

# Lokalise DevOps/SRE take-home assignment

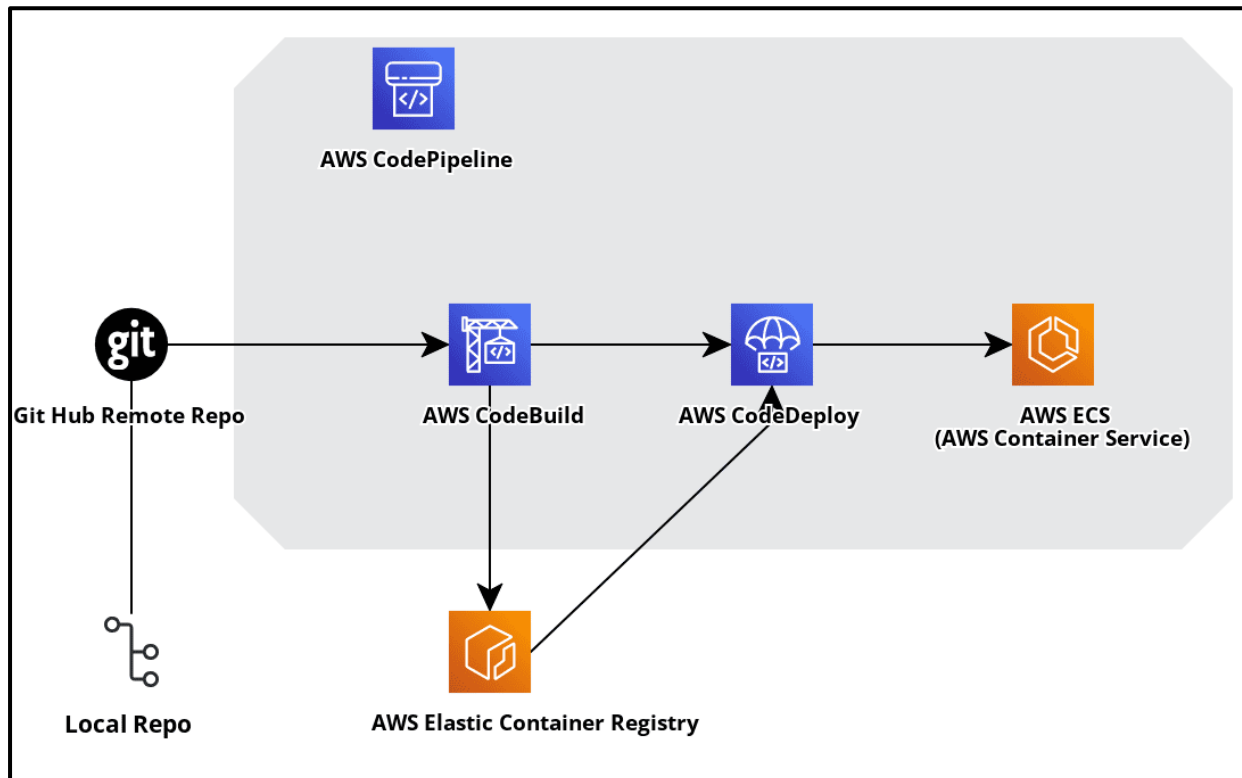
## Goal:

- To build a simple proof-of-concept of [a blue-green deployment](#) for a web application

## Result:

- Got the desired output as one of many alternatives.

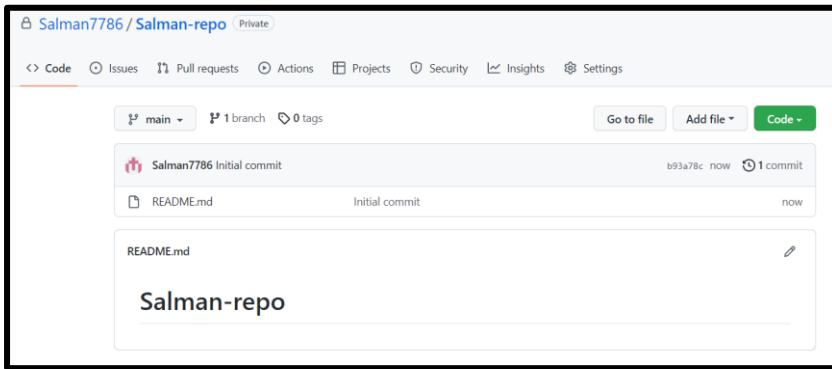
## Overview of the Solution:



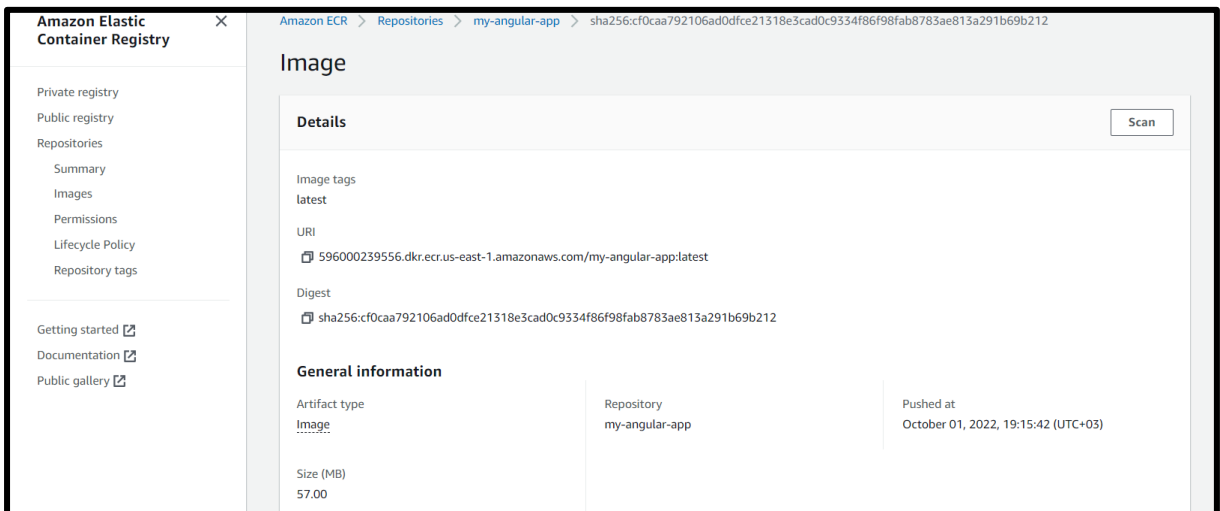
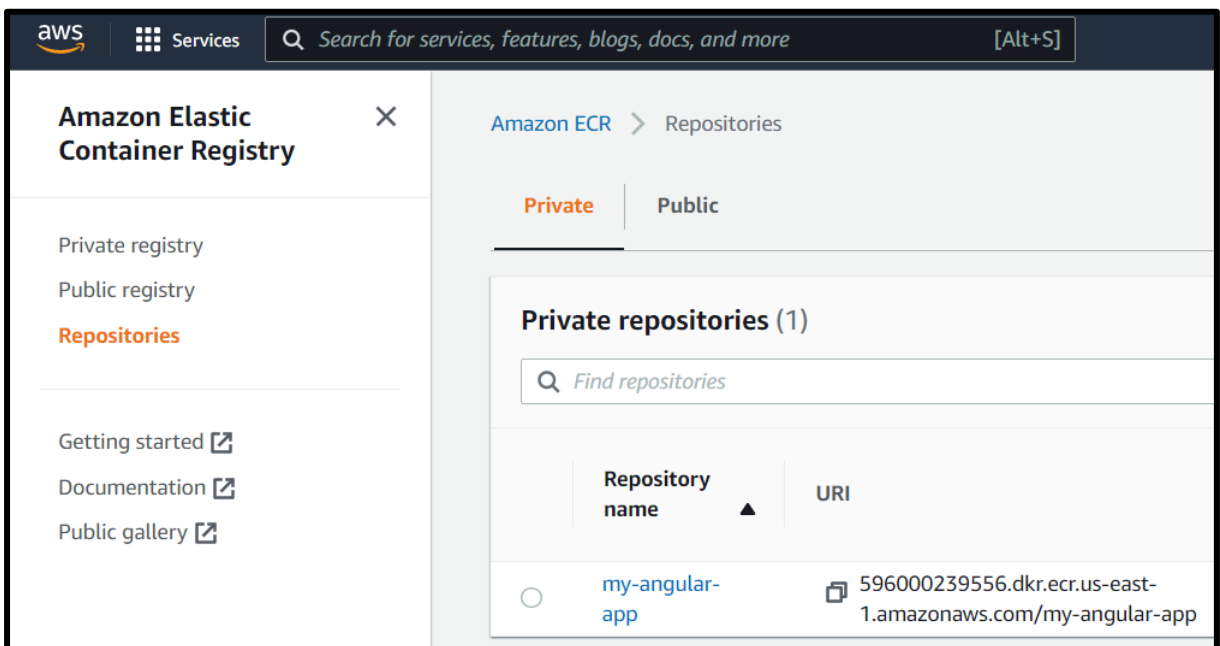
## Technology Stack used:

- AWS Cloud
- CloudFormation as IaC
- Build Docker Container image
- Elastic Container Repository and DockerHUB
- AWS Parameter Store for storing the secrets
- Container Service AWS ECS
- AWS Application Load balancer
- CICD tool used – AWS: CodePipeline, CodeBuild, CodeDeploy.
- For Blue Green deployments, Created Deployment Group with ECS Service along with ALB which routes traffic to Listener target group as per the deployment cycle.s

## 1. Create Fresh GitHub Remote Repository



## 2. AWS ECR Private registry for storing docker container Images



### 3. Git Local machine - initialization

- Inside my sample application folder
- `git init`
- `git remote add origin https://github.com/<repo-name>`
- `git add .`
- `git status`
- `git commit -m "Docker and build Spec were added."`
- `git push origin master`

### 4. ECSCloudFormation.yaml

The CloudFormation code is creating below resources.

- Using the existing VPC and two Subnet
- ECS Task Definition
- ECS Cluster
- Application Load Balancer
- Target Group for ALB
- Listener
- Security Groups for ALB and Security Group for Containers
- ECS Service.
- Using the Custom docker image from the ECR repository.

➡ At this point I can access my web application through Application Load balancer URL.  
It's showing the current version and 1.0

Clusters > MyFargateCluster

## Cluster : MyFargateCluster

Get a detailed view of the resources on your cluster.

**Cluster ARN** arn:aws:ecs:us-east-1:596000239556:cluster/MyFargateCluster

**Status** ACTIVE

**Registered container instances** 0

**Pending tasks count** 0 Fargate, 0 EC2, 0 External

**Running tasks count** 1 Fargate, 0 EC2, 0 External

**Active service count** 1 Fargate, 0 EC2, 0 External

**Draining service count** 0 Fargate, 0 EC2, 0 External

Services | Tasks | ECS Instances | Metrics | Scheduled Tasks | Tags | Capacity Providers

Create Update Delete Actions

Last updated on October 3, 2022 12:31:29 PM (0m ago)

Filter in this page Launch type ALL Service type ALL

	Service Name	Status	Service type...	Task Definiti...	Desired tas...	Running tas...	Launch typ...	Platform ver...
<input type="checkbox"/>	Demo-MyECSService-wQKfcinx9aI0	ACTIVE	REPLICA	Demo-ECST...	1	1	FARGATE	LATEST(1.4.0)

5. Create Service Link IAM Role for CodePipeline, CodeBuild, and CodeDeploy or we can dynamically use existing AWS managed policy.

## 6. CICD using CodePipeline

6.1 We need to integrate the GitHub repository with our AWS CodePipeline so we need to create a new connection in AWS console.

Developer Tools > Connections > Create connection

## Create a connection

Select a provider

☐ Bitbucket ☒ GitHub


Create GitHub App connection


Connection name


Salman-repo

Developer Tools > Connections

Connections Hosts

Connections [Info](#)  [View details](#) [Update pending connection](#) [Delete](#) [Create connection](#)

< 1 > 

	Connection name	Provider	Status	ARN
<input type="radio"/>	Salman-repo	GitHub	 Available	arn:aws:codestar-connections:us-east-1:596000239556:connection/dea780c4-7299-4cc6-9fe9-80899c9e72e2

6.2 Create CodePipeline: So that every Push (`git push origin master`) will invoke the AWS CodePipeline and trigger the subsequent event as creating docker images using CodeBuild and pushing it to docker registry. And Automated Deployment to ECS using CodeDeploy.

### Add Source Stage:

Step 1  
Choose pipeline settings

Step 2  
Add source stage

Step 3  
Add build stage

Step 4  
Add deploy stage

Step 5  
Review

## Choose pipeline settings [Info](#)

### Pipeline settings

**Pipeline name**  
Enter the pipeline name. You cannot edit the pipeline name after it is created.

No more than 100 characters

**Service role**

☒ New service role  
Create a service role in your account

☐ Existing service role  
Choose an existing service role from your account

**Role name**

Type your service role name

☒ Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

### Advanced settings

**Artifact store**

☒ Default location  
Use the default artifact store (Amazon S3 codepipeline-us-east-1-480954390210) designated in the same region and account as your pipeline

☐ Custom location  
Choose an existing S3 location from your account in the same region and account as your pipeline

**Encryption key**

☒ Default AWS Managed Key  
Use the AWS managed customer master key for CodePipeline in your account to encrypt the data in the artifact store.

☐ Customer Managed Key  
To encrypt the data in the artifact store under an AWS KMS customer managed key, specify the key ID, key ARN, or alias ARN.

Step 2

### Add source stage

Step 3

Add build stage

Step 4

Add deploy stage

Step 5

Review

## Source

### Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 2) ▼



#### New GitHub version 2 (app-based) action

To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

### Connection

Choose an existing connection that you have already configured, or create a new one and then return to this task.

🔍  ✕ or [Connect to GitHub](#)



#### Ready to connect

Your GitHub connection is ready for use.

### Repository name

Choose a repository in your GitHub account.

🔍  ✕

<account>/<repository-name>

### Branch name

Choose a branch of the repository.

🔍  ✕

### Change detection options

☒ **Start the pipeline on source code change**

Automatically starts your pipeline when a change occurs in the source code. If turned off, your pipeline only runs if you start it manually or on a schedule.

### Output artifact format

Choose the output artifact format.



#### CodePipeline default

AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include git metadata about the repository.



#### Full clone

AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full git clone. Only supported for AWS CodeBuild actions.

Cancel

Previous

Next

## Add Build Stage:

The screenshot shows the AWS CodePipeline console with the 'Add build stage' page. The left sidebar shows the 'CodePipeline' service selected. The main content area is titled 'Add build stage' and includes a 'Build - optional' section. In this section, the 'Build provider' is set to 'AWS CodeBuild', the 'Region' is 'US East (N. Virginia)', and the 'Project name' is 'DockerbuildDeploy'. A green success message states 'Successfully created DockerbuildDeploy in CodeBuild.' Below this, the 'Environment variables - optional' section is visible, showing two variables: 'DOCKERHUB\_USER' and 'DOCKERHUB\_TOKEN', both of type 'Plaintext'. At the bottom, the 'Build type' is set to 'Single build'.

Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1  
Choose pipeline settings

Step 2  
Add source stage

Step 3  
Add build stage

Step 4  
Add deploy stage

Step 5  
Review

### Add build stage [Info](#)

**Build - optional**

**Build provider**  
This is the tool of your build project. Provide build artifact details like operating system, build spec file, and output file names.

AWS CodeBuild

**Region**  
US East (N. Virginia)

**Project name**  
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.

DockerbuildDeploy or Create project

Successfully created DockerbuildDeploy in CodeBuild.

**Environment variables - optional**  
Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. [Learn more](#)

Name	Value	Type	
DOCKERHUB_USER		Plaintext	Remove
DOCKERHUB_TOKEN		Plaintext	Remove

Add environment variable

**Build type**

☒ Single build  
Triggers a single build.

☐ Batch build  
Triggers multiple builds as a single execution.

## Add CodeDeploy Stage: Create CodeDeploy Application

The screenshot shows the AWS CodeDeploy console with the 'Create application' page. The left sidebar shows the 'CodeDeploy' service selected. The main content area is titled 'Create application' and includes an 'Application configuration' section. In this section, the 'Application name' is 'fargate-blue-green-app' and the 'Compute platform' is 'Amazon ECS'.

Developer Tools > CodeDeploy > Applications > Create application

## Create application

### Application configuration

**Application name**  
Enter an application name

fargate-blue-green-app

100 character limit

**Compute platform**  
Choose a compute platform

Amazon ECS

6.3 Update a Service definition in ECS console.

Clusters > MyFargateCluster > Service: Demo-MyECSService-wQKfcinx9aI0

## Service : Demo-MyECSService-wQKfcinx9aI0 Update

Cluster	MyFargateCluster	Desired count	1
Status	ACTIVE	Pending count	0
Task definition	Demo-ECSTaskDefinition-EK0tezTol5gU:1	Running count	1
Service type	REPLICA		
Launch type	FARGATE		
Service role	AWSServiceRoleForECS		
Created By	arn:aws:iam::596000239556:root		

Details Tasks Events Auto Scaling Deployments Metrics Tags

### Load Balancing

Target Group Name	Container Name	Container Port
MyFargateTG1	my-angular-app	80

### Network Access

Health check grace period 0

Allowed VPC [vpc-d23d26a8](#)

Use below Steps:

- Launch type: FARGATE
- Operating system family: Linux
- Choose a deployment option for the service.
- Blue/green deployment

Choose a deployment option for the service.

**Deployment type\*** ☐ Rolling update ⓘ ☒ Blue/green deployment (powered by AWS CodeDeploy) ⓘ

This sets AWS CodeDeploy as the deployment controller for the service. A CodeDeploy application and deployment group are created automatically with [default settings](#) for the service. To change to the rolling update deployment type after the service has been created, you must re-create the service and select the "rolling update" deployment type.

**Deployment configuration\*** [CodeDeployDefault.ECSAllAtOnce](#) ▼

The deployment configuration specifies how traffic is shifted to the updated Amazon ECS task set. [Learn more](#)

**Service role for CodeDeploy\*** [CodeDeploy-service\\_linked-role-ECS](#) ▼

The IAM role the service uses to make API requests to authorized AWS services. Create a service role for CodeDeploy in the IAM console. [Learn more](#)

Above config sets AWS CodeDeploy as the deployment controller for the service. A CodeDeploy application and deployment group are created automatically.



Container to load balance

my-angular-app : 80

Remove ✕

Production listener port\*

80:HTTP

Production listener protocol\*

HTTP

Test listener

☒

An optional test listener is used to test the new application revision before routing traffic to it.

Test listener port\*

create new

8080

Test listener protocol\*

HTTP

Load balancer type Application Load Balancer (ALB)

Load balancer name FargateAlb

Container to load balance

Container port 80

**Production listener port 80**

Target group 1 name MyFargateTG1

Production listener path-pattern /

Production listener health-check path /

**Test listener port 8080**

Target group 2 name MyFargateTG2

Test listener path-pattern /

Test listener health-check path /

Create Service

Create service: fargate-service

✓ Service created

Service created. Tasks will start momentarily. View: [fargate-service](#)

✓ The CodeDeploy application was created.

CodeDeploy The application and deployment group was created successfully.  
CodeDeploy application: [AppECS-MyFargateCluster-fargate-service](#)  
To view or change deployment settings, go to the CodeDeploy deployment group: [DgpECS-MyFargateCluster-fargate-service](#)

The CodeDeploy application was created.

CodeDeploy The application and deployment group was created successfully.

Clusters > MyFargateCluster

Cluster : MyFargateCluster

Update Cl

Get a detailed view of the resources on your cluster.

Cluster ARN

arn:aws:ecs:us-east-1:596000239556:cluster/MyFargateCluster

Status

ACTIVE

Registered container instances

0

Pending tasks count

0 Fargate, 0 EC2, 0 External

Running tasks count

2 Fargate, 0 EC2, 0 External

Active service count

2 Fargate, 0 EC2, 0 External

Draining service count

0 Fargate, 0 EC2, 0 External

Services

Tasks

ECS Instances

Metrics

Scheduled Tasks

Tags

Capacity Providers

Create

Update

Delete

Actions

Last updated on October 3, 2022 1:10:11 f

Filter in this page

Launch type

ALL

Service type

ALL

1 selected

	Service Name	Status	Service type	Task Definition ...	Desired tasks	Running tasks	Launch type
<input type="checkbox"/>	fargate-service	ACTIVE	REPLICA	Demo-ECSTaskD...	1	1	FARGATE

Developer Tools

CodeDeploy

Developer Tools > CodeDeploy > Applications

Notify

View details

Deploy application

Create application

Applications

Q

Application name	Compute platform	Created
<input type="radio"/> AppECS-MyFargateCluster-fargate-service	Amazon ECS	4 minutes ago

## 6.4 Stage CodeDeploy

Action name

Choose a name for your action

Deploy

No more than 100 characters

Action provider

Amazon ECS (Blue/Green)

Region

US East (N. Virginia)

Input artifacts

Choose an input artifact for this action. [Learn more](#)

BuildArtifact

Add

No more than 100 characters

AWS CodeDeploy application name

Choose one of your existing applications, or create a new one in AWS CodeDeploy.

Q AppECS-MyFargateCluster-fargate-service

Create application

AWS CodeDeploy deployment group

Choose one of your existing deployment groups, or create a new one in AWS CodeDeploy.

Q DgpECS-MyFargateCluster-fargate-service

Amazon ECS task definition

Choose the input artifact where your Amazon ECS task definition file is stored. If other than the default file path, specify the path and filename of your task definition file.

BuildArtifact

taskdef.json

The default path is taskdef.json.

AWS CodeDeploy AppSpec file

Choose the input artifact where your AWS CodeDeploy AppSpec file is stored. If other than the default file path, specify the path and filename of your AppSpec file.

BuildArtifact

appspec.yaml

## 6.5 Commit the code and push

Triggerred our pipeline

The screenshot shows the AWS CodePipeline console for a pipeline named 'BuildDockerandDeployECS'. The left sidebar shows the 'CodePipeline' section with options for Source, Artifacts, Build, Deploy, Pipeline, and Settings. The main area displays the pipeline execution details, including the Source, Build, and Deploy stages. The Source stage is 'Succeeded' (12 minutes ago), the Build stage is 'Succeeded' (8 minutes ago), and the Deploy stage is 'In progress' (8 minutes ago). The pipeline execution ID is 447ab8d4-29b9-414a-b7c9-ec85849aea8.

**BuildDockerandDeployECS**

**Source** Succeeded  
Pipeline execution ID: 447ab8d4-29b9-414a-b7c9-ec85849aea8

Source  
GitHub (Version 2) [Details](#)  
Succeeded - 12 minutes ago  
151fe4de [Details](#) Source: BlueGreenTest Version 2.0

Disable transition

**Build** Succeeded  
Pipeline execution ID: 447ab8d4-29b9-414a-b7c9-ec85849aea8

Build  
AWS CodeBuild  
Succeeded - 8 minutes ago  
[Details](#)  
151fe4de [Details](#) Source: BlueGreenTest Version 2.0

Disable transition

**Deploy** In progress  
Pipeline execution ID: 447ab8d4-29b9-414a-b7c9-ec85849aea8

Deploy  
Amazon ECS (Blue/Green) [Details](#)  
In progress - 8 minutes ago  
[Details](#)  
151fe4de [Details](#) Source: BlueGreenTest Version 2.0

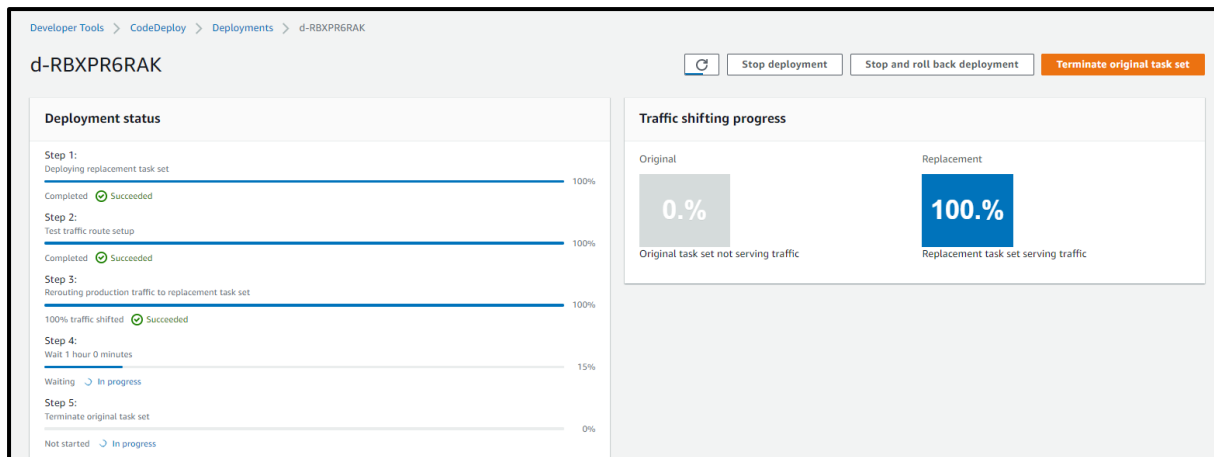
Feedback Looking for language selection? Find it in the new [Unified Settings](#)

## Deployment in Progress

Developer Tools > CodeDeploy > Deployments

Deployment history [Refresh](#) [View details](#) [Actions](#) [Copy deployment](#)

	Deployment Id	Status	Deployment type	Compute platform	Application	Deployment group	Revision location	Initiating event
<input type="radio"/>	d-RBXPGRRAK	<span>In progress</span>	Blue/green	Amazon ECS	AppECS-MyFargateCluster-fargate-service	DgpECS-MyFargateCluster-fargate-service	79b2b350bcc6799...	User action



In Deployment Stage, blue green deployment will initiate, and it will spin the additional container based on service and task definition.

And automatically it will register the Target group 2 to our Public Facing Load Balancer.

Clusters > MyFargateCluster

### Cluster : MyFargateCluster

Get a detailed view of the resources on your cluster.

Cluster ARN: am:aws:ecs:us-east-1:596000239556:cluster/MyFargateCluster

Status: ACTIVE

Registered container instances: 0

Pending tasks count: 0 Fargate, 0 EC2, 0 External

Running tasks count: 2 Fargate, 0 EC2, 0 External

Active service count: 1 Fargate, 0 EC2, 0 External

Draining service count: 0 Fargate, 0 EC2, 0 External

Services Tasks ECS Instances Metrics Scheduled Tasks Tags Capacity Providers

Create Update Delete Actions

Filter in this page Launch type: ALL Service type: ALL

<input type="checkbox"/>	Service Name	Status	Service type	Task Definition	Desired tasks	Running tasks
<input type="checkbox"/>	fargate-service	ACTIVE	REPLICA	Demo-ECSTaskDefinit...	1	2

Clusters > MyFargateCluster > Service: fargate-service

Service : fargate-service

ClusterMyFargateCluster

StatusACTIVE

Task definitionDemo-ECSTaskDefinition-EK0tezTol5gU:1

Service typeREPLICA

Launch typeFARGATE

Service roleAWSServiceRoleForECS

Created Byam.aws.iam::596000239556:root

Desired count1

Pending count0

Running count2

Details

Tasks

Events

Auto Scaling

Deployments

Metrics

Tags

Task Placement

StrategyNo strategies

ConstraintNo constraints

Blue/green deployment

Deployment IDd-RBXP6RAK

TypeBlue/green

Started byAWS CodeDeploy

StatusInProgress

CodeDeploy deployment groupDgpECS-MyFargateCluster-fargate-service

Start time2022-10-03 01:42:26

End time-

Deployment historyDgpECS-MyFargateCluster-fargate-service

Details

Tasks

Events

Auto Scaling

Deployments

Metrics

Tags

Task Placement

StrategyNo strategies

ConstraintNo constraints

Blue/green deployment

Deployment IDd-RBXP6RAK

TypeBlue/green

Started byAWS CodeDeploy

StatusInProgress

CodeDeploy deployment groupDgpECS-MyFargateCluster-fargate-service

Start time2022-10-03 01:42:26

End time-

Deployment historyDgpECS-MyFargateCluster-fargate-service

Stop and rollback deployment

Task set ID	Environment	Task set status	Traffic	Desired count	Running count	Pending count
ecs-svc/8910503786521628847	Replacement	PRIMARY	100%	1	1	0
ecs-svc/6965737905120952339	Original	ACTIVE	0%	1	1	0

**Target groups (1/3)** Info Refresh Actions Create target group

Search or filter target groups

	Name	Port	Target type	Load balancer	VPC ID
<input checked="" type="checkbox"/>	MyFargateTG1	80	IP	<a href="#">None associated</a>	vpc-d23d26a8
<input type="checkbox"/>	MyFargateTG2	80	IP	FargateAlb	vpc-d23d26a8

---

**Target group: MyFargateTG1**

Details | **Targets** | Monitoring | Health checks | Attributes | Tags

**Registered targets (1)** Refresh Deregister Register targets

Filter resources by property or value

	IP address	Port	Zone	Health status	Health status details
<input type="checkbox"/>	172.31.86.64	80	us-east-1a	⚠ unused	Target group is not configured to receive traffic from the load balancer

**Target groups (1/3)** Info Refresh Actions Create target group

Search or filter target groups

	Name	Port	Target type	Load balancer	VPC ID
<input type="checkbox"/>	MyFargateTG1	80	IP	<a href="#">None associated</a>	vpc-d23d26a8
<input checked="" type="checkbox"/>	MyFargateTG2	80	IP	FargateAlb	vpc-d23d26a8

---

**Target group: MyFargateTG2**

Details | **Targets** | Monitoring | Health checks | Attributes | Tags

**Registered targets (1)** Refresh Deregister Register targets

Filter resources by property or value

	IP address	Port	Zone	Health status	Health status details
<input type="checkbox"/>	172.31.81.112	80	us-east-1a	✅ healthy	

Application Load Balancer URL.

← → ↻ fargatealb-1250000926.us-east-1.elb.amazonaws.com ☆

# Sample Angular App for Demo Blue Green Deployment by- SALMAN SHAIKH

Version: V 2.0

Application is on latest version. Let's say We want to go back to previous state then simply **"Stop and Rollback the Deployment"**.

I will continue with the changes and terminate the old container using **"Terminate Original task set"**

