



# CI/CD on AWS

Yang Hochal

Solutions Architect

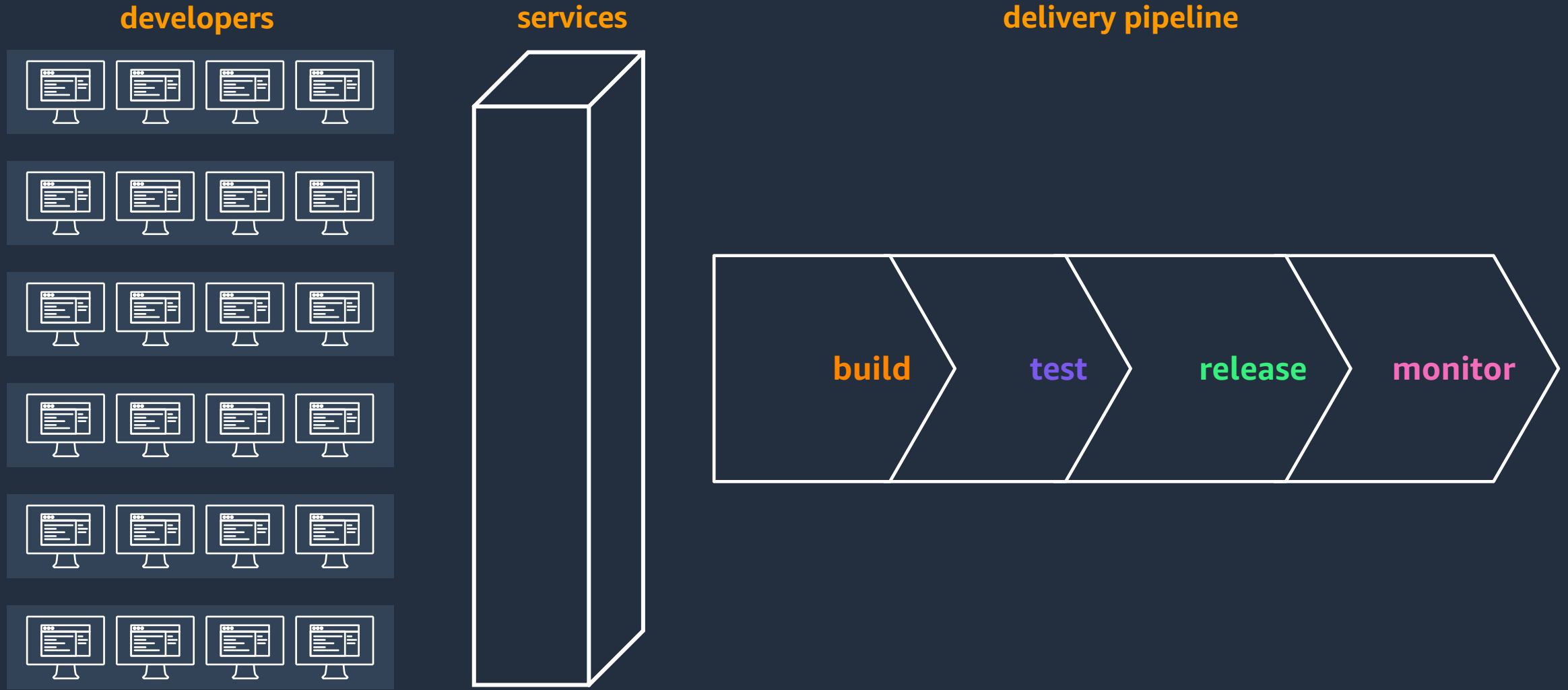
# Agenda

DevOps?

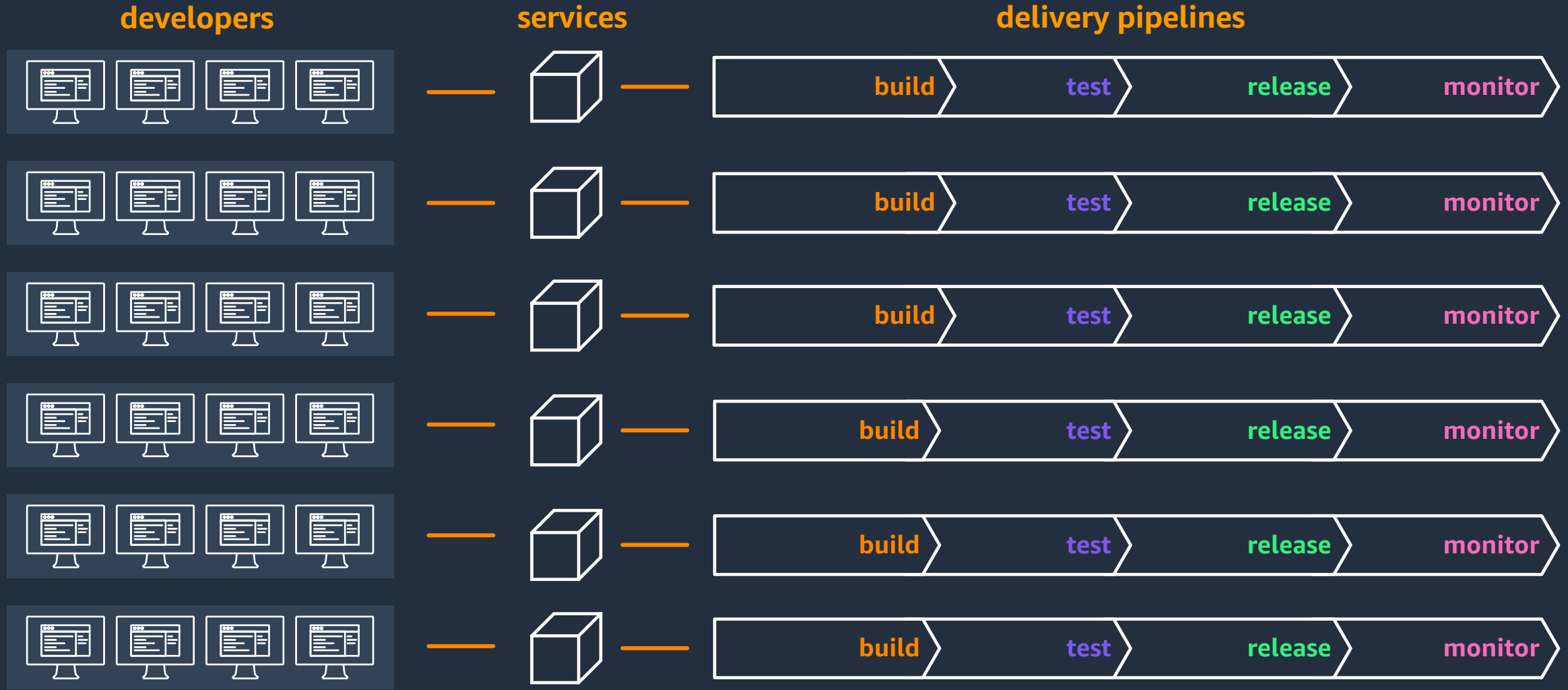
CI/CD Pipeline

# DevOps - Application의 고도화

# Deployment: Monolith 배포 생명주기



# Deployment: Microservice 배포 생명주기



# DevOps?

Culture + Practices + Tools

# DevOps?

## Culture

Ownership

Organization

Individuals

## Practices

## Tools



# DevOps?

## Culture

Ownership

Organization

Individuals

## Practices

Architecture  
patterns

Small, frequent  
updates

Coding practice

Microservice  
patterns

Code packaging

Caching and object  
access patterns

Service discovery

Circuit breaker

Secrets  
management

Code promotions

## Tools

SCM

Continuous  
integration

Continuous  
delivery

Infrastructure as  
code

Static analysis





# DevOps?

## Culture

Ownership

Organization

Individuals

## Practices

Architecture  
patterns

Small, frequent  
updates

Coding practice

Microservice  
patterns

Code packaging

Caching and object  
access patterns

Service discovery

Circuit breaker

Secrets  
management

Code promotions

SCM

Continuous  
integration

Continuous  
delivery

Infrastructure as  
code

Static analysis

## Tools

CodeCommit  
/ Git

CodeBuild  
/ Jenkins

CodeDeploy  
/ Spinnaker

CodePipeline

Cloud9



# 지속적 통합과 배포, 전달

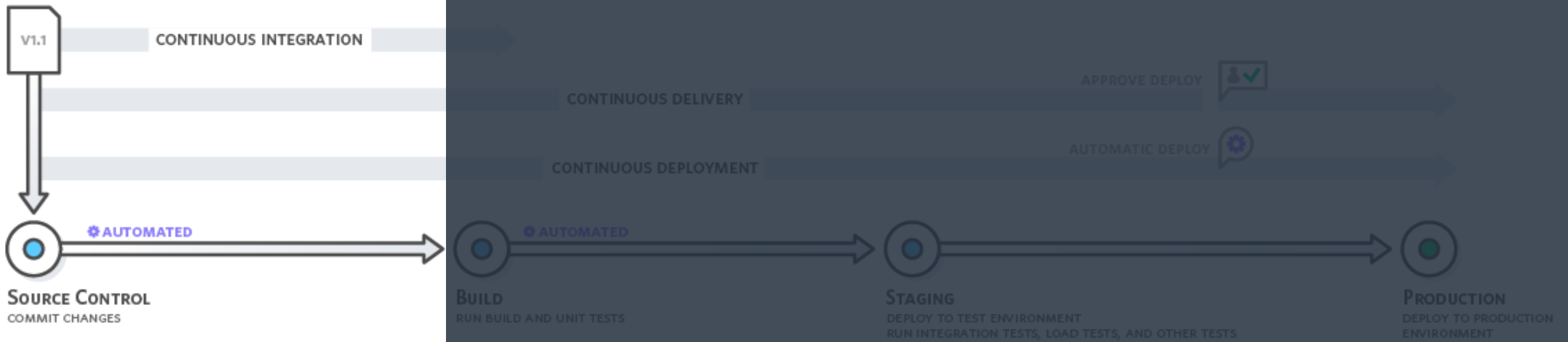
# 더욱 빠른 변화와 높은 성과

자동화 된 소프트웨어 Delivery를 갖춘 경우,

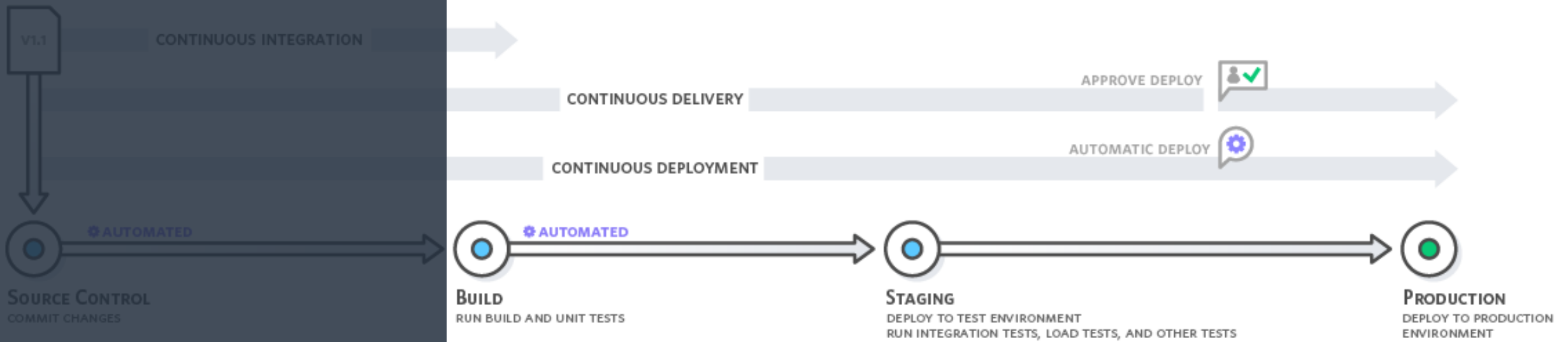
배 포 주 기	주간-월간	시간-일간
변 화 소 모 시 간	1-6 개월	1-7 일
장 애 확 률	46-60%	0%-15%

Source: 2019 DORA State of DevOps report

# CI/CD는 어떤 의미인가요?



# CI/CD는 어떤 의미인가요?



# 지속적 통합

- 중앙 저장소로의 병합
- 자동화 된 테스트를 통한 품질 검증
- 코드리뷰를 통한 개발 문화 활성화
- 개발자에게 직접적인 피드백



# 코드 리뷰

Developer Tools

CodeCommit

▼ Source • CodeCommit

Getting started

Repositories

Code

Pull requests

Commits

Branches

Tags

Settings

► Build • CodeBuild

► Deploy • CodeDeploy

► Pipeline • CodePipeline

Developer Tools > CodeCommit > Repositories > MyDemoRepo > Pull requests > Create pull request

Create pull request

Destination  
master

Source  
feature-randomizati...

Compare

Cancel

Not mergeable

There are conflicts between master and feature-randomizationfeature that cannot be merged in the AWS CodeCommit console. You can create this pull request, but you must merge the branches in Git.

Details

Title  
My Pull Request  
150 characters maximum

Description  
This pull request contains the code for the randomization feature as well as a draft tutorial for users. I'd like to get this merged before the end of the sprint. Review and comment in the next 72 hours and let me know what you think.

Changes

Commits



< Page 1 of 1 > Go to file

biobtest.txt Deleted





© 2023, Amazon Web Services, Inc. or its affiliates.

# 코드 리뷰

```
48 +  
49 +     Region region = Region.US_WEST_2;
```

 **Saanvi\_Sarkar** commented 3 minutes ago  **Edit** **Delete**

Should we add a comment here to help students understand that they should adjust the AWS Region value?

**Reply**    1  1

**New comment**

**Li\_Juan**  
**View all reactions**

```
50 +     S3Client s3 = S3Client.builder().region(region).build();  
51 +     DeleteBucketPolicyRequest delReq = DeleteBucketPolicyRequest.builder()  
52 +         .bucket(bucket_name)  
53 +         .build();  
54 +     try {
```



# AWS CodeCommit



- 완전 관리형 안전한 Git 기반의 레포지토리
- 안전하고 확장 용이한 환경에서 팀원들간에 Code 협업툴
- 전송 중 저장시 자동으로 파일 암호화 제공
- IAM과 통합된 계정 관리

# 3rd Party Code Repositories

**GitHub**



# 지속적 배포 및 전달

- 코드 변경사항을 자동으로 반영
- 빌드 후 테스트/프로덕션 환경에 변경사항을 배포
- 애플리케이션이 더욱 견고해짐
- 빠르게 실패하고, 빠르게 롤백
- 표준화 된 테스트 프로세스를 통과한 빌드 아티팩트가 생성



# AWS CodeBuild



- 소스 코드를 컴파일하는 단계부터 테스트 실행 후, 소프트웨어 패키지를 개발하여 배포하는 단계까지 마칠 수 있는 완전 관리형 빌드 서비스
- 빌드 볼륨에 따라 자동으로 확장 및 축소
- 빌드를 완료할 때까지 걸리는 시간(분)을 기준으로 과금
- 일관되고 불변하는 환경을 위해 격리된 빌드 컨테이너
- 모든 공식 CodeBuild 이미지는 Docker와 AWS CLI를 포함
- 빌드 환경 커스터마이징 가능

# AWS CodeBuild의 buildspec 파일

```
version: 0.2

env:
  variables:
    JAVA_HOME: "/usr/lib/jvm/java-8-openjdk-amd64"
phases:
  install:
    runtime-versions:
      java: corretto8
  build:
    commands:
      - echo Build started on `date`
      - mvn install
  post_build:
    commands:
      - echo Test started on `date`
      - mvn surefire-report:report
reports:
  SurefireReports:
    files:
      - '**/*'
    base-directory: 'target/surefire-reports'
artifacts:
  type: zip
  files:
    - target/messageUtil-1.0.jar
discard-paths: yes
```



빌드 단계에서 사용된 변수



빌드 시, 실행하는 명령



빌드 후, 명령 - 유닛 테스트 실행



리포트의 위치



Amazon S3에 빌드 아티팩트 저장

```

1  version: 0.2
2
3  phases:
4    pre_build:
5      commands:
6        - echo Logging in to Amazon ECR...
7        - aws --version
8        - AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
9        - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --username AWS --password-stdin $AWS_ACCOUNT_ID
10       - REPOSITORY_URI=$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/builders-app
11       - COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
12       - IMAGE_TAG=${COMMIT_HASH:=latest}
13    build:
14      commands:
15        - echo Build started on `date`
16        - chmod +x ./gradlew
17        - ./gradlew test --stacktrace
18        - ./gradlew bootJar
19        - echo Building the Docker image...
20        - docker build -t $REPOSITORY_URI:latest .
21        - docker tag $REPOSITORY_URI:latest $REPOSITORY_URI:$IMAGE_TAG
22    post_build:
23      commands:
24        - echo Build completed on `date`
25        - mkdir /codebuild/output/tmp/$CODEBUILD_RESOLVED_SOURCE_VERSION
26        - echo Pushing the Docker images...
27        - docker push $REPOSITORY_URI:latest
28        - docker push $REPOSITORY_URI:$IMAGE_TAG
29        - echo Update the taskdef.json
30        - cat script/taskdef.json
31        - cat script/taskdef.json | sed -e "s/<ACCOUNT_ID>/${AWS_ACCOUNT_ID}/g" -e "s/<REGION>/${AWS_DEFAULT_REGION}/g" >
32        - cat taskdef.json
33        - echo Writing image definitions file...
34        - printf '["name":"user-app","imageUri":"%s"]' $REPOSITORY_URI:$IMAGE_TAG > imagedefinitions.json
35  artifacts:
36    files:
37      - imagedefinitions.json
38      - appspec.yaml
39      - taskdef.json

```

빌드 전 실행하는 명령

- AWS Account ID 가져와서 Docker, ECR repo, Image Tag 등 설정

빌드 시, 실행하는 명령

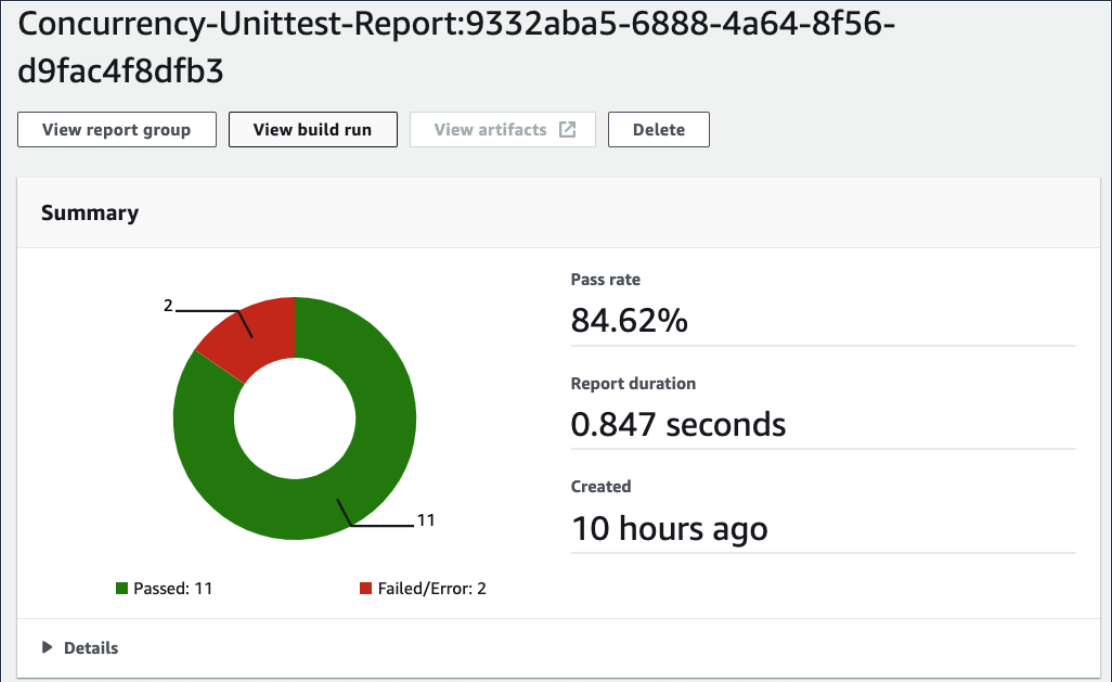
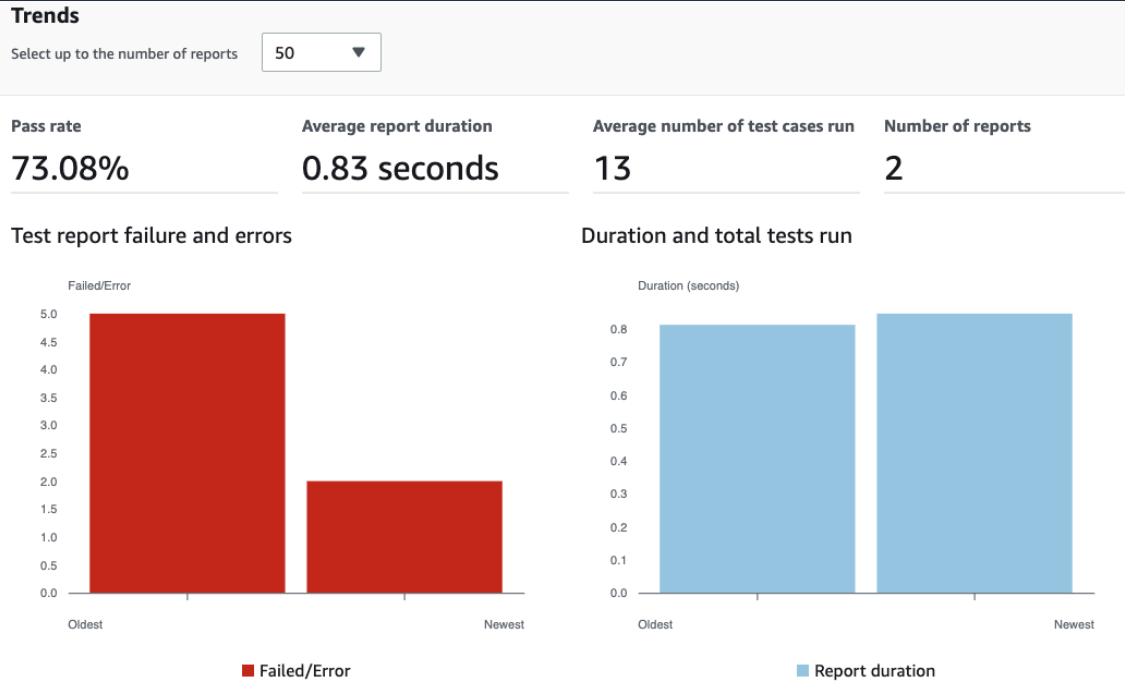
- Gradle 빌드, Unit Test
- Docker 빌드

빌드 후, 명령

- 도커 이미지 ECR에 푸시

빌드 아티팩트 정의 및 저장

# AWS CodeBuild Report



# 3rd Party Build



GitHub Actions





# AWS CodeDeploy



- 코드 배포를 자동화하는 완전 관리형 배포 서비스
- 복잡한 애플리케이션 업데이트 작업을 처리
- 배포 중, 다운타임 최소화
- 배포 중, 오류 감지 시, 자동으로 롤백
- Amazon EC2, AWS Lambda, Amazon ECS, 온프레미스 서버에 배포

# AWS CodeDeploy - Amazon EC2 배포

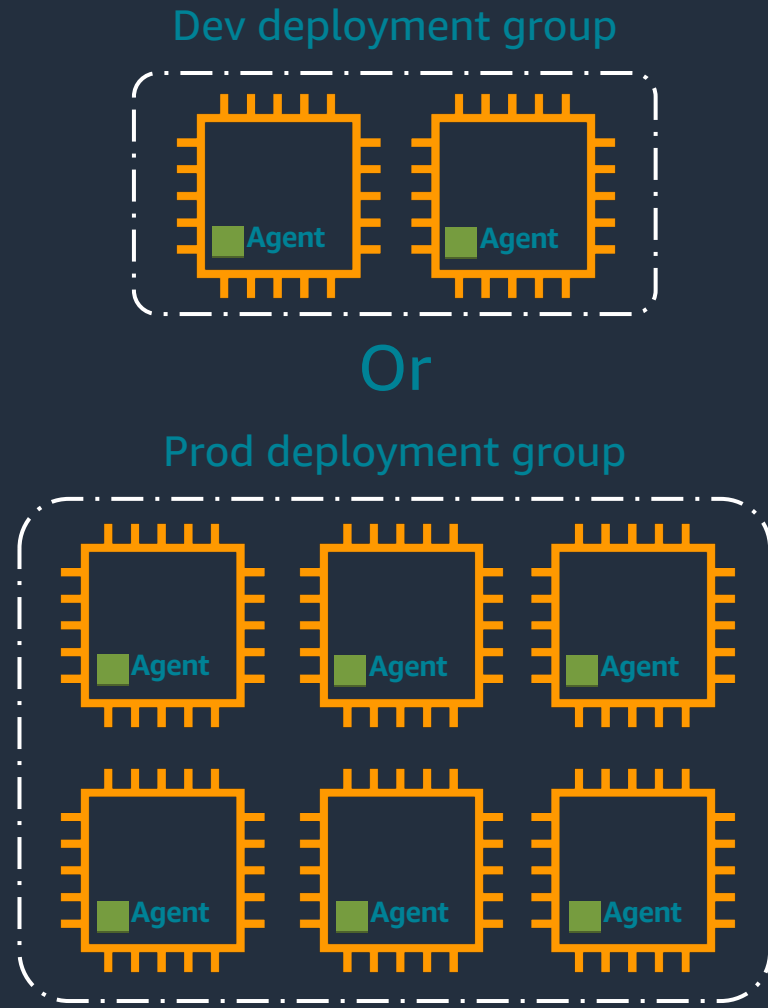
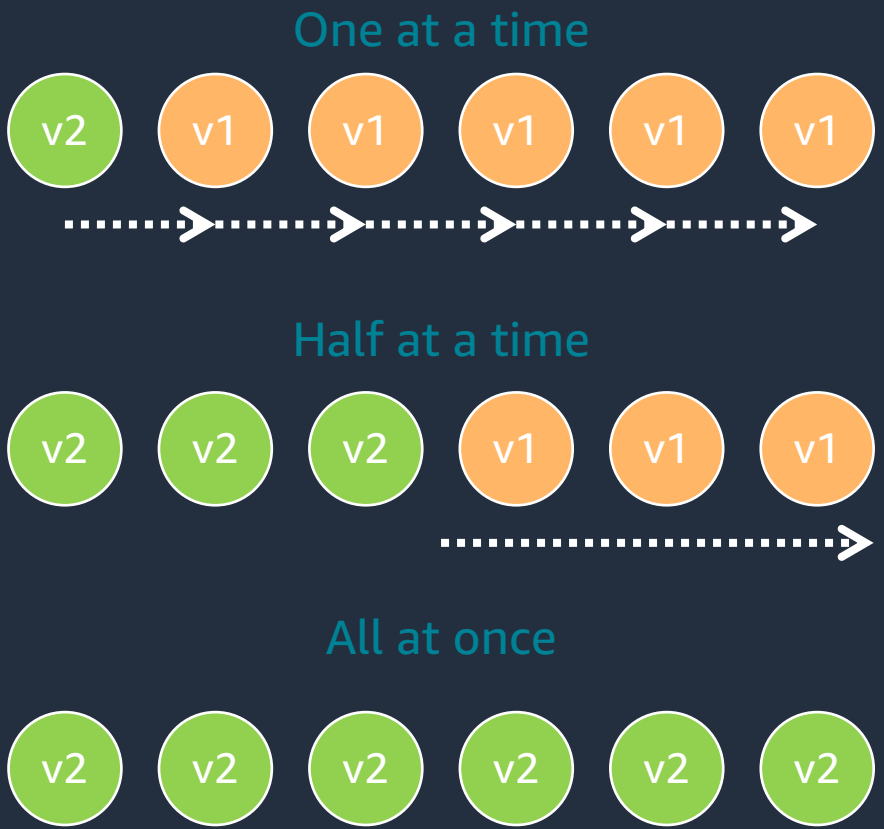
```
version: 0.0
os: linux
files:
  - source: /
    destination: /var/www/html
permissions:
  - object: /var/www/html
    pattern: "*.html"
    owner: root
    group: root
    mode: 755
hooks:
  ApplicationStop:
    - location: scripts/deregister_from_elb.sh
  BeforeInstall:
    - location: scripts/install_dependencies.sh
  ApplicationStart:
    - location: scripts/start_httpd.sh
  ValidateService:
    - location: scripts/test_site.sh
    - location: scripts/register_with_elb.sh
```

} 디렉토리로 응용 프로그램 파일 전송 및  
다른 디렉토리로 구성 파일을 전송

} 특정 디렉토리 및 파일에 대한 특정 사  
용 권한 설정

} 로드밸런서에 인스턴스 제거 및 추가  
디펜던시 패키지 설치  
웹 서버 시작  
배포가 성공적으로 이뤄졌는지 확인

# AWS CodeDeploy – 배포 속도 및 배포 그룹



# CI/CD Pipeline을 통해,

- Application과 **Infrastructure**의 최신상태 유지
- Workflow 시각화
- 다른 Pipeline을 제어
- 각 단계에 따른 Log를 분석하여 배포 프로세스 디버깅
- 공용 환경에서의 일관성 있는 빌드/배포
- “배포담당자”의 실종

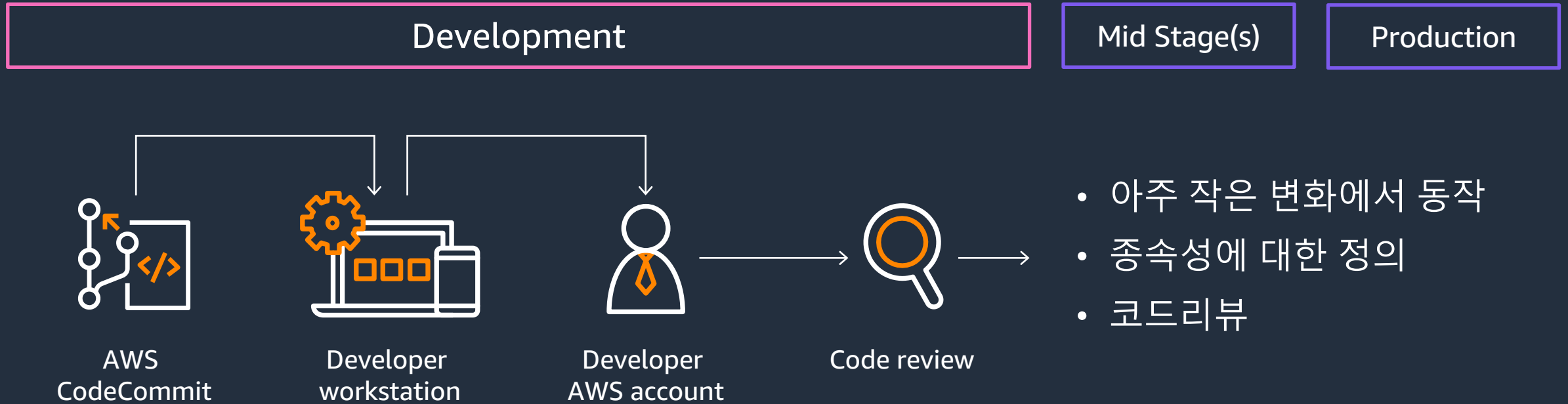
# CI/CD Pipeline 구성

- 빌드 및 배포의 단계를 나누어 구성
- 각 단계는,
  - 병렬 수행 가능
  - 전 단계 성공 시 수행
  - Pipelined이 실패할 수 있음을 고려
- 빌드에 필요한 Container Image를 찾거나, 제작
- 단계별 생산되는 Artifact를 다양하게 활용

# CI/CD Pipeline 실행

- Git branch 에 Push가 발생했을 때
- Pull/Merge Request가 승인되었을 때
- 수동 실행 (API, Web Console)
- 스케줄 기반
- 패키지 파일 또는 컨테이너 이미지가 제작되었을 때

# CI/CD 파이프라인



빨라야 합니다

# CI/CD 파이프라인



- 빌드 및 유닛 테스트 수행
- 코드 및 Runtime 의존성을 하나의 Artifact로

철저해야 합니다



# CI/CD 파이프라인

Development

Mid Stage(s)

Production

성공 시에만 다음 단계로, 테스트 실패 시 롤백, 베이킹 실패 중지



## Beta

- 다른 환경에는 영향을 미치지 않음
- 다양한 테스트 수행

## Gamma

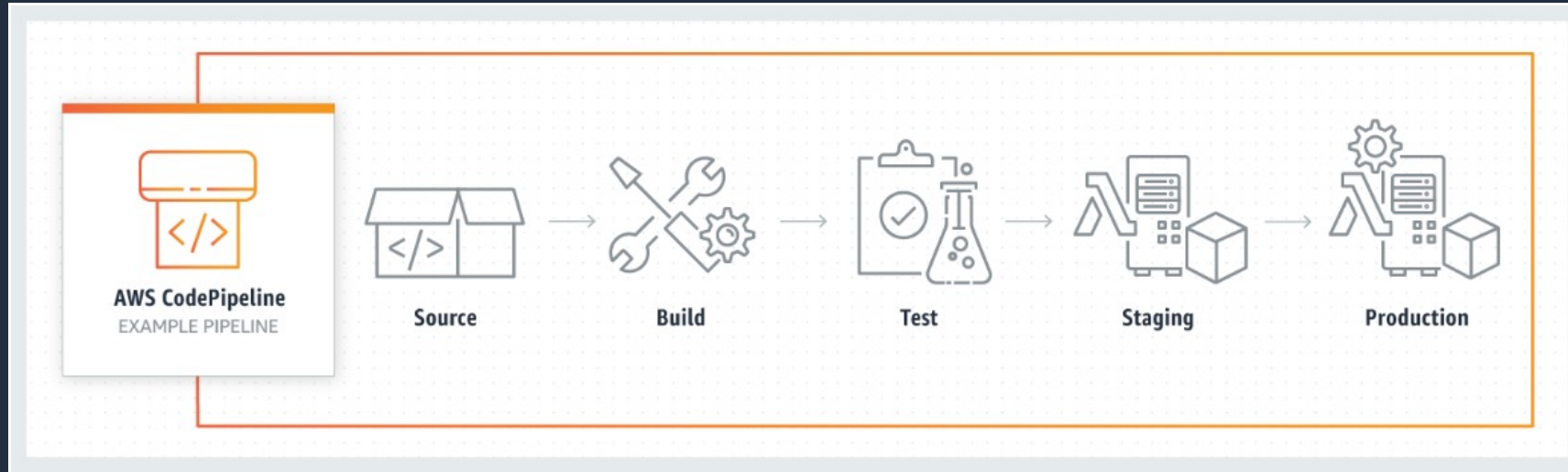
- 격리되어있지만 운영과 유사하게
- 종단간 통합 테스트 수행

## Production

- 조금씩 배포하며 확장
- 점진적으로 배포
- 항상 모니터링 하고, 민감하게 반응할 것

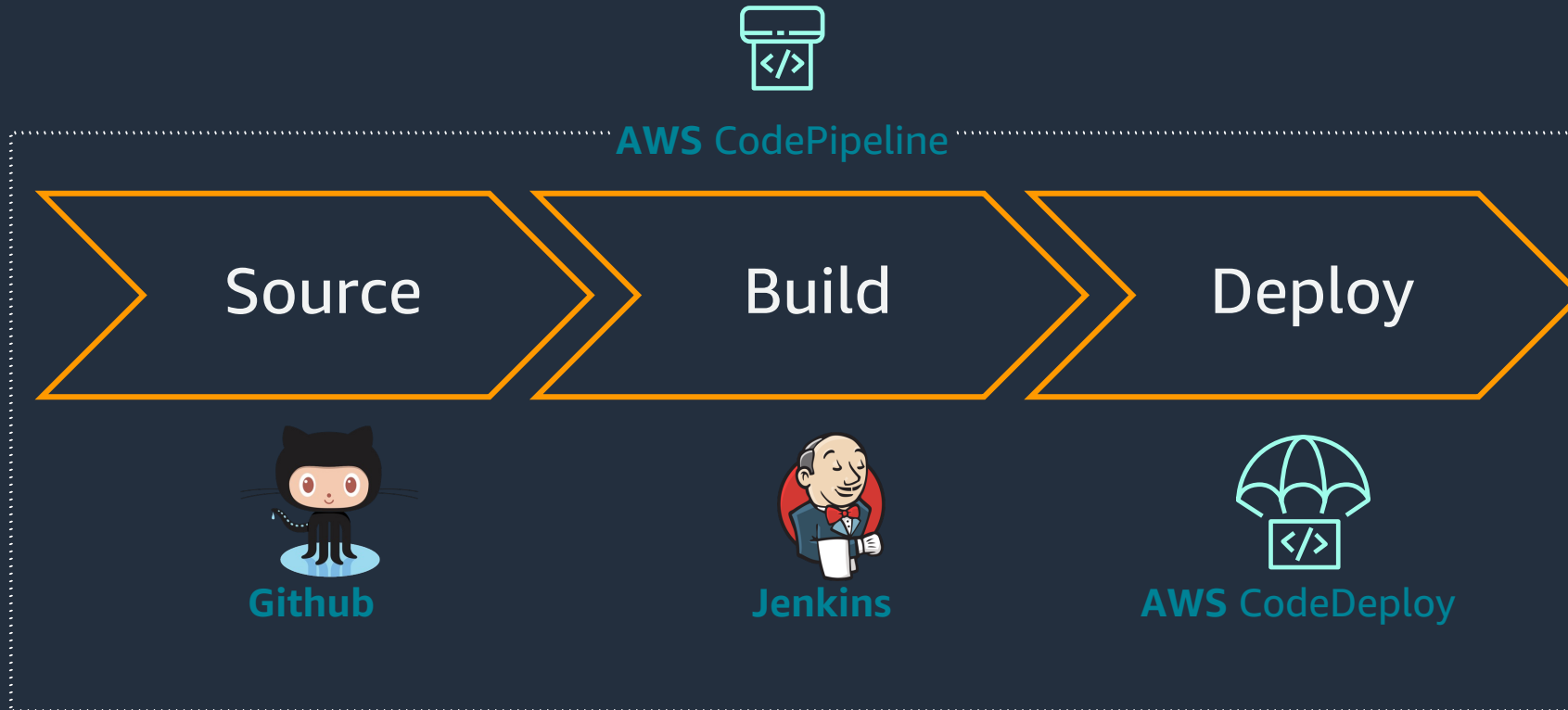
실패한 이유를 발견합니다

# AWS CodePipeline

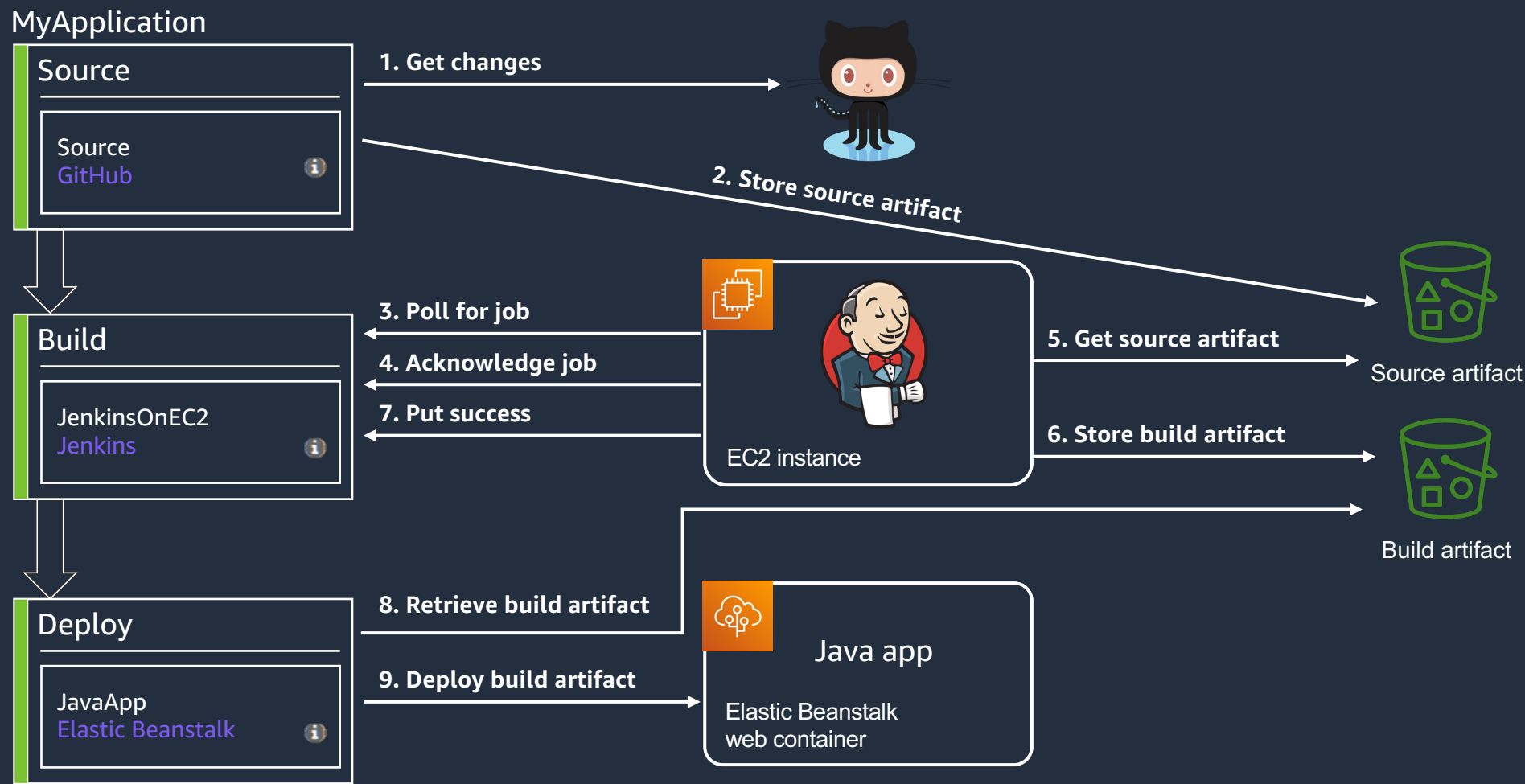


- 빠르고 안정적인 애플리케이션 업데이트를 위한 지속적인 제공 서비스
- 소프트웨어 릴리스 프로세스 모델링 및 시각화
- 코드가 변경될 때마다 코드를 빌드, 테스트 및 배포합니다.
- 타사 도구 및 AWS와 통합

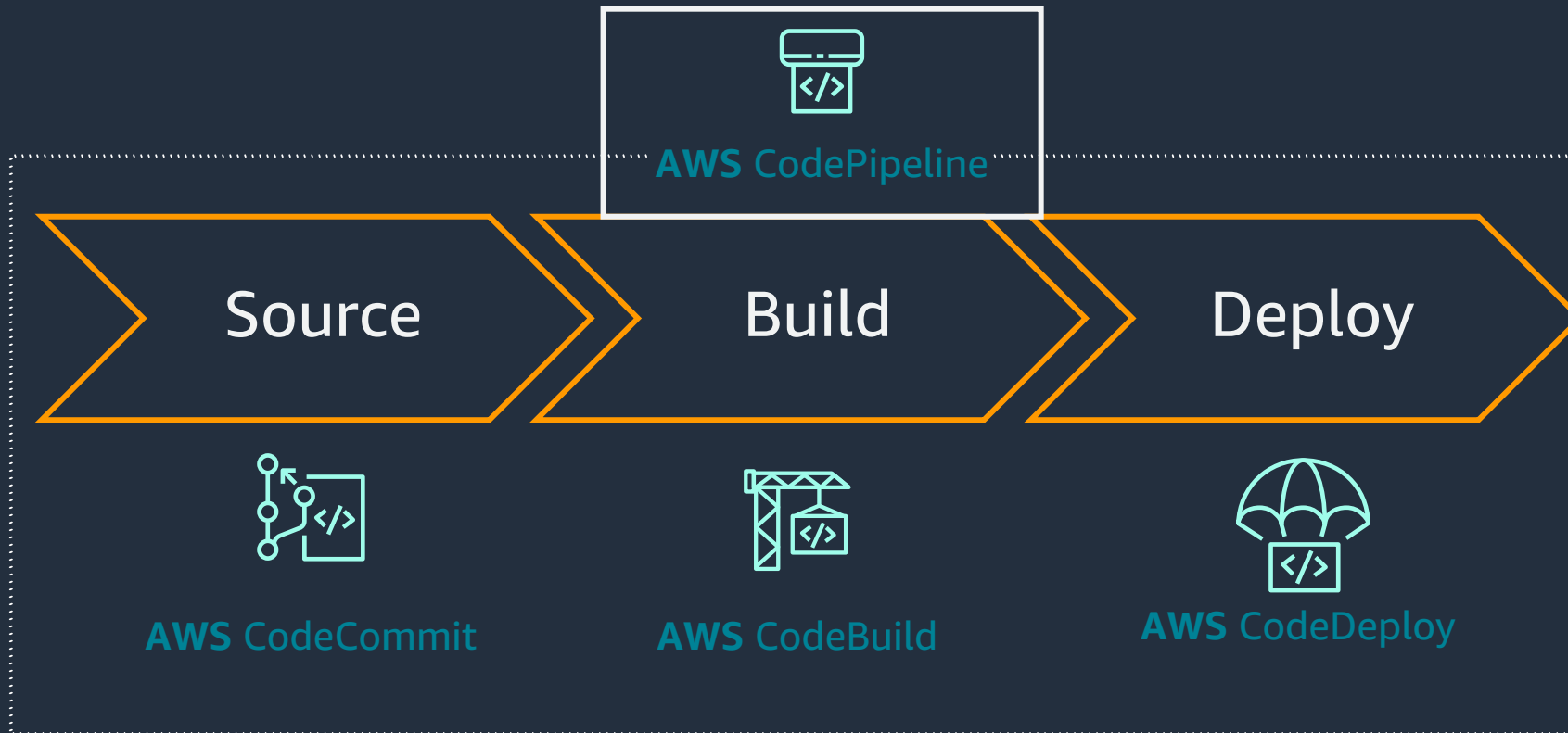
# Sample Pipeline - CodePipeline with 3<sup>rd</sup> party



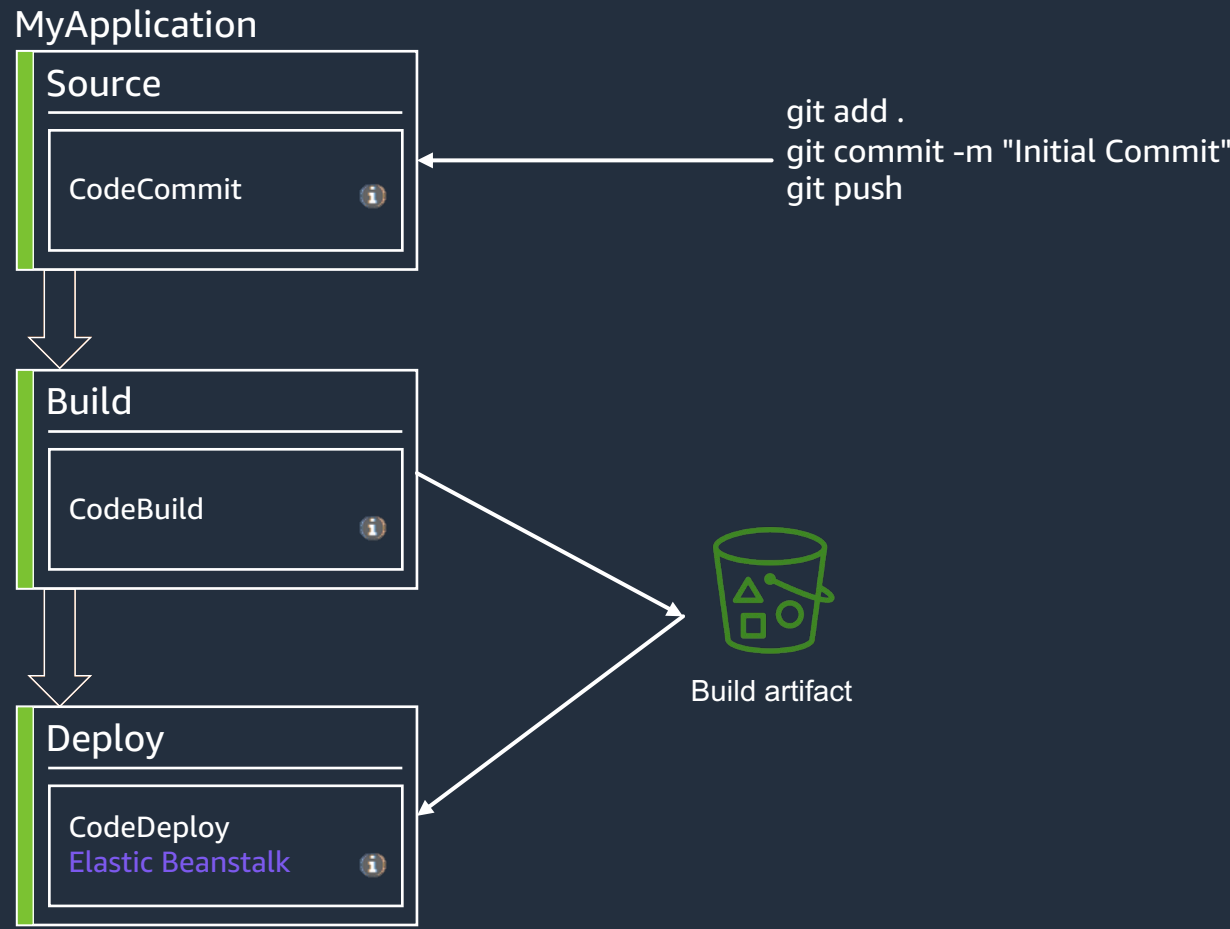
# Sample Pipeline with 3<sup>rd</sup> party



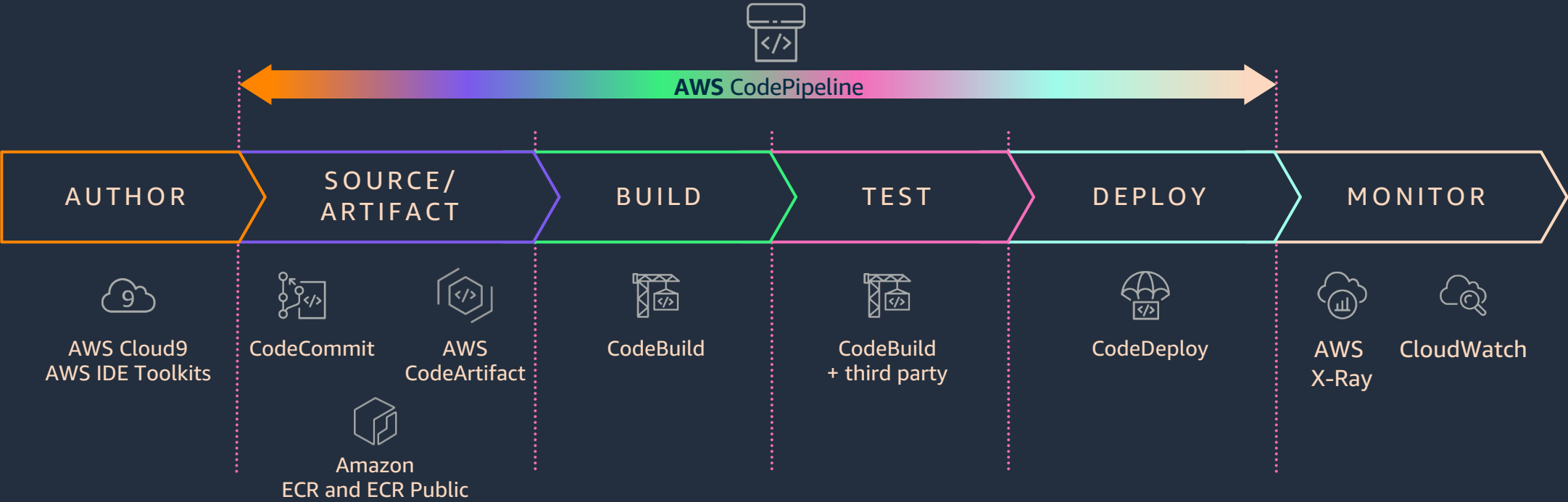
# Sample Pipeline - AWS Developer tools







# Sample Pipeline with AWS Developer Tools



# AWS Developer Tools



**MODEL**  **AWS CloudFormation**  **AWS Cloud Development Kit (CDK, CDK8s, CDK-terraform)**  **AWS Amplify**  **AWS Copilot** **Docker Compose**



# Thank you!