 - 84: echo
 - 85: echo
 - 86: echo
 - 87: echo
 - 88: echo
 - 89: echo
 - 90: echo
 - 91: echo
 - 92: echo
 - 93: echo
 - 94: echo
 - 95: echo
 - 96: echo
 - 97: echo
 - 98: echo
 - 99: echo
 - 100: echo
- Login to Amazon ECR 2s
 - Run aws-actions/amazon-ecr-login@v1
 - Logging into registry 116981802852.dkr.ecr.us-east-1.amazonaws.com
- Build and push backend Docker image 22s

Early dawn ahead 25°C