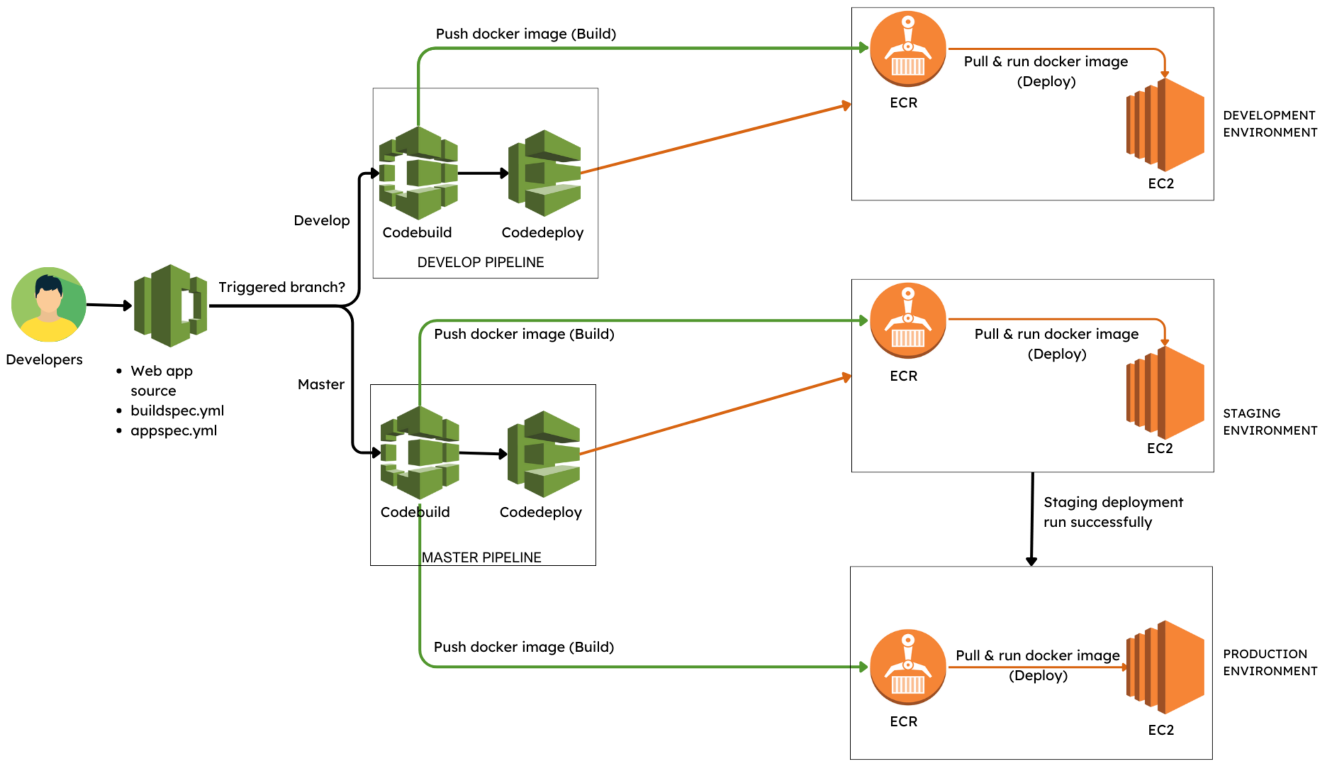
AWS CI/CD for deploying a FastAPI applcation

Goal

This is a guideline on how to setup a simple CI-CD pipeline using AWS services for deploying a FastAPI application on develop/staging/production environments.



Set up the CI/CD pipeline

1. CodeCommit:

- Create CodeCommit credentials for pushing and pulling source code.
- The repo has 2 branches: `dev` and `master`
- Directory structure:

```
.
├── app
│   └── main.py
├── appspec.yml
├── buildspec.yml
├── Dockerfile
├── .flake8
├── .gitignore
├── images
├── README.md
├── requirements.txt
├── scripts
    ├── after-install.sh
   ├── app-start.sh
   ├── app-stop.sh
   ├── push-image.sh
   └── validate.sh
```

- The repo includes:
  - FastAPI application source code.

```
# In ./app/main.py
from fastapi import FastAPI

app = FastAPI()


@app.get('/')
async def root():
    return {'greeting': 'Hello from root function'}


@app.get('/{name}')
async def hello(name: str):
    return {'greeting': f'Hello {name}!'}
```

  - CI pipeline: **buildspec.yml**

```
# In ./buildspec.yml
version: 0.2

phases:
install:
    runtime-versions:
    python: 3.11
    commands:
    # - nohup /usr/local/bin/dockerd --host=unix:///var/run/docker.sock --host=tcp://127.0.0.1:2375 --storage-driver=overlay2 &
    - timeout 15 sh -c "until docker info; do echo .; sleep 1; done"
    - pip install -r requirements.txt
pre_build:
    commands:
    - echo Pre-build phase
    - echo Logging in to Amazon ECR...
    - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --username AWS --password-stdin $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com
build:
    commands:
    - echo Build phase
    - echo Build started on `date`
    - echo Run Lint tests
    - printenv
    - flake8
    - echo "Building the Docker image (image tag -
$IMAGE_TAG)..."
    - docker build -t my-image:$IMAGE_TAG .
post_build:
    commands:
    - echo Post-build phase
    - echo Build completed on `date`
    - echo Pushing the Docker image...
    - chmod +x scripts/push-image.sh && sh scripts/push-image.sh
artifacts:
files:
    - ./appspec.yml
    - ./scripts/*
```

- CD pipeline: **appspec.yml**

```
# In ./appspec.yml
version: 0.0
os: linux
hooks:
    ApplicationStop:
        - location: scripts/app-stop.sh
        timeout: 10
        runas: root

    AfterInstall:
        - location: scripts/after-install.sh
        timeout: 10
        runas: root
    ApplicationStart:
        - location: scripts/app-start.sh
        timeout: 10
        runas: root
    ValidateService:
        - location: scripts/validate.sh
        timeout: 30
        runas: root
```

- Dockerfile and requirements.txt for the FastAPI application.

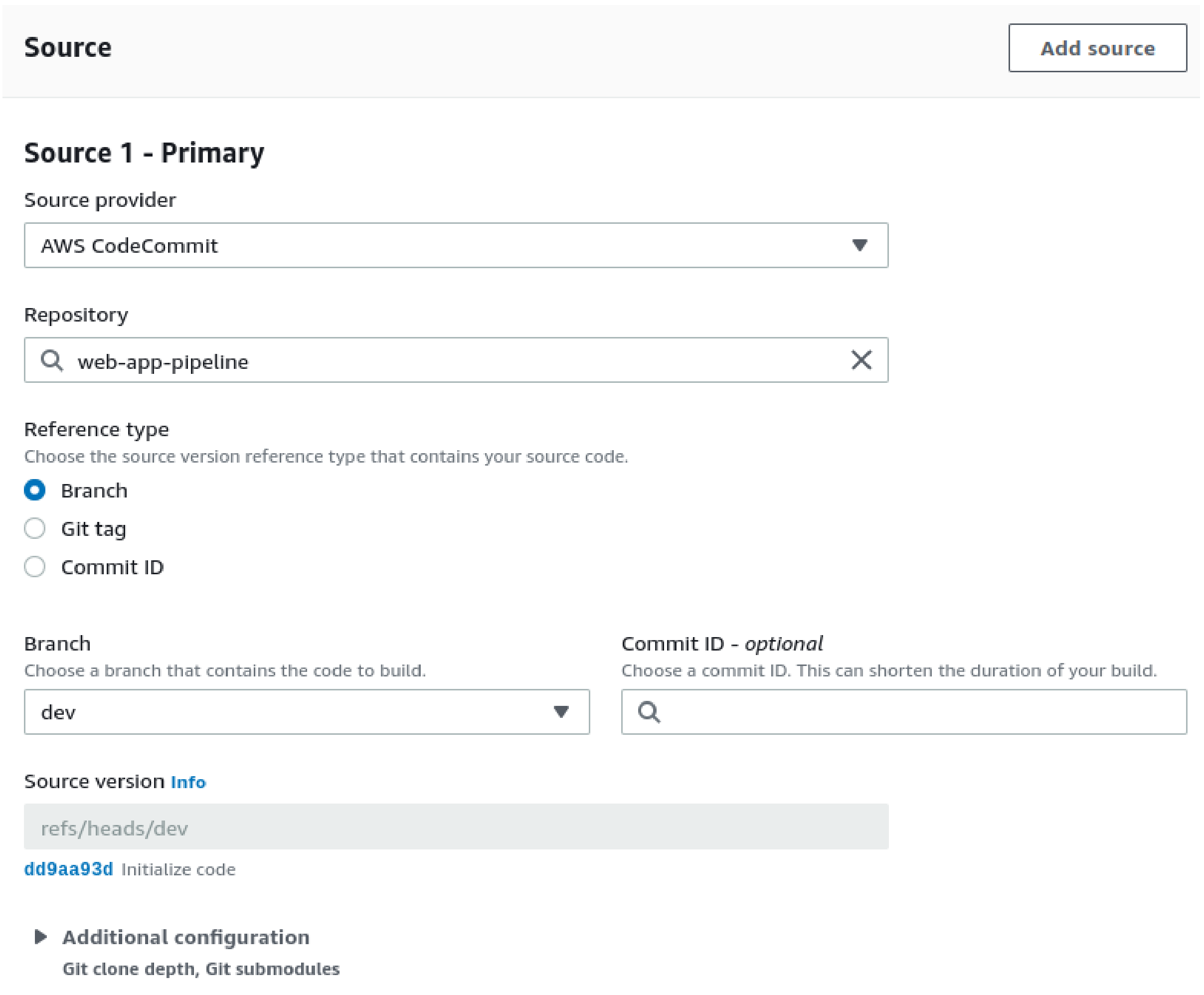
```
FROM python:3.9-slim

WORKDIR /code

RUN apt-get update \
&& apt-get install -y --no-install-recommends \
        curl\
&& apt-get autoremove -yqq --purge \
&& apt-get clean \
&& rm -rf /var/lib/apt/lists/*

COPY ./requirements.txt /code/requirements.txt

RUN pip install --no-cache-dir --upgrade -r
/code/requirements.txt

COPY ./app /code/app

CMD ["uvicorn", "app.main:app", "--host", "0.0.0.0", "--port",
"80"]
```

- Scripts for each hook in the CD pipeline: in `./scripts/` directory. Somme scripts are hard-coded so they may need to be modified during the setup process.

> **IMPORTANT:** Before you begin, make sure to check these scripts and modify some hard-coded variables such as ECR repo names, deployment group names, AWS account id,.. to fit your requirements.

2. Codebuild:

- Click **Create build projects** and configure the codebuild pipline:
  - `Project configuration` section: project name
  - `Source` section:



Source provider: AWS CodeCommit

Repository: web-app-pipeline

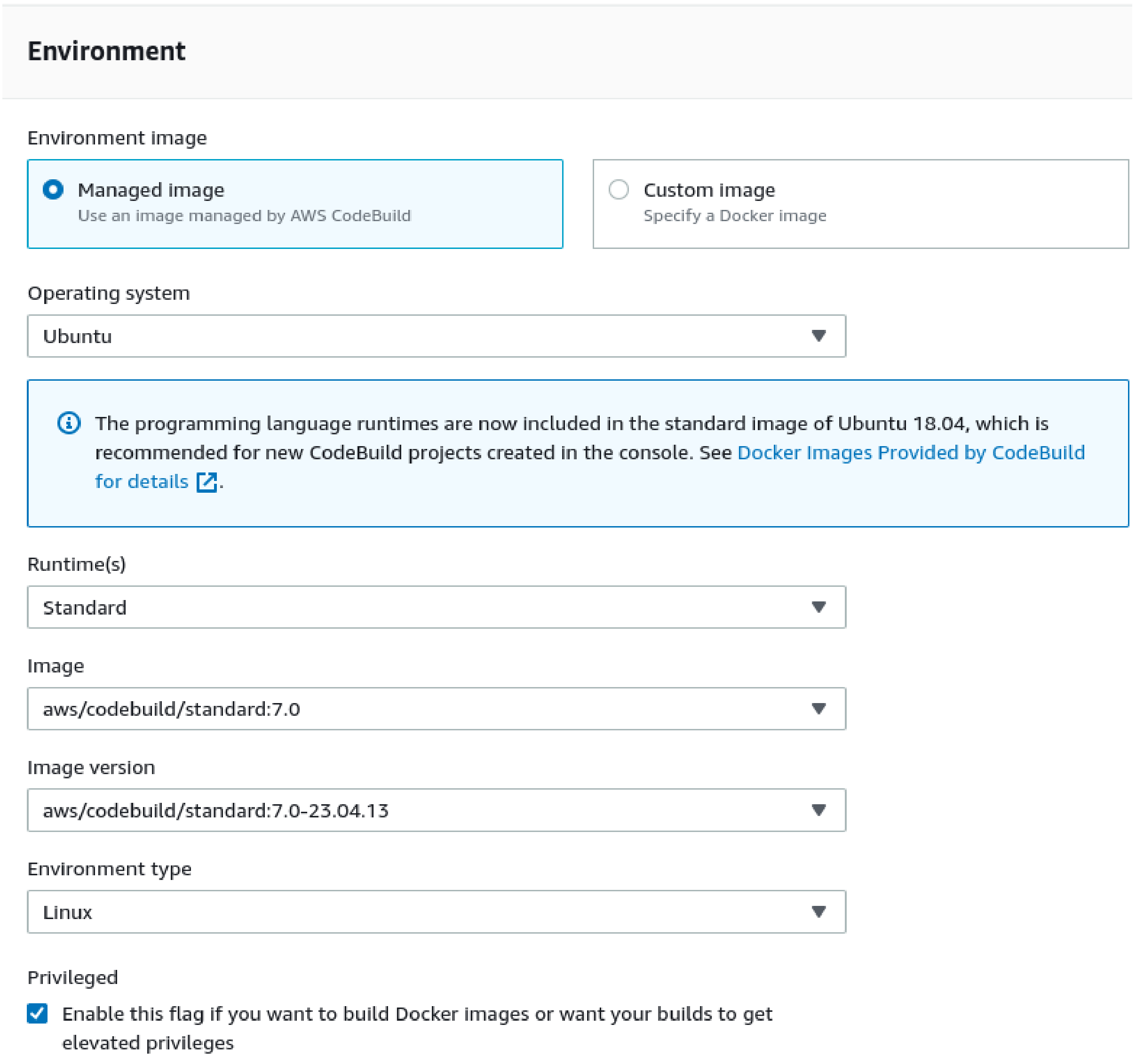
Reference type
Choose the source version reference type that contains your source code.
[✓ Branch] Git tag Commit ID

Branch
Choose a branch that contains the code to build.
dev

Commit ID - *optional*
Choose a commit ID. This can shorten the duration of your build.

Source version Info
refs/heads/dev
dd9aa93d Initialize code

- `Environment` section: Remember to add environment variables in the additional configuration.



Environment

Environment image

(●) Managed image
Use an image managed by AWS CodeBuild

( ) Custom image
Specify a Docker image

Operating system: Ubuntu

The programming language runtimes are now included in the standard image of Ubuntu 18.04, which is recommended for new CodeBuild projects created in the console. See Docker Images Provided by CodeBuild for details.

Runtime(s): Standard

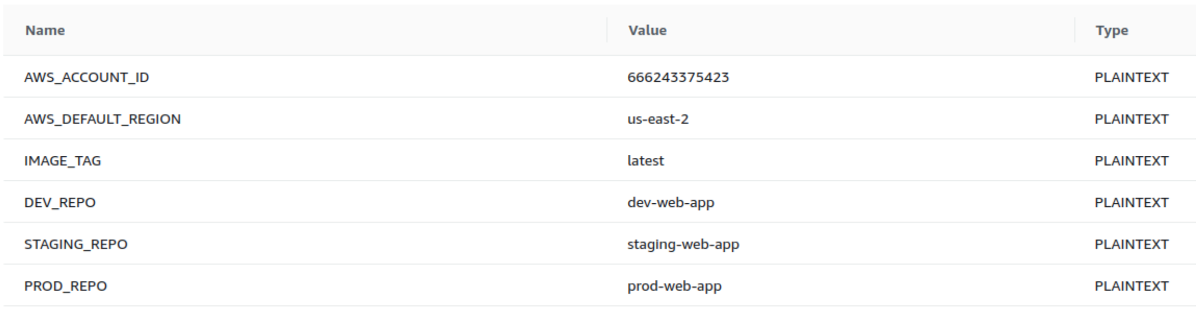
Image: aws/codebuild/standard:7.0

Image version: aws/codebuild/standard:7.0-23.04.13

Environment type: Linux

Privileged

[✓] Enable this flag if you want to build Docker images or want your builds to get elevated privileges



Name	Value	Type
AWS_ACCOUNT_ID	666243375423	PLAINTEXT
AWS_DEFAULT_REGION	us-east-2	PLAINTEXT
IMAGE_TAG	latest	PLAINTEXT
DEV_REPO	dev-web-app	PLAINTEXT
STAGING_REPO	staging-web-app	PLAINTEXT
PROD_REPO	prod-web-app	PLAINTEXT

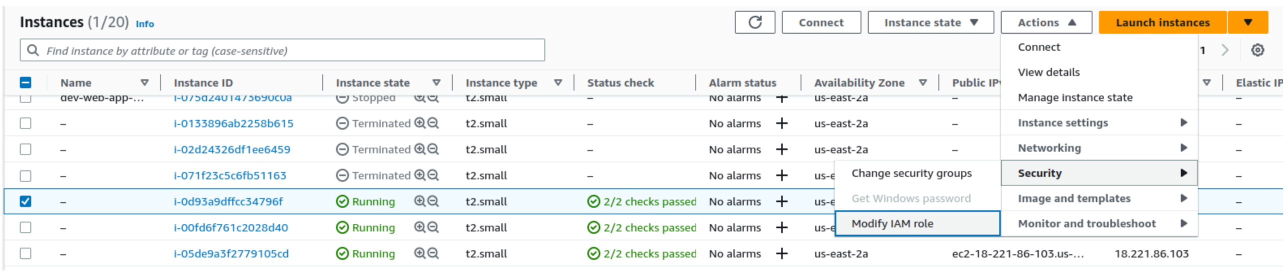
    - `*_REPO` variables are the names of the ECR repositories created for dev, staging, prod environments.
  - Just leave the default configuration for other sections and watch out for the errors for not having proper permissions
- Try to create a build and if it succeeds, you are good to go!

3. Codedeploy

- 3 EC2 instances need to be set up for 3 stages of the deployment process. Here a `Launch template` is used to easily create instances having the same system configuration. Their OS should be Ubuntu 20.04 and other settings can be configured at your own choice. Remember to add the following commands to `User data` in the advanced settings and attach to an IAM role having CodeDeploy related permissions for these instances and change their names after created. **Remember to change the region YOUR_REGION in the codedeploy download link to your current region** in the script below:

```
#!/bin/bash
sudo apt update
yes | sudo apt install ruby-full wget

cd /home/ubuntu
wget https://aws-codedeploy-{YOUR_REGION}.s3.
{YOUR_REGION}.amazonaws.com/latest/install
chmod +x ./install
sudo ./install auto > /tmp/logfile
sudo service codedeploy-agent start

curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

> ***IMPORTANT:*** default profile must be configured in the instances for docker commands to run successfully.



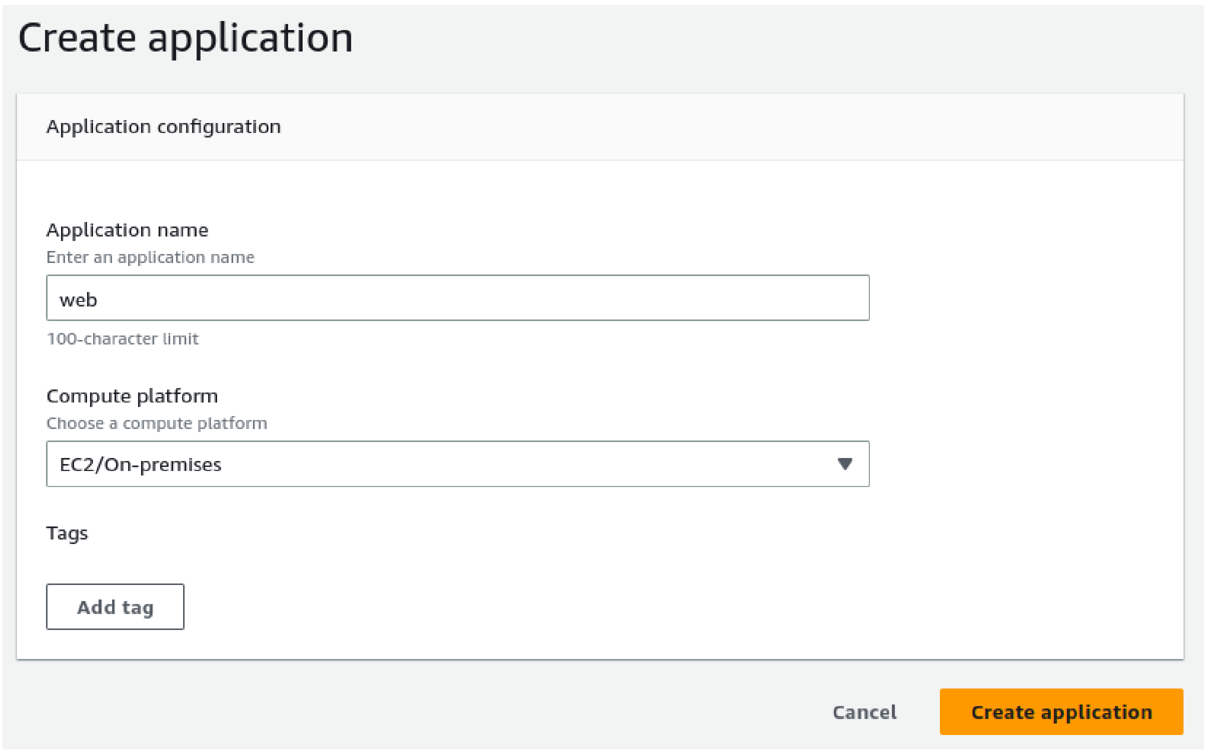
Instances (1/20) Info

Find instance by attribute or tag (case-sensitive)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP		Elastic IP
☐	dev-web-app-...	i-075d2401473690c0a	Stopped	t2.small	–	No alarms	us-east-2a	–		–
☐	–	i-0133896ab2258b615	Terminated	t2.small	–	No alarms	us-east-2a	–		–
☐	–	i-02d24326df1ee6459	Terminated	t2.small	–	No alarms	us-east-2a	–		–
☐	–	i-071f23c5c6fb51163	Terminated	t2.small	–	No alarms	us-e			–
✓	–	i-0d93a9dffcc34796f	Running	t2.small	2/2 checks passed	No alarms	us-e			–
☐	–	i-00fd6f761c2028d40	Running	t2.small	2/2 checks passed	No alarms	us-e			–
☐	–	i-05de9a3f2779105cd	Running	t2.small	2/2 checks passed	No alarms	us-east-2a	ec2-18-221-86-103.us-...	18.221.86.103	–

> **Note:** `sudo service codedeploy-agent restart` for the attached role to take effect

- Create application in CodeDeploy and 3 deployment groups for 3 stages of deployment:
  - Create an application:



Create application

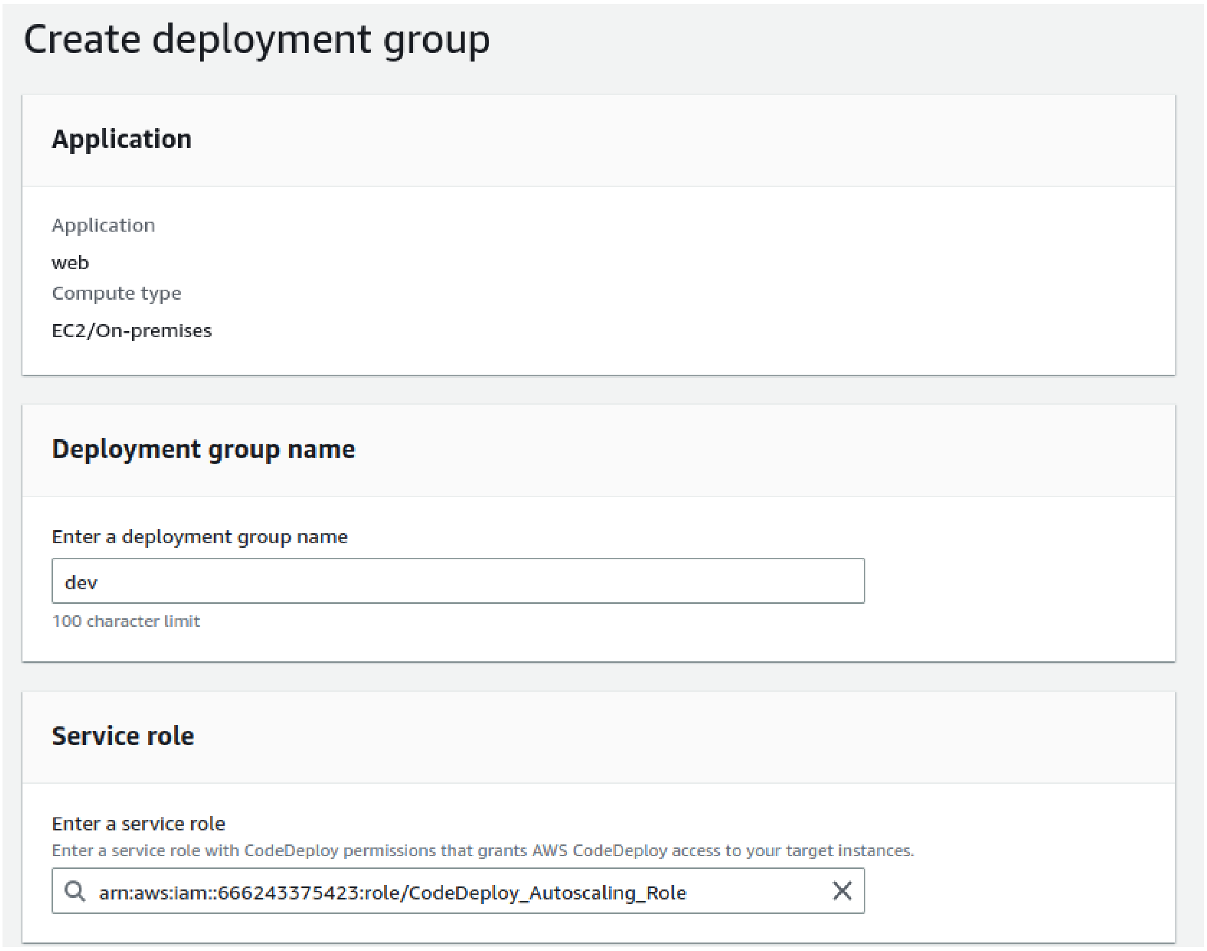
Application configuration

Application name
Enter an application name
web
100-character limit

Compute platform
Choose a compute platform
EC2/On-premises

Tags

- Create a dev deployment group inside the created application. You should name the deployment groups `dev-web-app-deployment`, `staging-web-app-deployment`, `prod-web-app-deployment` so that some of the hard-coded parts in available scripts don't need to be changed:



Create deployment group

Application

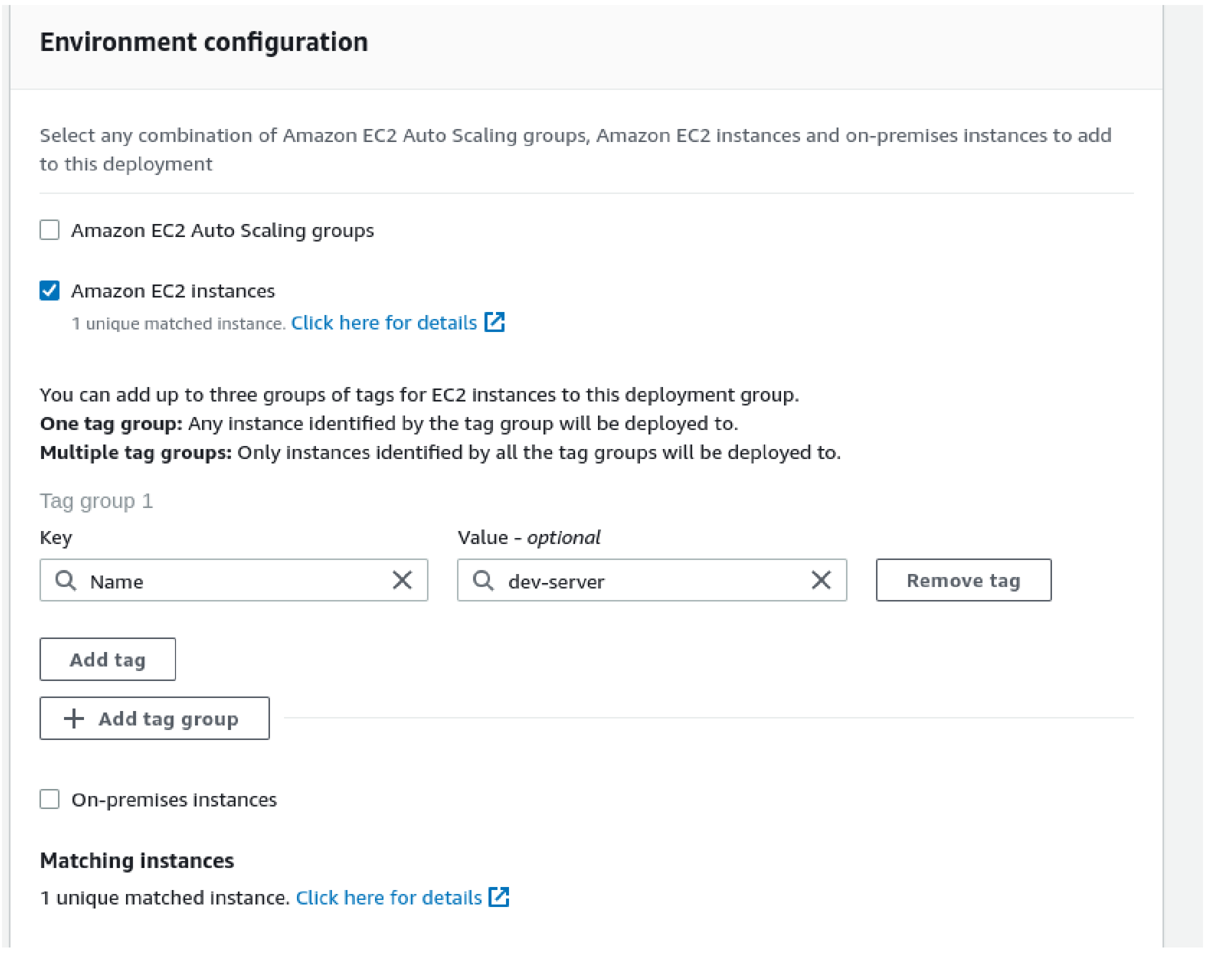
Application
web
Compute type
EC2/On-premises

Deployment group name

Enter a deployment group name
dev
100 character limit

Service role

Enter a service role
Enter a service role with CodeDeploy permissions that grants AWS CodeDeploy access to your target instances.
arn:aws:iam::666243375423:role/CodeDeploy_Autoscaling_Role



Environment configuration

Select any combination of Amazon EC2 Auto Scaling groups, Amazon EC2 instances and on-premises instances to add to this deployment

☐ Amazon EC2 Auto Scaling groups

☑ Amazon EC2 instances

1 unique matched instance. Click here for details

You can add up to three groups of tags for EC2 instances to this deployment group.
One tag group: Any instance identified by the tag group will be deployed to.
Multiple tag groups: Only instances identified by all the tag groups will be deployed to.

Tag group 1

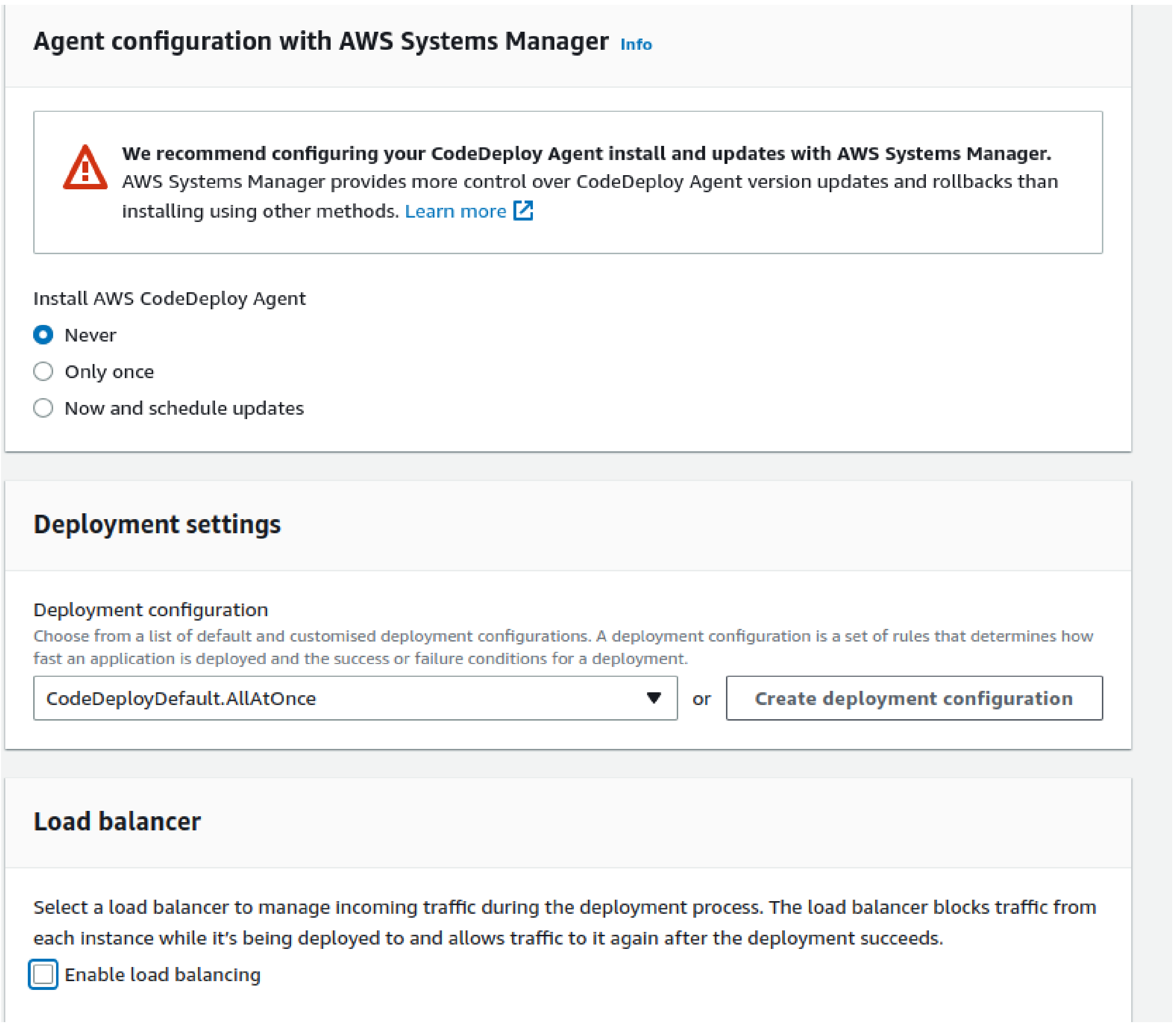
Key	Value - *optional*
Name	dev-server

☐ On-premises instances

Matching instances

1 unique matched instance. Click here for details

> **Note:** the value of the key "Name" in the tag group must be the name of the instance of the environment you want to deploy.



Agent configuration with AWS Systems Manager Info

We recommend configuring your CodeDeploy Agent install and updates with AWS Systems Manager. AWS Systems Manager provides more control over CodeDeploy Agent version updates and rollbacks than installing using other methods. Learn more

Install AWS CodeDeploy Agent

[✓ Never]
Only once
Now and schedule updates

Deployment settings

Deployment configuration
Choose from a list of default and customised deployment configurations. A deployment configuration is a set of rules that determines how fast an application is deployed and the success or failure conditions for a deployment.

CodeDeployDefault.AllAtOnce

Load balancer

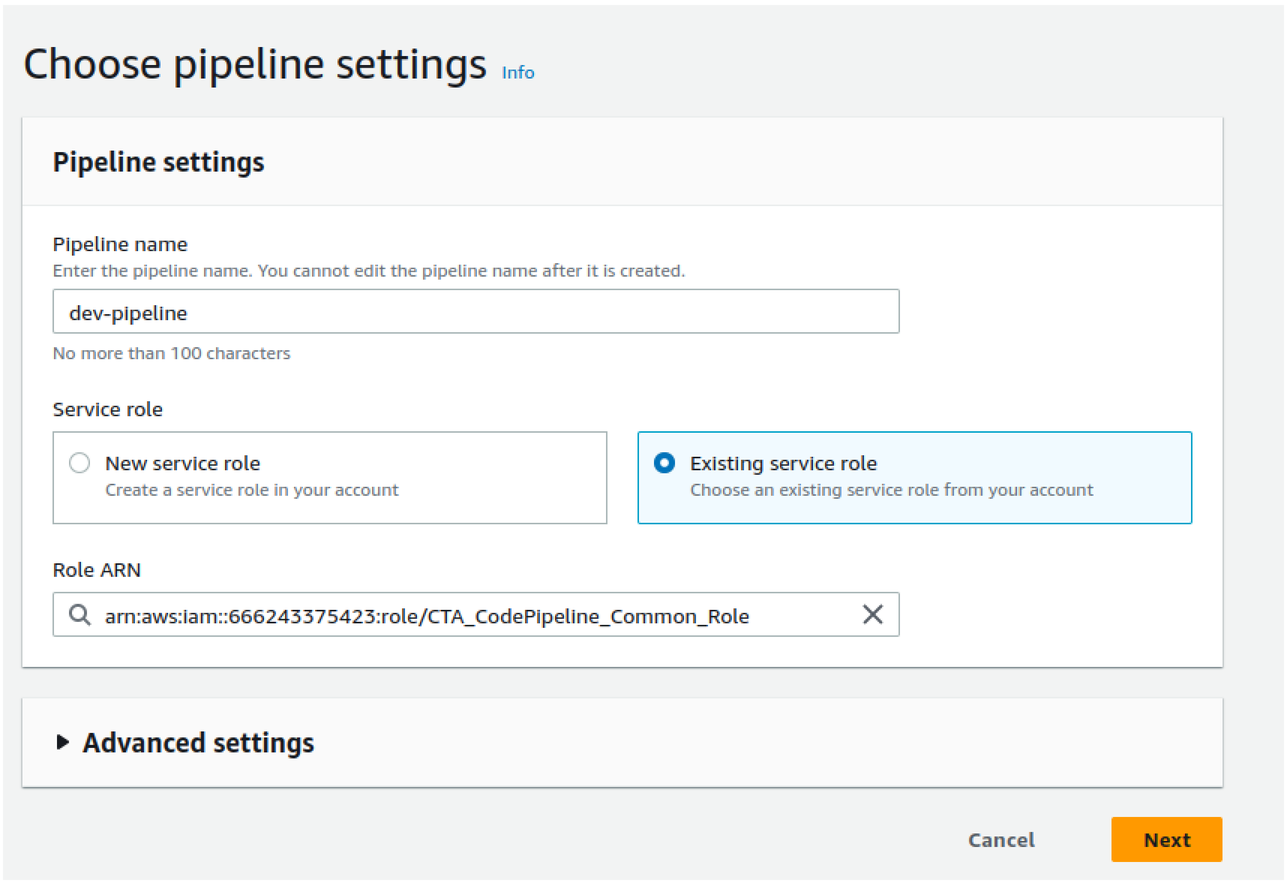
Select a load balancer to manage incoming traffic during the deployment process. The load balancer blocks traffic from each instance while it's being deployed to and allows traffic to it again after the deployment succeeds.

☐ Enable load balancing

- Repeat these steps similarly for the other stages (staging, prod)

4. CodePipeline

- Create 2 pipelines, one for each branch of the git repository: `master` (default) and `dev`
- Pipeline for dev branch:



Choose pipeline settings

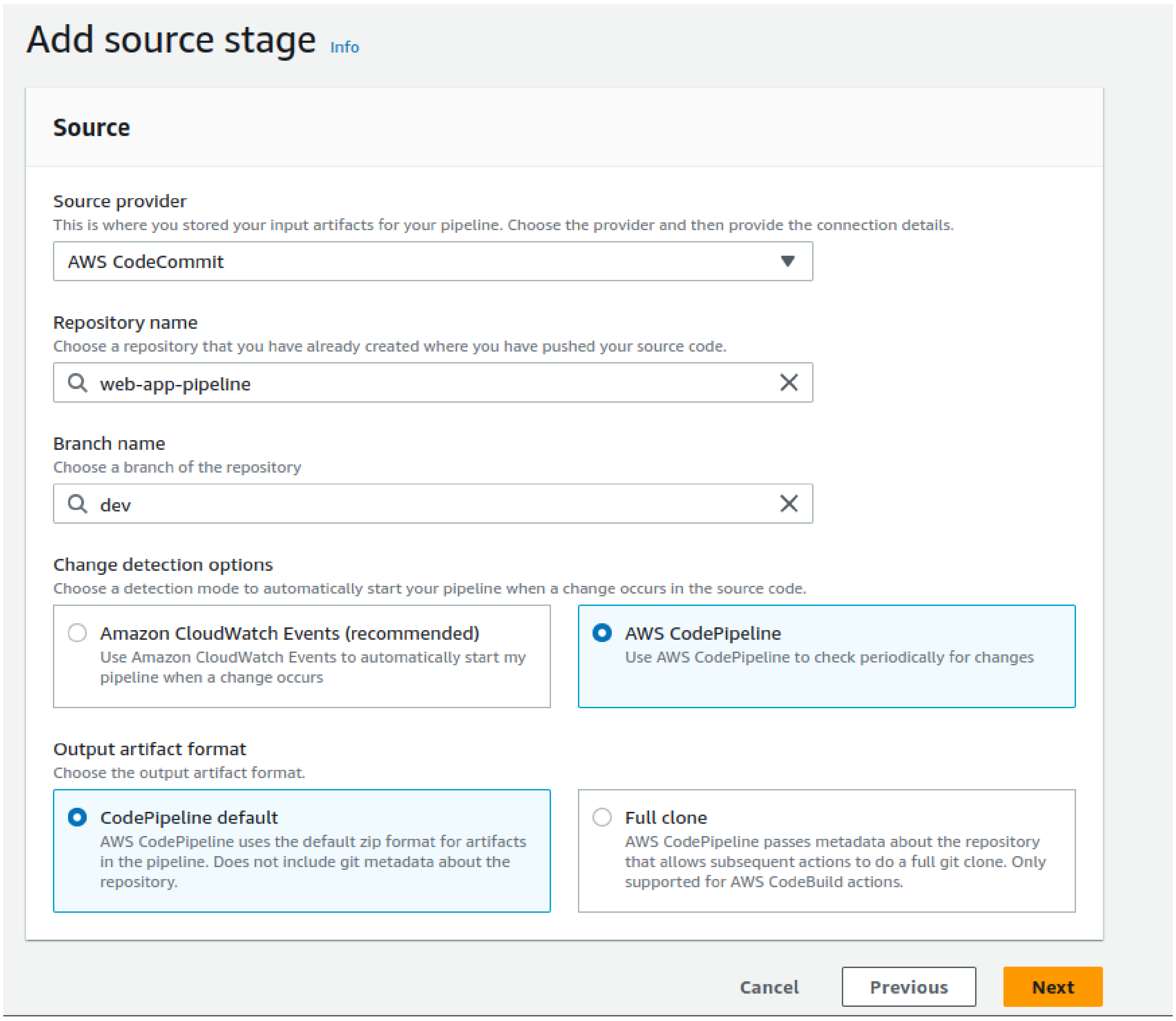
Pipeline settings

Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
dev-pipeline
No more than 100 characters

Service role
New service role [✓ Existing service role]
Create a service role in your account / Choose an existing service role from your account

Role ARN
arn:aws:iam::666243375423:role/CTA_CodePipeline_Common_Role

Advanced settings



Add source stage

Source

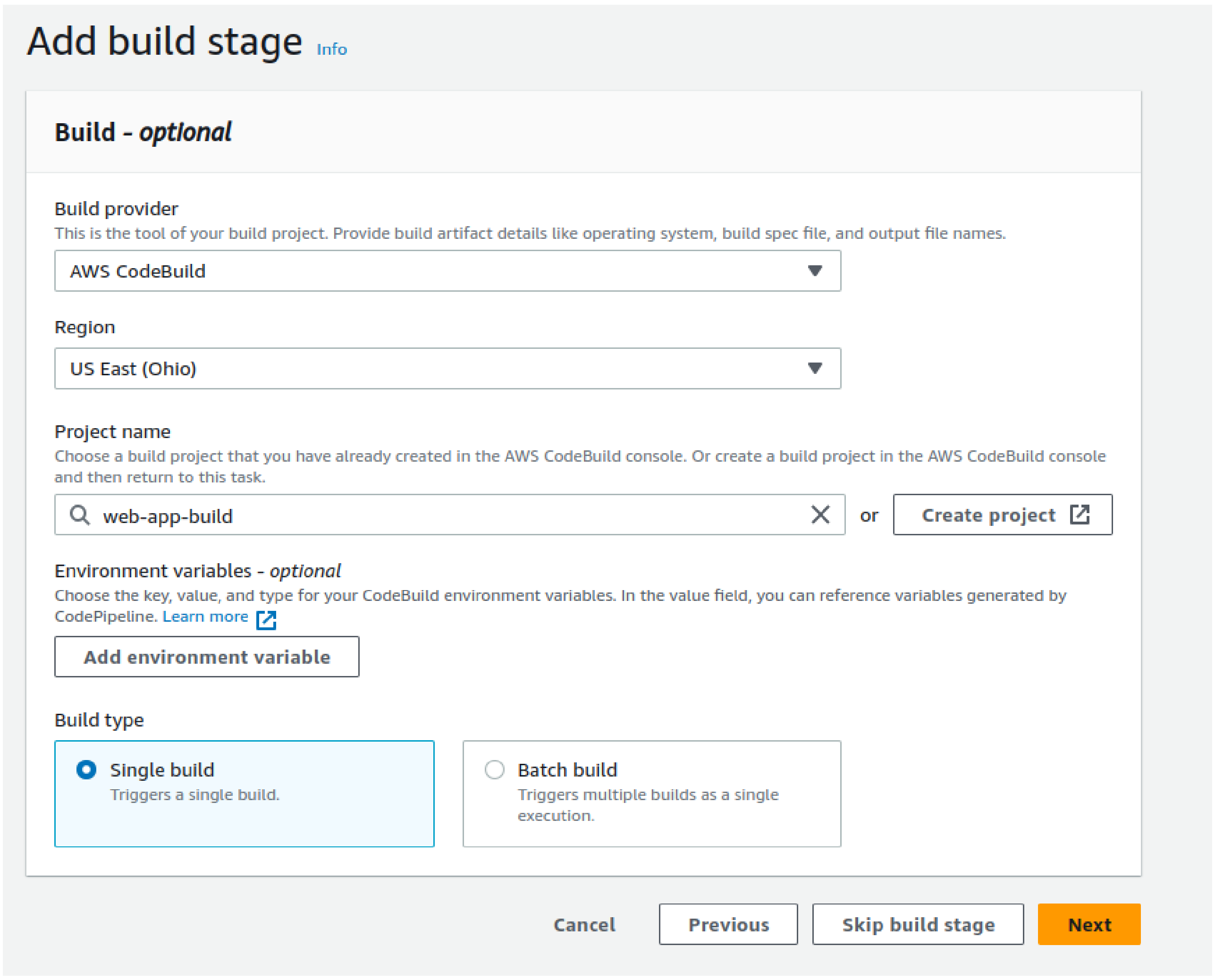
Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
AWS CodeCommit

Repository name
Choose a repository that you have already created where you have pushed your source code.
web-app-pipeline

Branch name
Choose a branch of the repository
dev

Change detection options
Choose a detection mode to automatically start your pipeline when a change occurs in the source code.
Amazon CloudWatch Events (recommended) — Use Amazon CloudWatch Events to automatically start my pipeline when a change occurs
[✓ AWS CodePipeline] — Use AWS CodePipeline to check periodically for changes

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.



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 (Ohio)

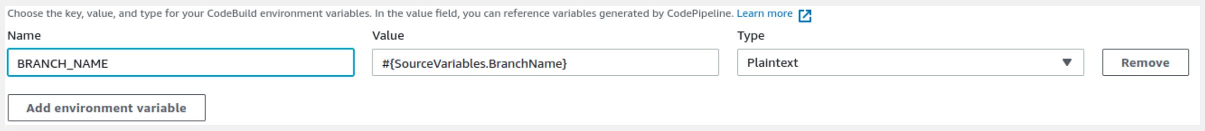
Project name
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.
web-app-build or Create project

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
Add environment variable

Build type
[✓ Single build] Triggers a single build.
Batch build Triggers multiple builds as a single execution.

Cancel Previous Skip build stage Next

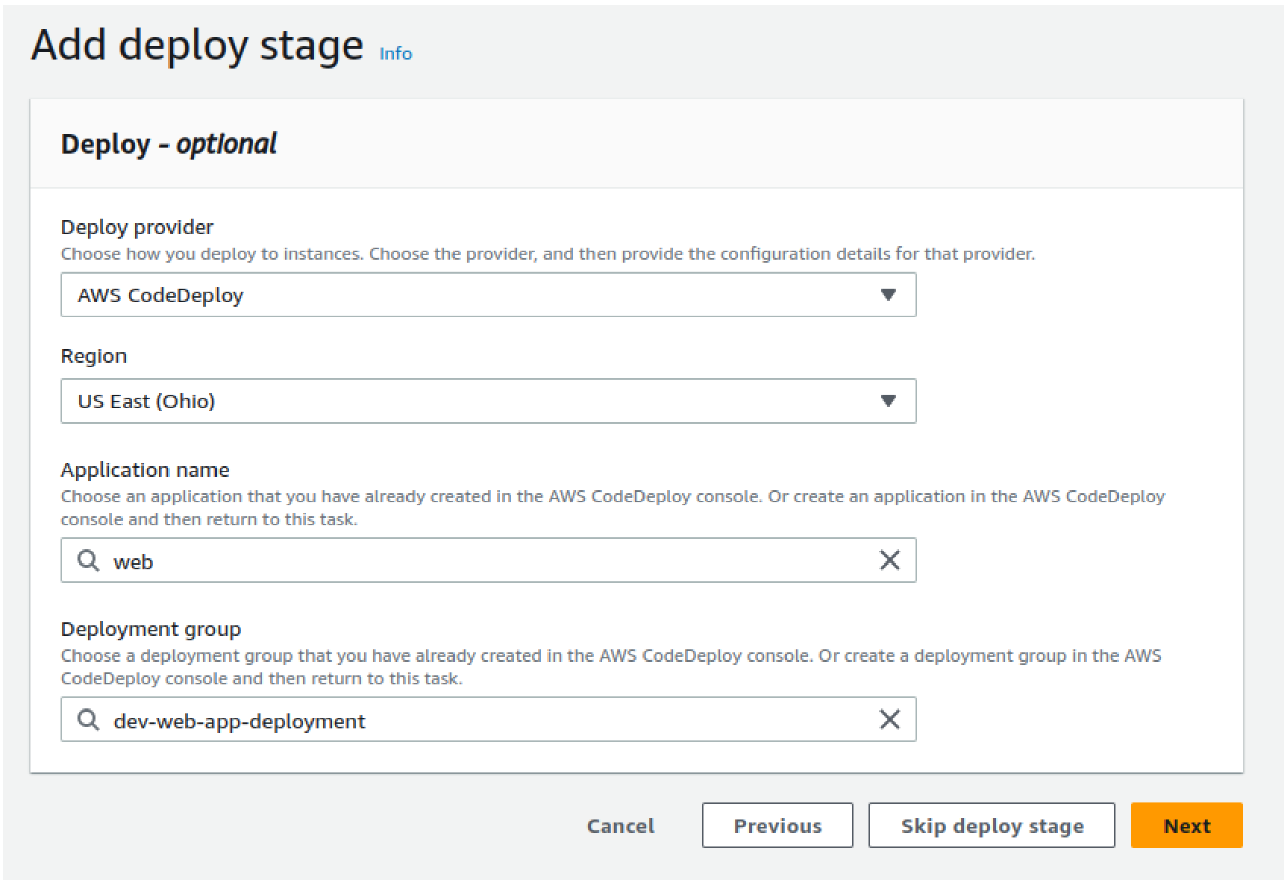
> **IMPORTANT:** Add BRANCH_NAME to the environment variables as shown below:



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
BRANCH_NAME	#{SourceVariables.BranchName}	Plaintext

Remove
Add environment variable


Add deploy stage Info

Deploy - *optional*

Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.
AWS CodeDeploy

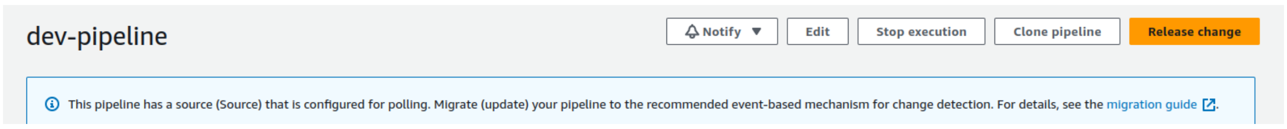
Region
US East (Ohio)

Application name
Choose an application that you have already created in the AWS CodeDeploy console. Or create an application in the AWS CodeDeploy console and then return to this task.
web

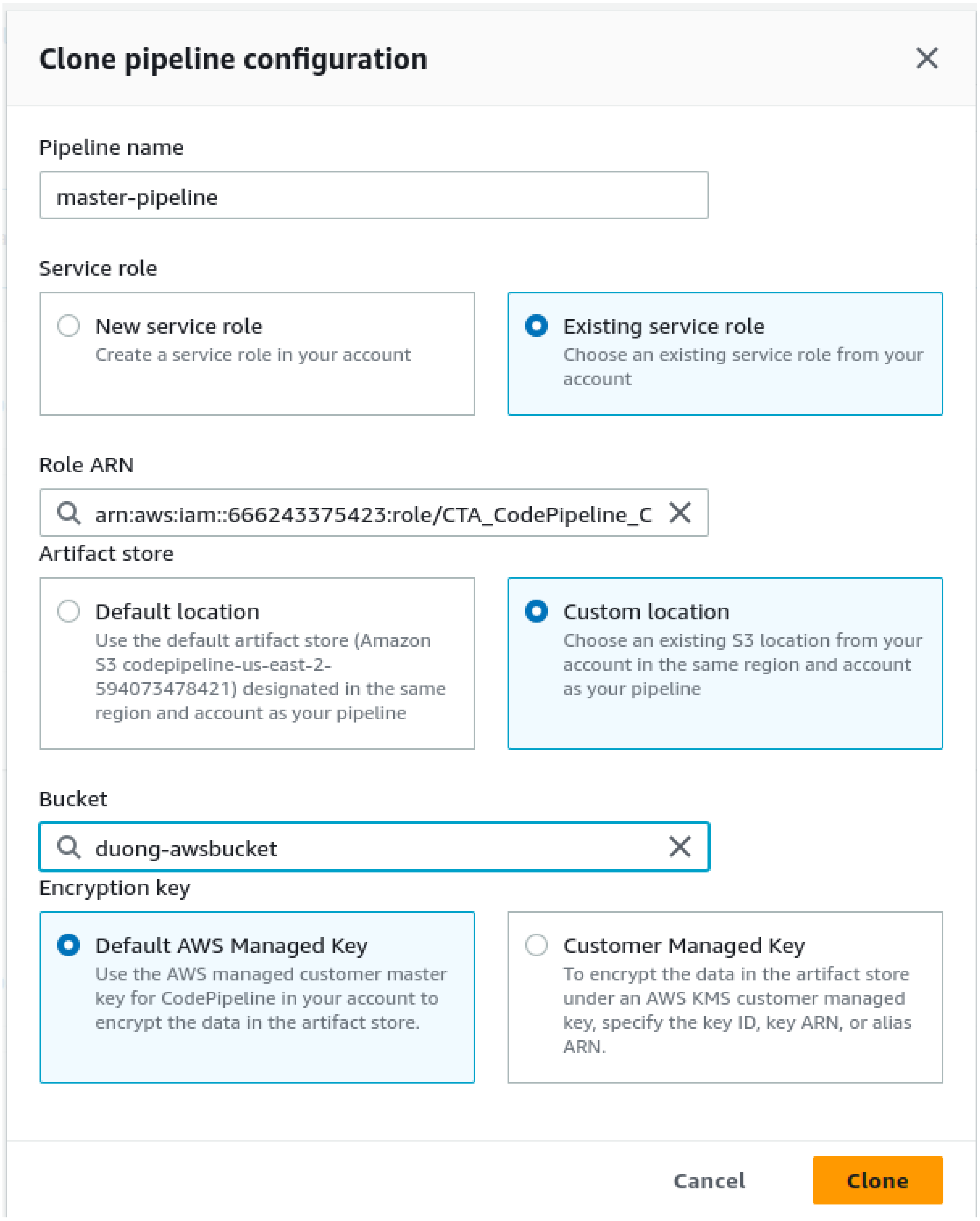
Deployment group
Choose a deployment group that you have already created in the AWS CodeDeploy console. Or create a deployment group in the AWS CodeDeploy console and then return to this task.
dev-web-app-deployment

    - When the pipeline is created, it will automatically run and should return successful status.

- Pipeline for master branch: Clone the dev pipeline and make some modifications:


dev-pipeline

This pipeline has a source (Source) that is configured for polling. Migrate (update) your pipeline to the recommended event-based mechanism for change detection. For details, see the migration guide.



Clone pipeline configuration

Pipeline name

master-pipeline

Service role

( ) New service role
Create a service role in your account

(●) Existing service role
Choose an existing service role from your account

Role ARN

arn:aws:iam::666243375423:role/CTA_CodePipeline_C

Artifact store

( ) Default location
Use the default artifact store (Amazon S3 codepipeline-us-east-2-594073478421) 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

Bucket

duong-awsbucket

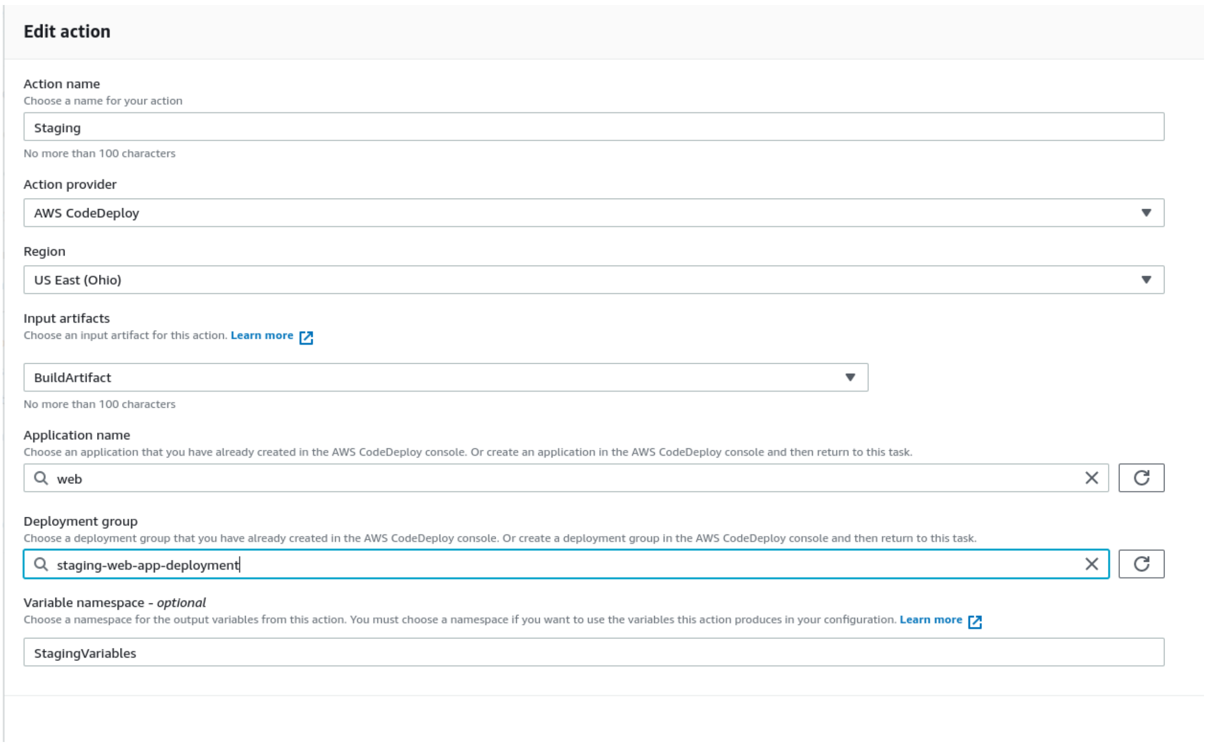
Encryption key

(●) 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.

- Edit Source stage in the cloned pipeline: change branch `dev` to `master` and save.
- Edit Deploy stage to have `Staging` and `Production` deployment.

  Modify the existing block in the `Deploy` stage as shown below:



Edit action

Action name
Choose a name for your action
Staging
No more than 100 characters

Action provider
AWS CodeDeploy

Region
US East (Ohio)

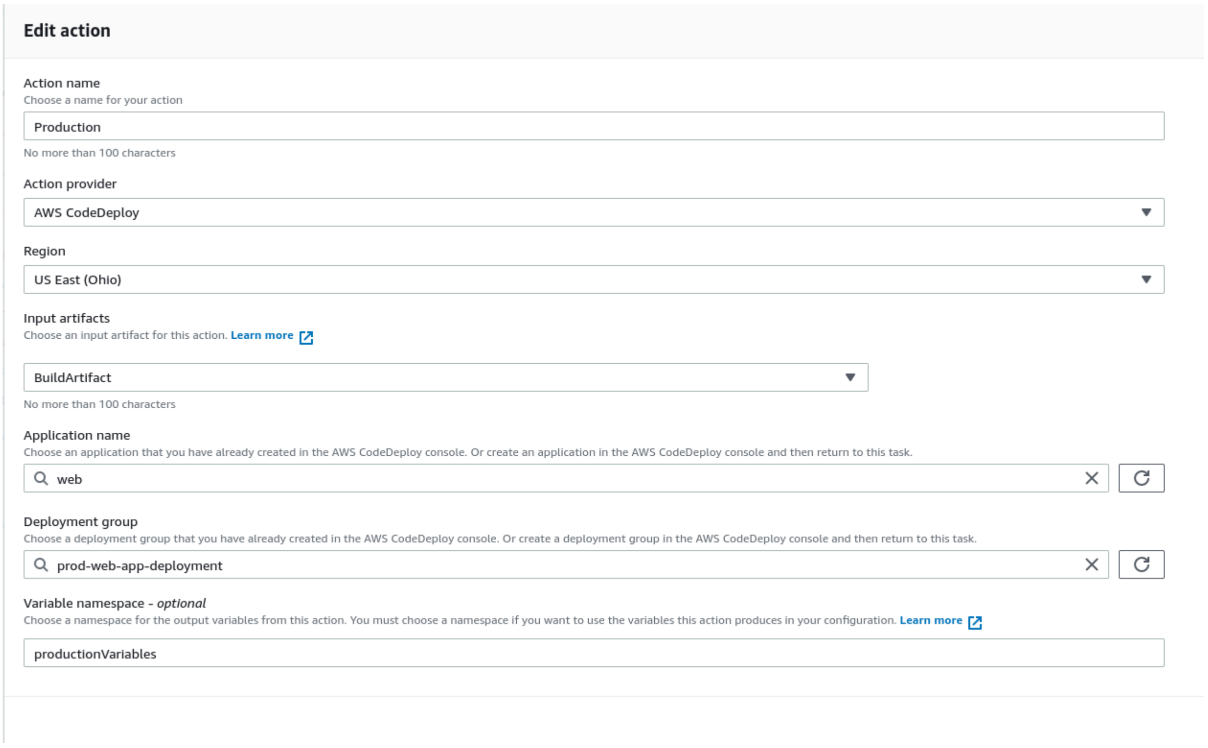
Input artifacts
Choose an input artifact for this action. Learn more
BuildArtifact
No more than 100 characters

Application name
Choose an application that you have already created in the AWS CodeDeploy console. Or create an application in the AWS CodeDeploy console and then return to this task.
web

Deployment group
Choose a deployment group that you have already created in the AWS CodeDeploy console. Or create a deployment group in the AWS CodeDeploy console and then return to this task.
staging-web-app-deployment

Variable namespace - *optional*
Choose a namespace for the output variables from this action. You must choose a namespace if you want to use the variables this action produces in your configuration. Learn more
StagingVariables

Add action group below the `Staging` block with configuration below:



Edit action

Action name
Choose a name for your action
Production
No more than 100 characters

Action provider
AWS CodeDeploy

Region
US East (Ohio)

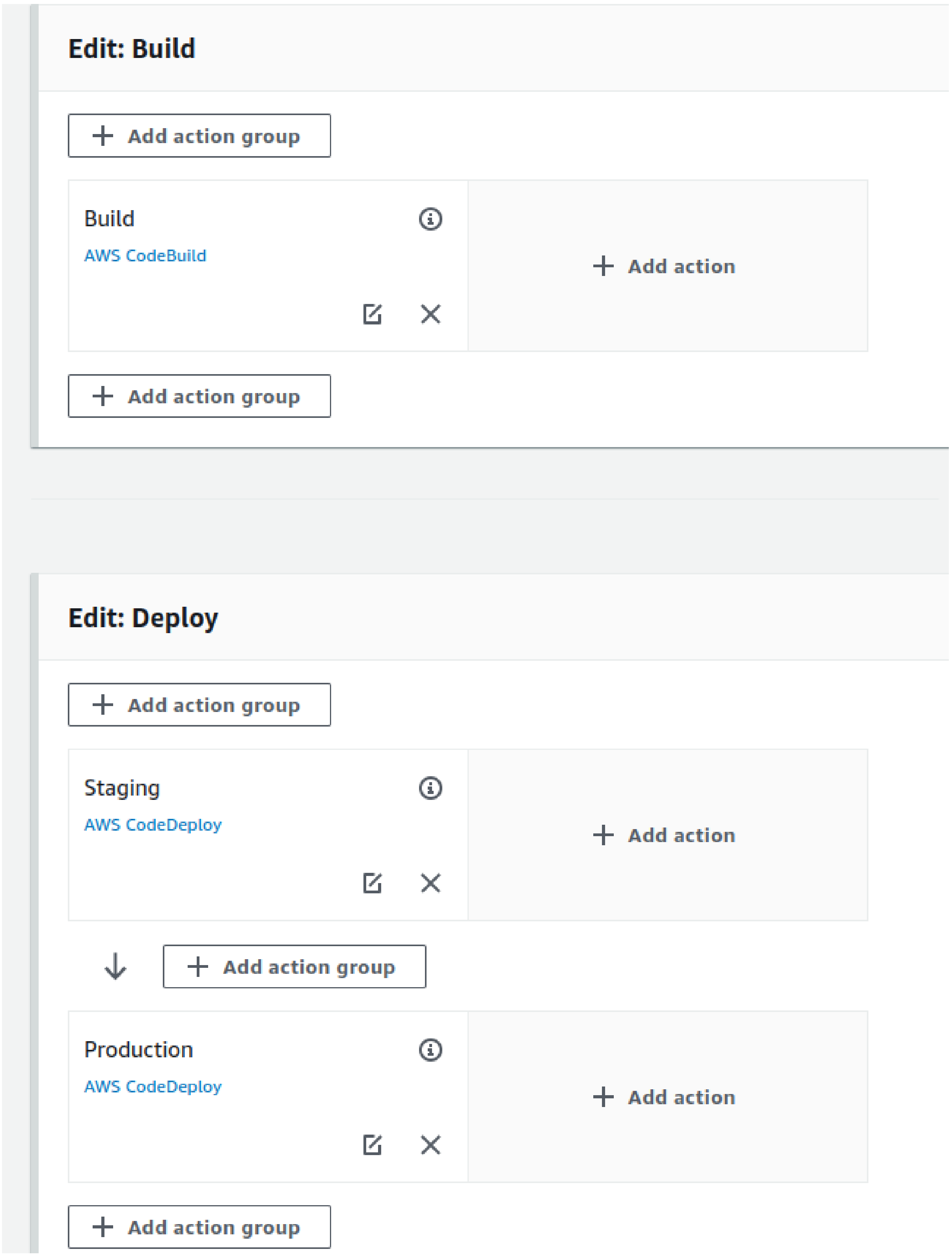
Input artifacts
Choose an input artifact for this action. Learn more
BuildArtifact
No more than 100 characters

Application name
Choose an application that you have already created in the AWS CodeDeploy console. Or create an application in the AWS CodeDeploy console and then return to this task.
web

Deployment group
Choose a deployment group that you have already created in the AWS CodeDeploy console. Or create a deployment group in the AWS CodeDeploy console and then return to this task.
prod-web-app-deployment

Variable namespace - *optional*
Choose a namespace for the output variables from this action. You must choose a namespace if you want to use the variables this action produces in your configuration. Learn more
productionVariables

- The resulted pipeline will be like this:



Referenes:

- Build specification reference for CodeBuild
- AWS CodePipeline Documentation
- AWS CodeDeploy Documentation
- FastAPI