

CodePipeline

!!!선행!!!

- CodeBuild가 먼저 설정되어야한다.

code commit에 코드를 push하고 자동 배포가 되려면 code pipeline이 설정되어야한다.

<https://ap-northeast-2.console.aws.amazon.com/codesuite/codepipeline/pipelines?region=ap-northeast-2&pipelines-meta=eyJmIjp7InRleHQiOiIifSwicyI6eyJwcm9wZXJ0eSI6InVwZGF0ZWQlLCJkaXJlY3Rpb24iOi0xfSwibiI6MzAsImkiOiB9>

생성 옵션 선택

- deployment는 github gitlab등 다른 사이트의 코드를 가져온다 aws codeCommit은 사용자 지정파이프라인 빌드로 간다.

Step 1
생성 옵션 선택

Step 2
파이프라인 설정 선택

Step 3
소스 스테이지 추가

Step 4
빌드 스테이지 추가

Step 5
테스트 스테이지 추가

Step 6
배포 스테이지 추가

Step 7
검토

생성 옵션 선택
Step 1 of 7

Category

☐ Deployment

☐ Continuous Integration

☐ Automation

☒ 사용자 지정 파이프라인 빌드

취소 다음

파이프라인 설정

파이프라인 설정 선택
정보
Step 2 of 7

파이프라인 설정

파이프라인 이름
파이프라인 이름을 입력합니다. 생성된 후에는 파이프라인 이름을 편집할 수 없습니다.

pet-web

100자를 초과할 수 없음

실행 모드
정보
파이프라인의 실행 모드를 선택합니다. 이에 따라 파이프라인 실행 방식이 결정됩니다.

☐ 대체

☒ 대기됨

☐ 병렬

서비스 역할

☒ 새 서비스 역할
계정에 서비스 역할 생성

☐ 기존 서비스 역할
계정에서 기존 서비스 역할 선택

역할 이름

AWSCodePipelineServiceRole-ap-northeast-2-pet-web

서비스 역할 이름 입력

☒ AWS CodePipeline이 이 새 파이프라인에 사용할 서비스 역할을 생성하도록 허용

고급 설정
파이프라인의 아티팩트 저장소 위치, 암호화 설정, 파이프라인 변수를 구성합니다.

취소 이전 다음

소스 스테이지 추가

소스 스테이지 추가 정보

Step 3 of 7

소스

소스 공급자
파이프라인의 입력 아티팩트를 저장한 위치입니다. 공급자를 선택한 후 연결 세부 정보를 제공하십시오.

AWS CodeCommit

리포지토리 이름
이미 생성하여 소스 코드를 푸시한 리포지토리를 선택합니다.

pet-web

브랜치 이름
리포지토리의 브랜치 선택

main

☒ EventBridge 규칙을 생성하여 자동으로 소스 변화를 감지합니다.
비활성화된 경우 AWS 설정서에 따라 소스에 대한 EventBridge 규칙을 생성하세요. [자세히 알아보기](#)

출력 아티팩트 형식
출력 아티팩트 형식을 선택합니다.

☒ CodePipeline 기본값
AWS CodePipeline은 파이프라인의 아티팩트에 대해 기본 zip 형식을 사용합니다. 리포지토리에 대한 Git 메타데이터는 포함하지 않습니다.

☐ 전체 복제
AWS CodePipeline은 후속 작업에서 전체 Git 복제를 수행할 수 있도록 리포지토리에 대한 메타데이터를 전달합니다. AWS CodeBuild 작업에 대해서만 지원됩니다. [자세히 알아보기](#)

☒ 스테이지 장애 시 자동 재시도 활성화

취소

이전

다음

빌드 스테이지 추가

- CodeBuild를 먼저 설정하였으면 프로젝트 이름에 프로젝트가 뜬다. 그리고 프로젝트에 코드에 있는 buildspec.yml을 사용한다.

빌드 스테이지 추가 정보

Step 4 of 7

빌드 - 선택 사항

빌드 공급자
빌드 명령을 실행하고 빌드 작업의 아티팩트를 저장하는 데 사용할 도구를 선택합니다.

☐ 명령

☒ 기타 빌드 공급자

AWS CodeBuild

프로젝트 이름
AWS CodeBuild 콘솔에서 이미 생성한 빌드 프로젝트를 선택합니다. 또는 AWS CodeBuild 콘솔에서 빌드 프로젝트를 생성한 후 이 작업으로 들어옵니다.

pet-app-router-test

또는

프로젝트 생성

☒ buildspec 오버라이드 정의 - 선택 사항
빌드 프로젝트에 정의된 최신 항목을 오버라이드하는 buildspec 파일 또는 정의(이 빌드에만 해당)

buildspec 오버라이드

☒ buildspec 파일 사용
YAML 형식의 buildspec 파일에 빌드 명령 저장

☐ 빌드 명령 삽입
빌드 명령을 빌드 프로젝트 구성으로 저장

buildspec 이름
소스 루트의 buildspec 파일 경로를 입력합니다.(예: configuration/buildspec.yml).

buildspec.yml

환경 변수 - 선택 사항
CodeBuild 환경 변수의 키, 값 및 유형을 선택합니다. 값 빌드에서 CodePipeline에서 생성된 변수를 참조할 수 있습니다. [자세히 알아보기](#)

환경 변수 추가

빌드 유형

☒ 단일 빌드
단일 빌드를 트리거합니다.

☐ 배치 빌드
여러 빌드를 단일 실행으로 트리거합니다.

환경 변수 - 선택 사항
CodeBuild 환경 변수의 키, 값 및 유형을 선택합니다. 값 필드에서 CodePipeline에서 생성된 변수를 참조할 수 있습니다. 자세히 알아보십시오 >

환경 변수 추가

빌드 유형

☒ 단일 빌드
단일 빌드를 트리거합니다.

☐ 배치 빌드
여러 빌드를 단일 실행으로 트리거합니다.

리전
아시아 태평양 (서울) ▼

입력 아티팩트
이 작업의 입력 아티팩트를 선택합니다. 자세히 알아보십시오 >

SourceArtifact X
Source에 의해 정의됨

☒ 스테이지 장애 시 자동 재시도 활성화

취소 이전 빌드 스테이지 건너뛰기 다음

테스트 스테이지 건너뛰기 클릭

테스트 스테이지 추가 정보
Step 5 of 7

테스트 - 선택 사항

테스트 제공자
애플리케이션 또는 콘텐츠를 테스트할 방법을 선택합니다. 공급자를 선택한 다음 해당 공급자에 대한 구성 세부 정보를 입력하세요.

▼

☒ 스테이지 장애 시 자동 재시도 활성화

취소 이전 테스트 스테이지 건너뛰기 다음

배포 스테이지 건너뛰기 클릭

배포 스테이지 추가 정보
Step 6 of 7

배포 - 선택 사항

배포 공급자
애플리케이션 또는 콘텐츠를 배포할 방법을 선택합니다. 공급자를 선택한 다음 해당 공급자에 대한 구성 세부 정보를 입력하세요.

▼

☒ 스테이지 장애 시 자동 롤백 구성

☐ 스테이지 장애 시 자동 재시도 활성화

취소 이전 배포 스테이지 건너뛰기 다음

파이프라인 생성 클릭

ecr로 ecs 배포 파이프라인

본 기 관영역서인 합니다.

1. CodePipeline + ECR 소스 액션 사용

파이프라인 구성:

- 소스 스테이지: ECR 리포지토리를 소스로 설정
- 배포 스테이지: ECS 서비스에 배포

파이프라인 생성 단계:

1단계: CodePipeline 생성

- AWS 콘솔에서 CodePipeline 서비스로 이동
- "파이프라인 생성" 선택

2단계: 소스 스테이지 설정

- 소스 공급자: **Amazon ECR** 선택
- 리포지토리 이름: 개발팀이 푸시하는 ECR 리포지토리 이름
- 이미지 태그: `latest` 또는 특정 태그 패턴

3단계: 빌드 스테이지 (선택사항)

- 이미지가 이미 준비되어 있으므로 빌드 스테이지는 생략 가능

4단계: 배포 스테이지 설정

- 배포 공급자: **Amazon ECS** 또는 **Amazon ECS (Blue/Green)**
- 클러스터 이름: 대상 ECS 클러스터
- 서비스 이름: 대상 ECS 서비스
- 이미지 정의 파일: `imageDetail.json` (ECR 소스에서 자동 생성)

2. EventBridge 규칙 자동 생성

CodePipeline을 콘솔에서 생성하면 **EventBridge** 규칙이 자동으로 생성됩니다:

```
{
  "detail-type": ["ECR Image Action"],
  "source": ["aws.ecr"],
  "detail": {
    "action-type": ["PUSH"],
    "image-tag": ["latest"],
    "repository-name": ["your-repo-name"],
    "result": ["SUCCESS"]
  }
}
```

3. 동작 방식

1. 개발팀이 ECR에 이미지 푸시

```
docker push your-account.dkr.ecr.region.amazonaws.com/your-repo:latest
```

2. EventBridge가 ECR 푸시 이벤트 감지

3. CodePipeline 자동 트리거

4. ECS 서비스 자동 업데이트

- 새로운 태스크 정의 리버전 생성
- 롤링 업데이트 또는 Blue/Green 배포 실행

4. 필요한 IAM 권한

CodePipeline 서비스 역할:

- ECR 이미지 읽기 권한
- ECS 서비스 업데이트 권한
- EventBridge 규칙 실행 권한

EventBridge 역할:

- CodePipeline 실행 권한

5. 추가 고려사항

이미지 태그 전략:

- `latest` 태그 사용 시: 매번 같은 태그로 덮어쓰기
- 버전 태그 사용 시: `v1.0.1`, `build-123` 등 고유 태그

배포 전략:

- 롤링 업데이트: 기본 방식, 빠른 배포
- Blue/Green: 무중단 배포, 빠른 롤백 가능

이렇게 설정하면 개발팀이 ECR에 이미지를 푸시하는 즉시 자동으로 ECS 서비스가 업데이트됩니다. 추가로 궁금한 점이나 구체적인 설정 방법이 필요하시면 말씀해 주세요!

▼ Sources

```
1 version: 0.2
2 phases:
3   pre_build:
4     commands:
5       - echo "Starting imagedefinitions.json generation..."
6       - aws --version
7   build:
8     commands:
9       - echo "Creating imagedefinitions.json file..."
10      - echo '["name":"pet-web-container","imageUri":"132334511678.dkr
11  post_build:
12    commands:
13      - echo "Verifying imagedefinitions.json:"
14      - cat imagedefinitions.json
15      - echo "File size:"
16      - wc -c imagedefinitions.json
17 artifacts:
18   files:
19     - imagedefinitions.json
```