



# Builders Online Series

## DevOps 문화와 모범 사례 및 필수 도구

박선준 ( June Park )

AWS 솔루션즈 아키텍트

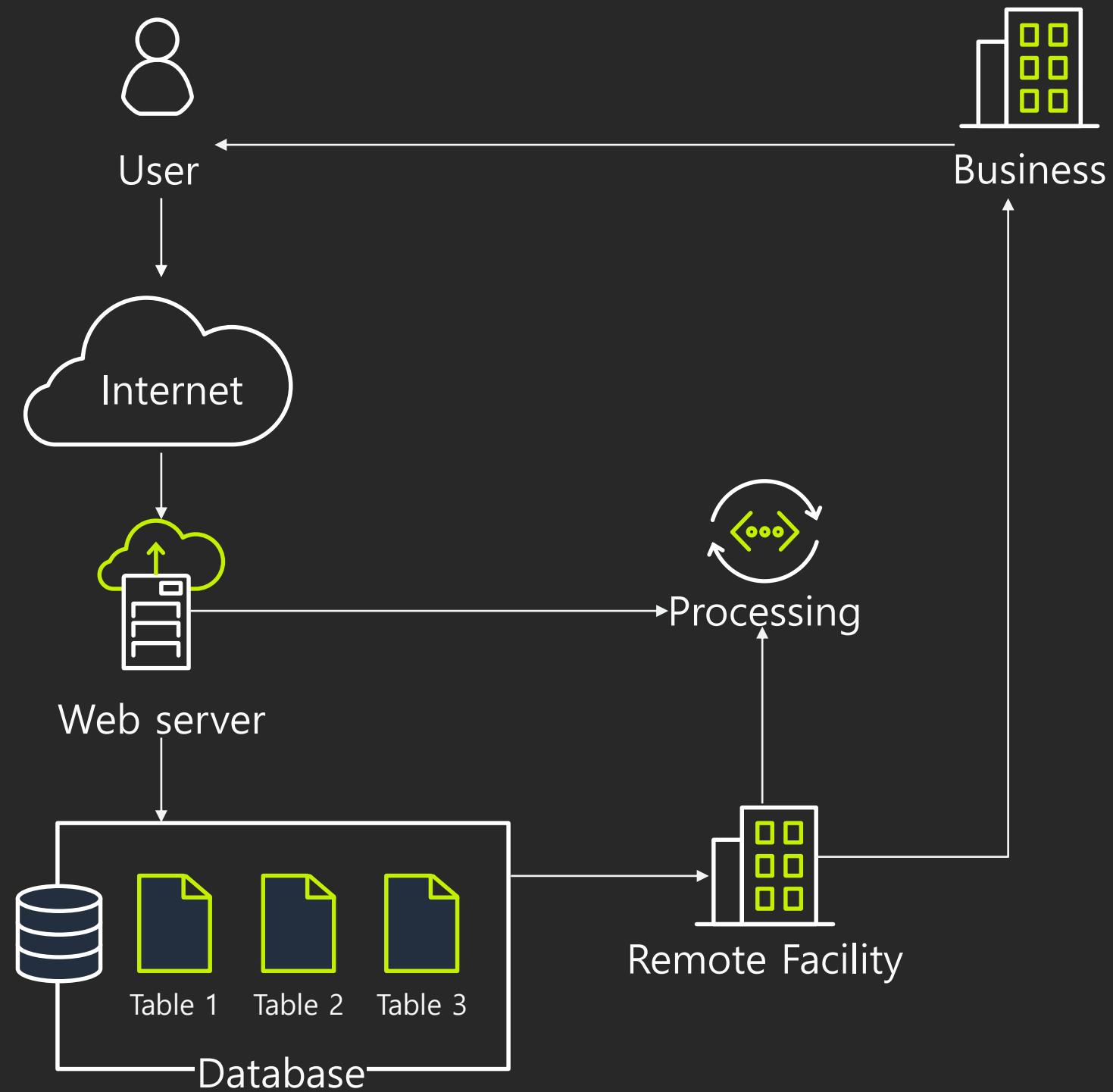
# Agenda

- 아마존의 DevOps
- 모범 사례
- DevOps 도구

# 아마존의 DevOps 문화



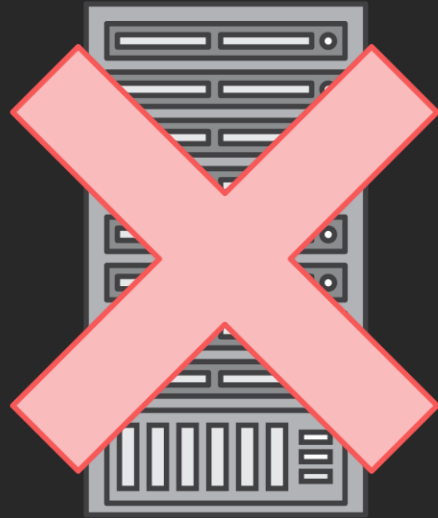
# 2000년대 초반의 Amazon



# 2000년대 초반의 Amazon



확장성 부재



컴포넌트  
문제로 전체  
시스템 장애

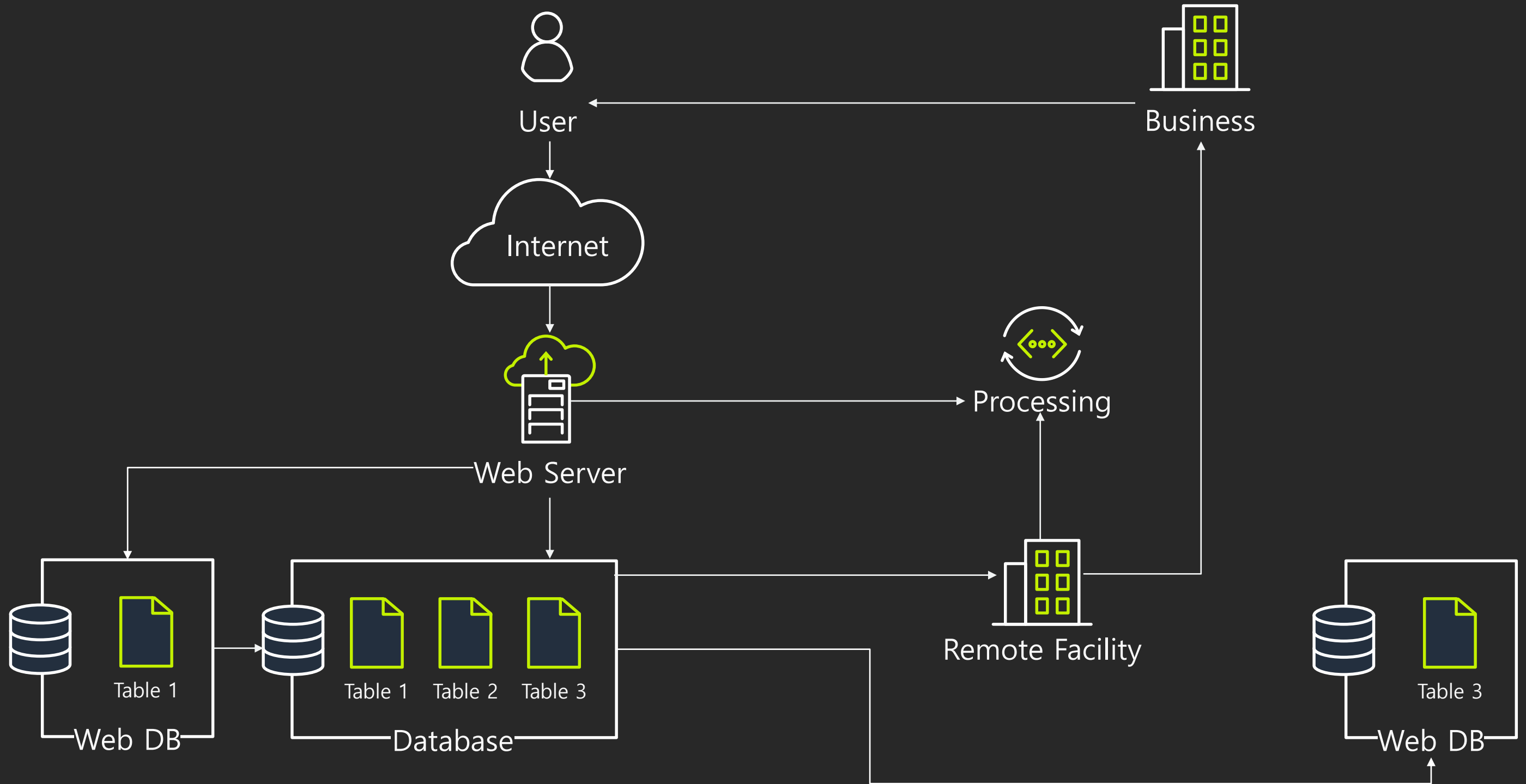


느린 배송  
속도

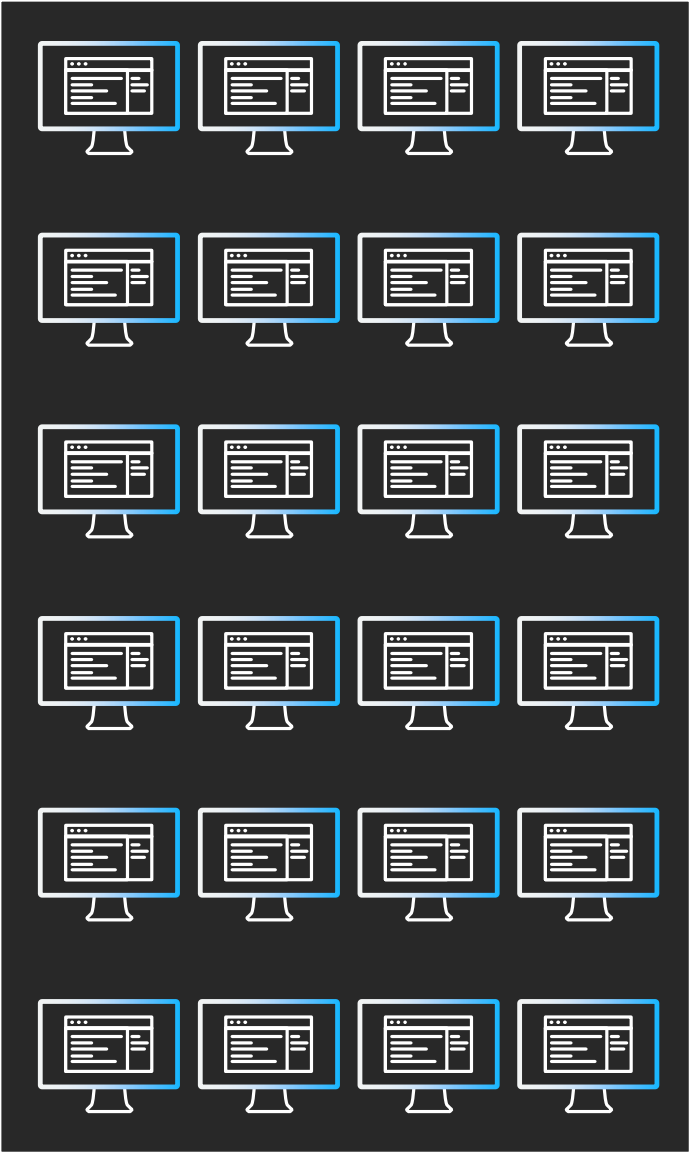


다양성 부족

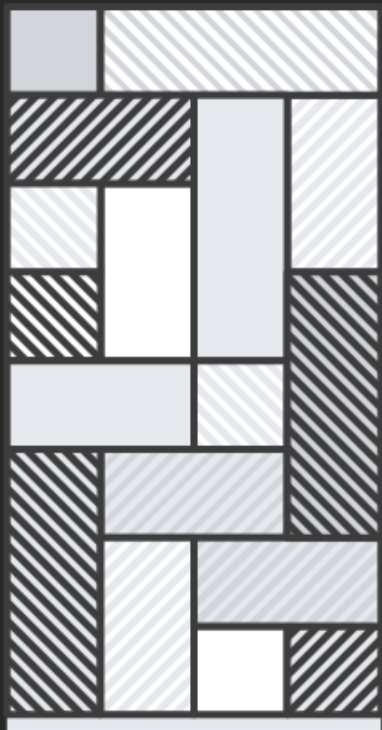
# 2000년대 초반의 Amazon



# 서비스 진화



개발팀



어플리케이션

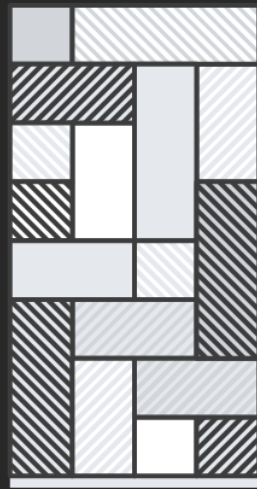


배포 파이프라인

# 서비스 진화

## SOA – Service Oriented Architecture

- 서비스 지향 아키텍처의 도입으로 서비스를 분리
- 애자일 개발 방식 도입
- 개발 속도 향상
- 작은 장애 범위 구성

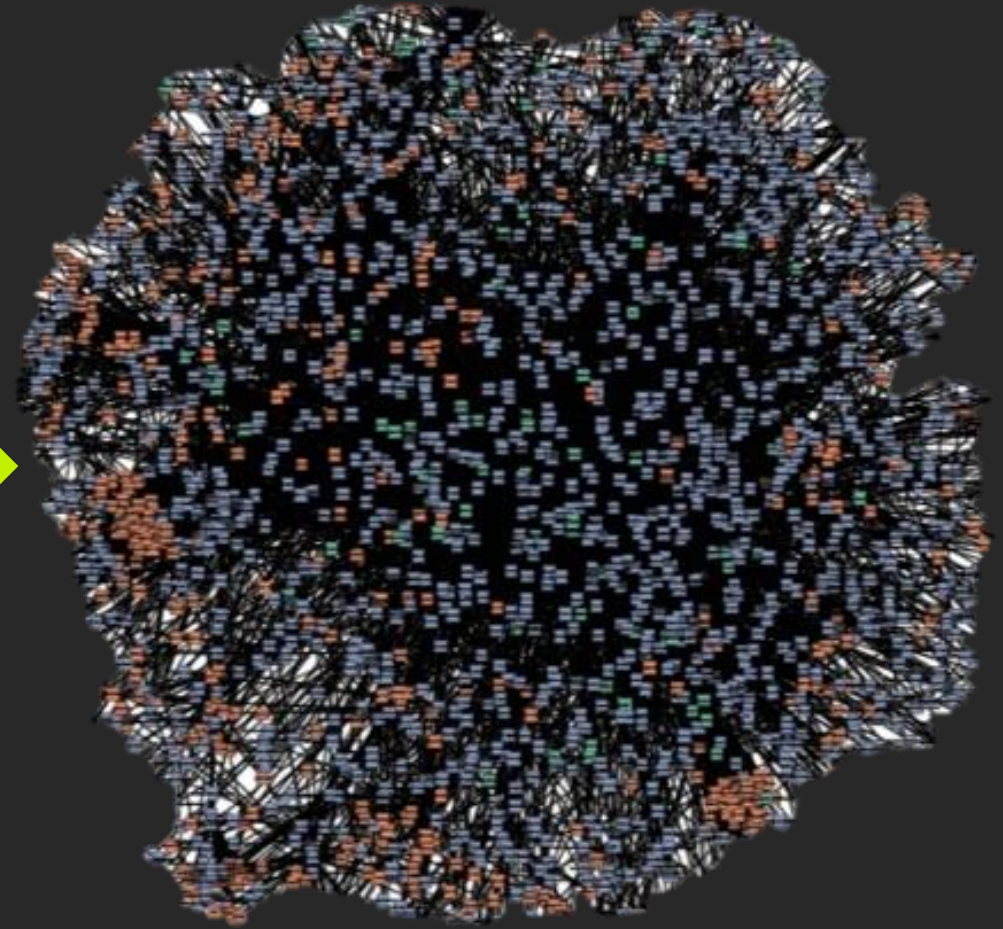
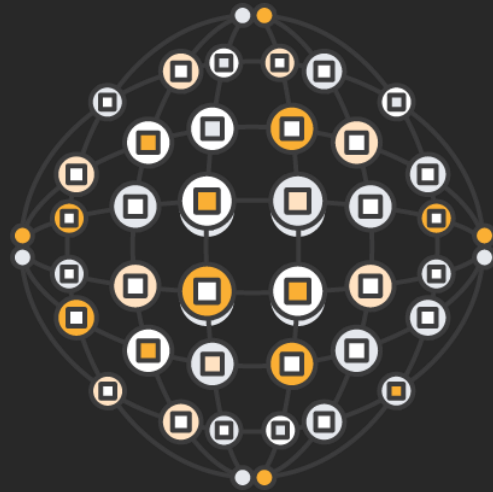




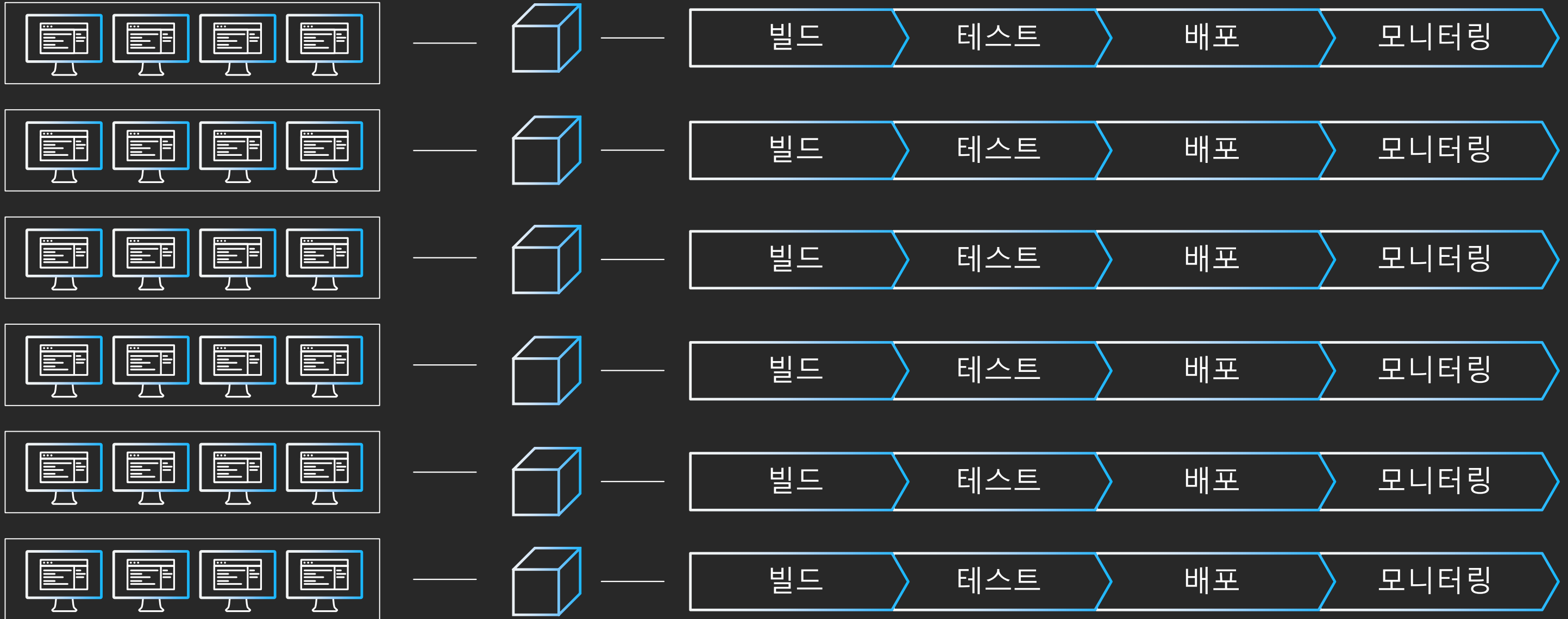
# 서비스 진화

SOA 의 한계 그리고 MSA 도입

- 고객 지향적
- 서비스에 대한 책임 부여
- 빠른 혁신
- 빠른 움직임



# 지속적인 진화



DevOps팀

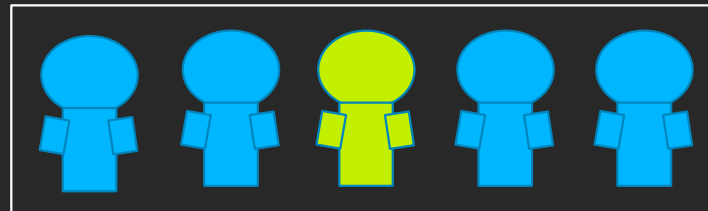
어플리케이션

배포 파이프라인

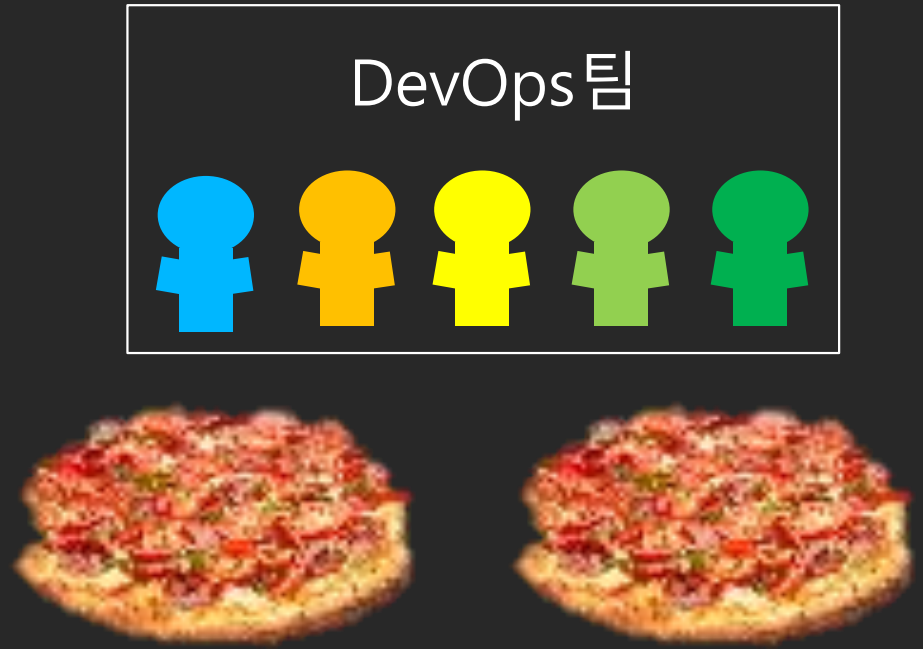
# 2-Pizza 팀

## Amazon Software Development Team


few developers (SDEs) + manager (SDM)

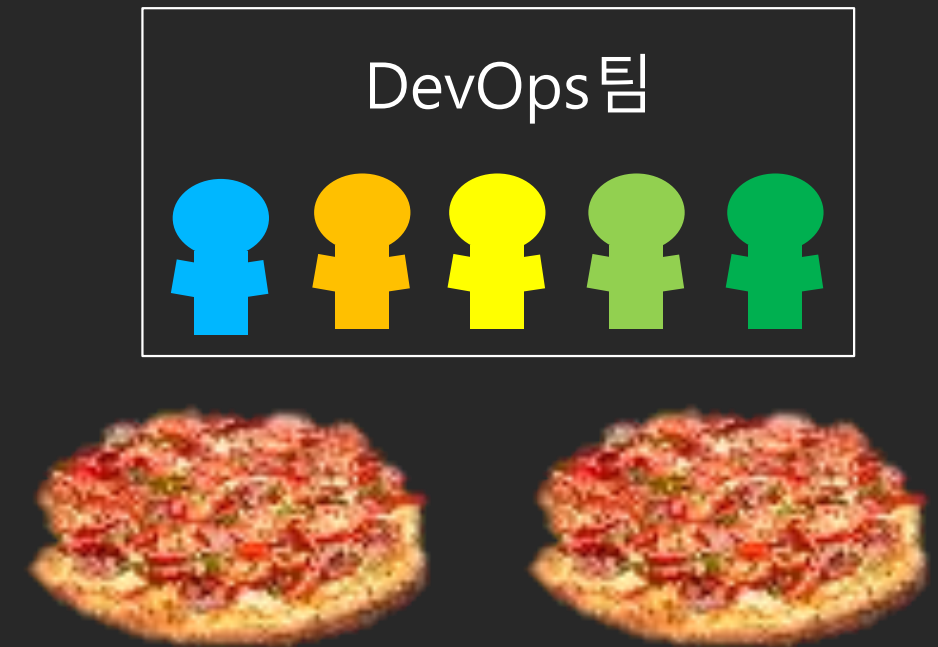


# 2-Pizza 팀



## 2-Pizza 팀의 책임 및 역할

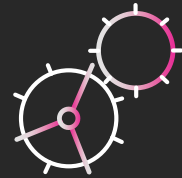
- 어플리케이션 코드 개발
- 팀 내부 및 외부 코드 리뷰
- 유닛/통합/성능 테스트 시나리오 개발
- Database schemas 와 SQL쿼리 생성
- CI/CD 파이프라인으로 자동화
- 테스트와 운영 환경 구축
- 보안 APIs 사용 및 보안 정책 적용
- 운영 환경 모니터링
- 돌아가며 on-call 실행 



# DevOps Best Practices



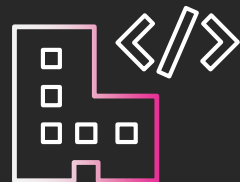
# Best Practices



자동화



Belts and suspenders  
*(governance, templates)*

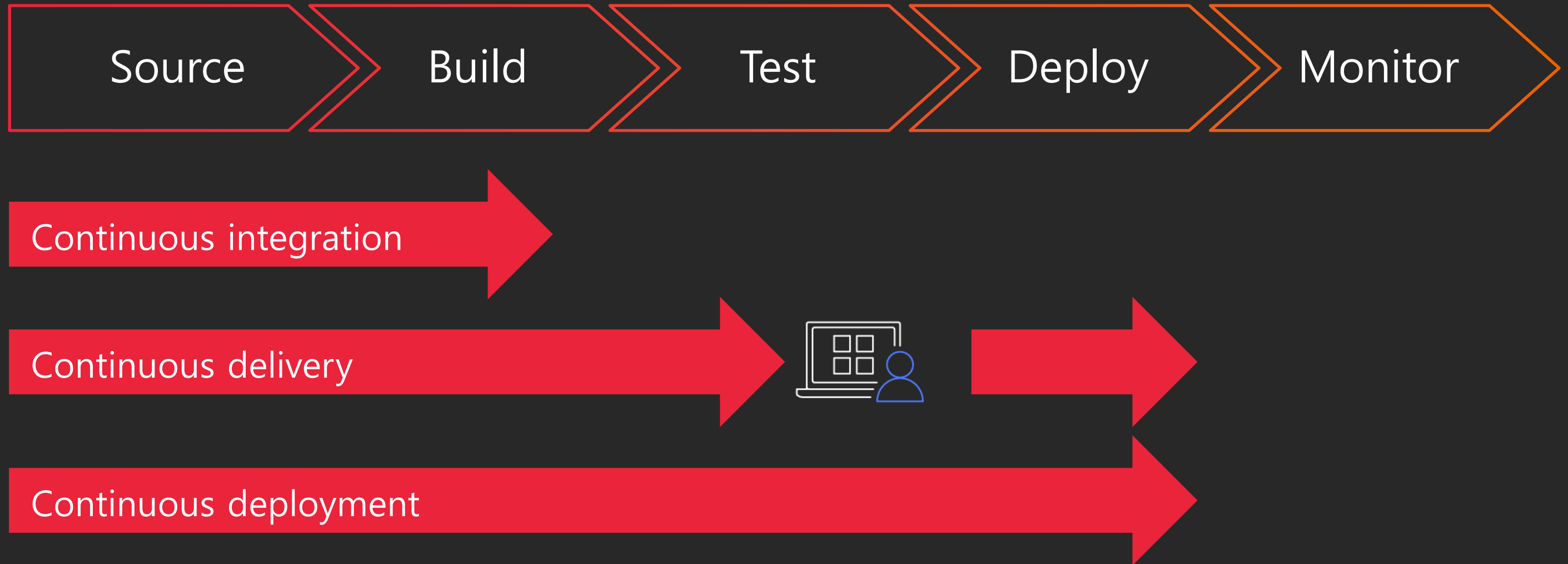


인프라 코드화 (IaC)



표준화 툴

# Automate everything





# Continuous Integration



Continuous integration

1. 새로운 코드가 소스 저장소에 체크인되면 자동으로 새 빌드 시작
2. 일관되고 반복 가능한 환경에서 코드 작성 및 테스트
3. 아티팩트를 지속적으로 배포할 수 있도록 준비
4. 빌드 실패 시 피드백 루프 진행 및 해결

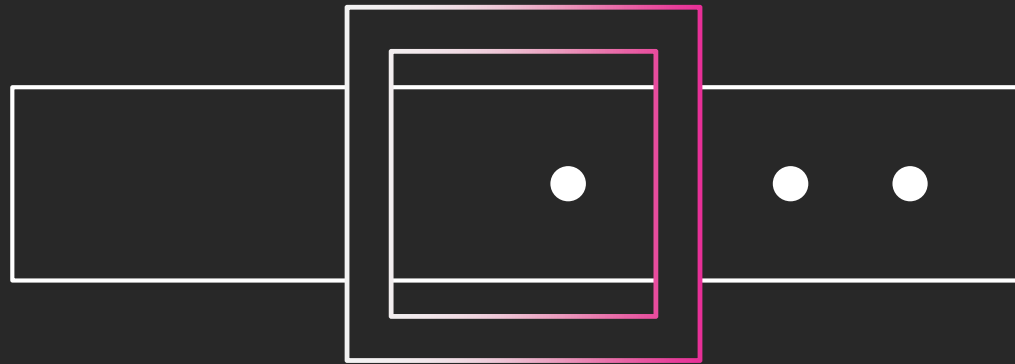
# Continuous Delivery/Deployment



Continuous deployment

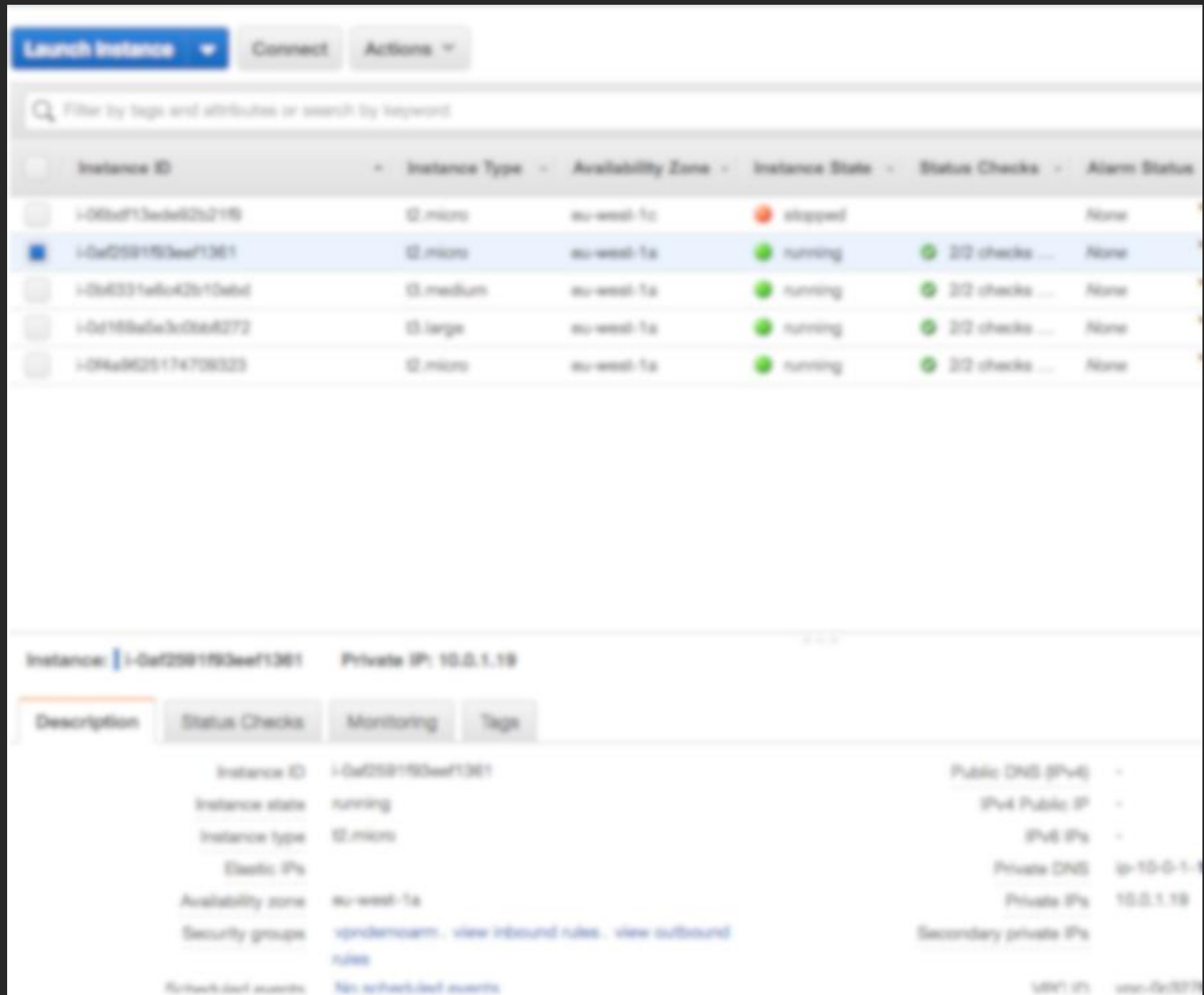
1. 테스트를 위해 스테이징 환경에 새로운 변경 사항을 자동으로 배포
2. 고객이나 서비스에 영향없이 운영 환경에 안전하게 구축
3. 고객에게 보다 신속하게 서비스 제공: 구축 빈도를 높이고 변경 리드 타임 및 변경 실패율 감소

# Belts and suspenders



- Best Practices
- Blueprints
- Find anti-pattern
- Share good patterns
- Make a use of a scalable templates

# Infrastructure as a Code – 걸음마 단계



수동적

고수레벨  
↑  
초보레벨

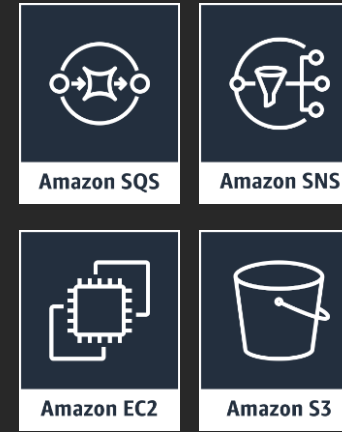
쉽게 시작 가능  
반복적인 행동 불가능  
실수 발생 가능  
오래 걸림

# Infrastructure as a Code – 걷기 단계

```
require 'aws-sdk-ec2'

ec2 = Aws::EC2::Resource.new(region: 'us-west-2')

instance = ec2.create_instances({
  image_id: 'IMAGE_ID',
  min_count: 1,
  max_count: 1,
  key_name: 'MyGroovyKeyPair',
  security_group_ids: ['SECURITY_GROUP_ID'],
  instance_type: 't2.micro',
  placement: {
    availability_zone: 'us-west-2a'
  },
  subnet_id: 'SUBNET_ID',
  iam_instance_profile: {
    arn: 'arn:aws:iam::' + 'ACCOUNT_ID' + ':instance-profile/aws-opsworks-ec2-role'
  }
})
```



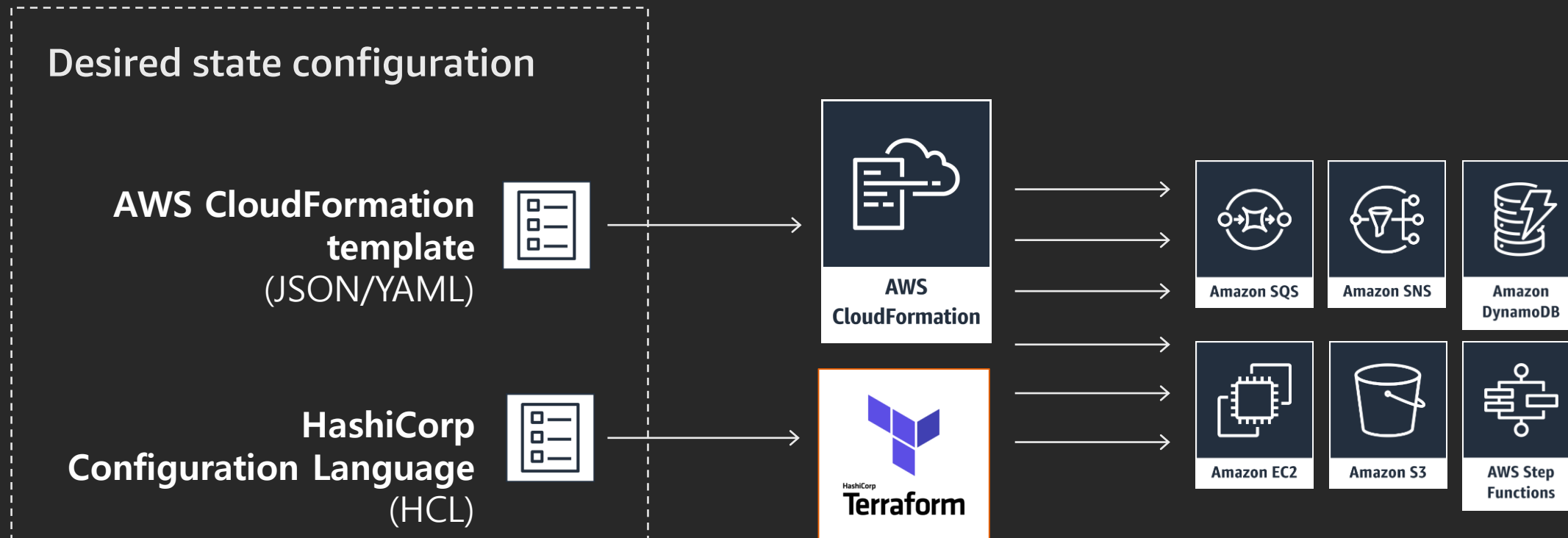
스크립트화

수동적

고수레벨  
↑  
초보레벨

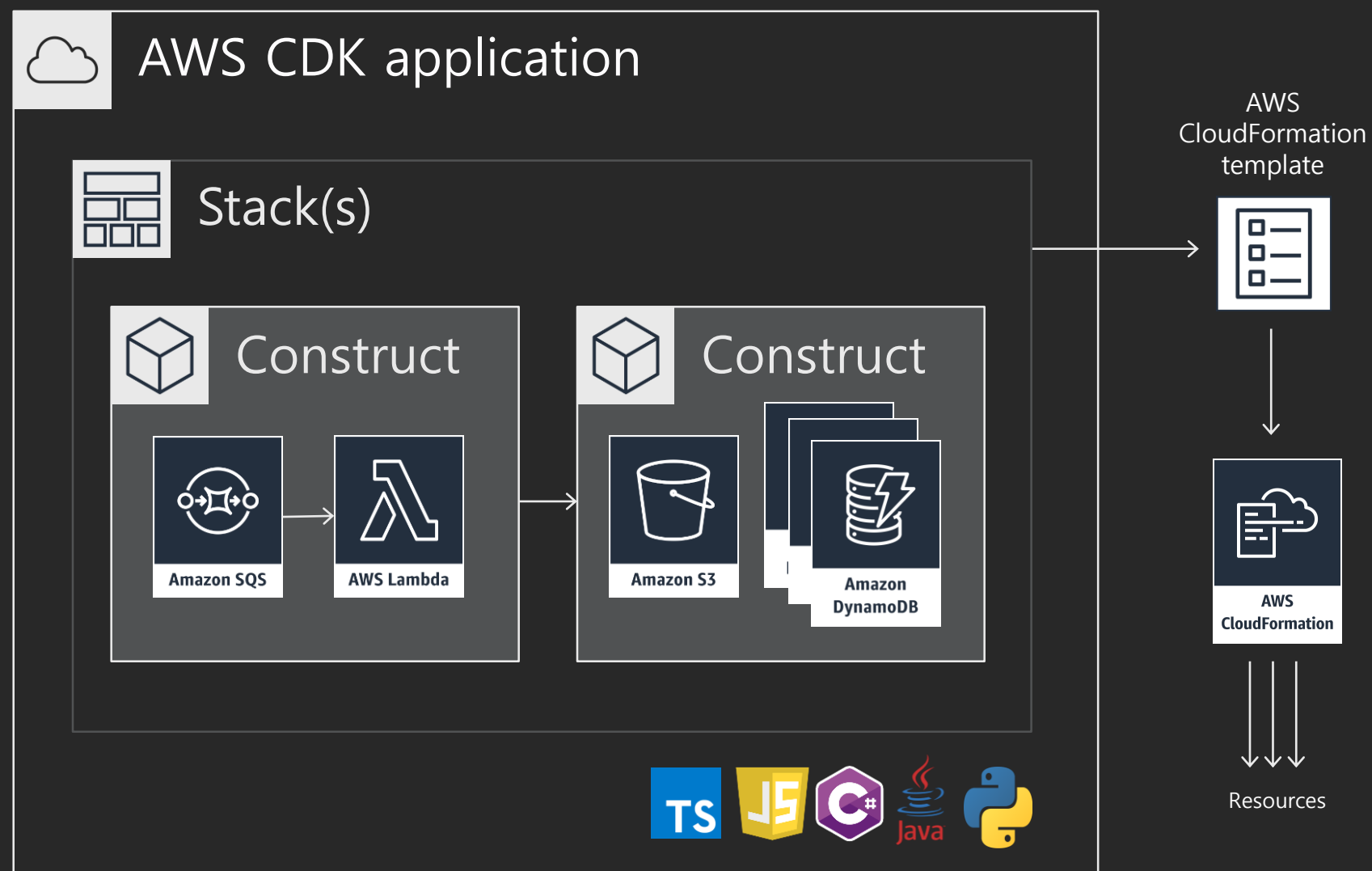
만약 API 호출이 실패한다면?  
업데이트 방법은?  
리소스 생성의 완성도 확인은?  
Roll-back 방법은?

# Infrastructure as a Code – Resource Provisioning engine



자동화  
재생성 가능  
설정 문법 이해 난이도  
추상화 불가능 및 많은 설정 내용 필요

# Infrastructure as a Code – CDK



# CDK 를 활용하여 인프라 환경 배포하기

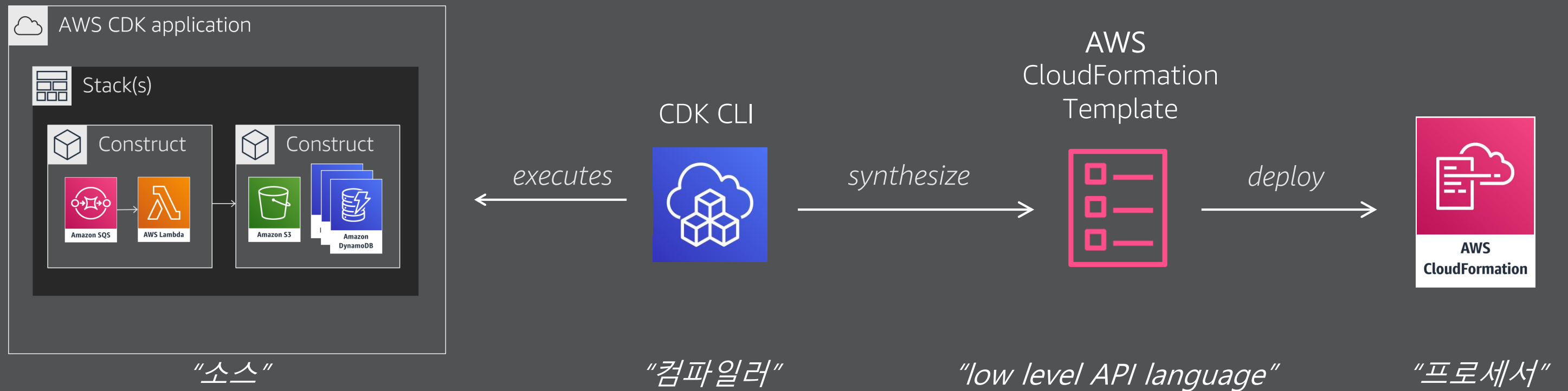
CDK: Background/AutoScalingGroup

```
//// create userdata script to install for each environment
const userData = UserData.forLinux();
const vpc = new Vpc(this, 'YourAppVpc', {
  // create static web site as S3 assets subnets in 2 AZ
  // this also creates a NAT Gateway
  vpcPath = require('path');
  instanceType: ec2.InstanceType.of(ec2.InstanceClass.BURSTABLE3, ec2.InstanceSize.MICRO),
  userData: userData.addCommands('yum install -y nginx', 'chkconfig nginx on', 'service nginx start');
  const vpc = new ec2.Vpc(this, 'YourAppVpc', {
    path: vpcPath.join(__dirname, './html')
    maxAZs: 2,
    desiredCapacity: 2,
    role: role,
    userData: userData
  });
```

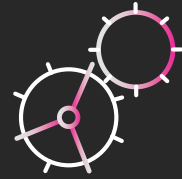


# CDK 를 활용하여 인프라 환경 배포하기

```
$ cdk deploy
```



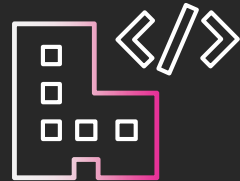
# Best Practices



자동화



Belts and suspenders  
*(governance, templates)*



인프라 코드화 (IaC)



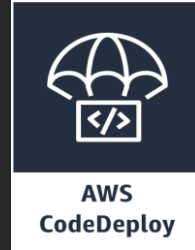
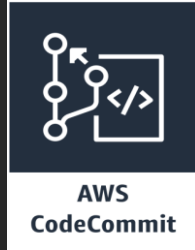
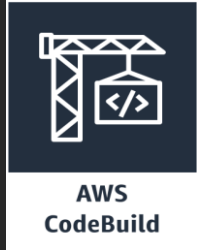
표준화 툴

# DevOps Tools

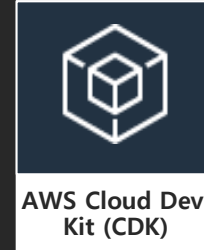
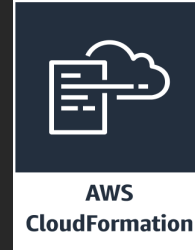


# AWS DevOps 도구들 한눈에 보기

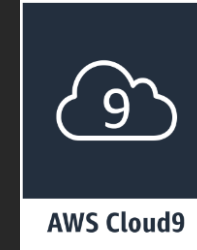
## CI/CD Tools



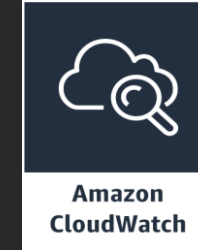
## Infrastructure as Code



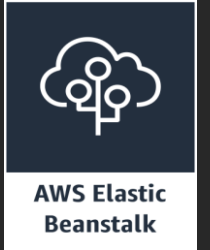
## IDE



## Monitoring & Tracing



## Web Apps



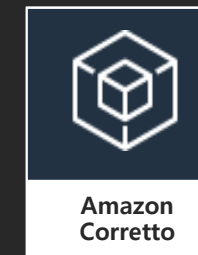
## IDE and DevOps Toolkits



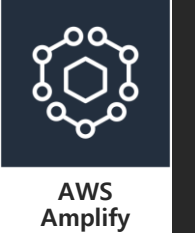
## CLI and Scripting Tools



## Languages



## Mobile



## SDKs



# AWS 서비스를 활용하여 DevOps 1단계



AWS  
Cloud9



AWS Toolkit  
for PyCharm

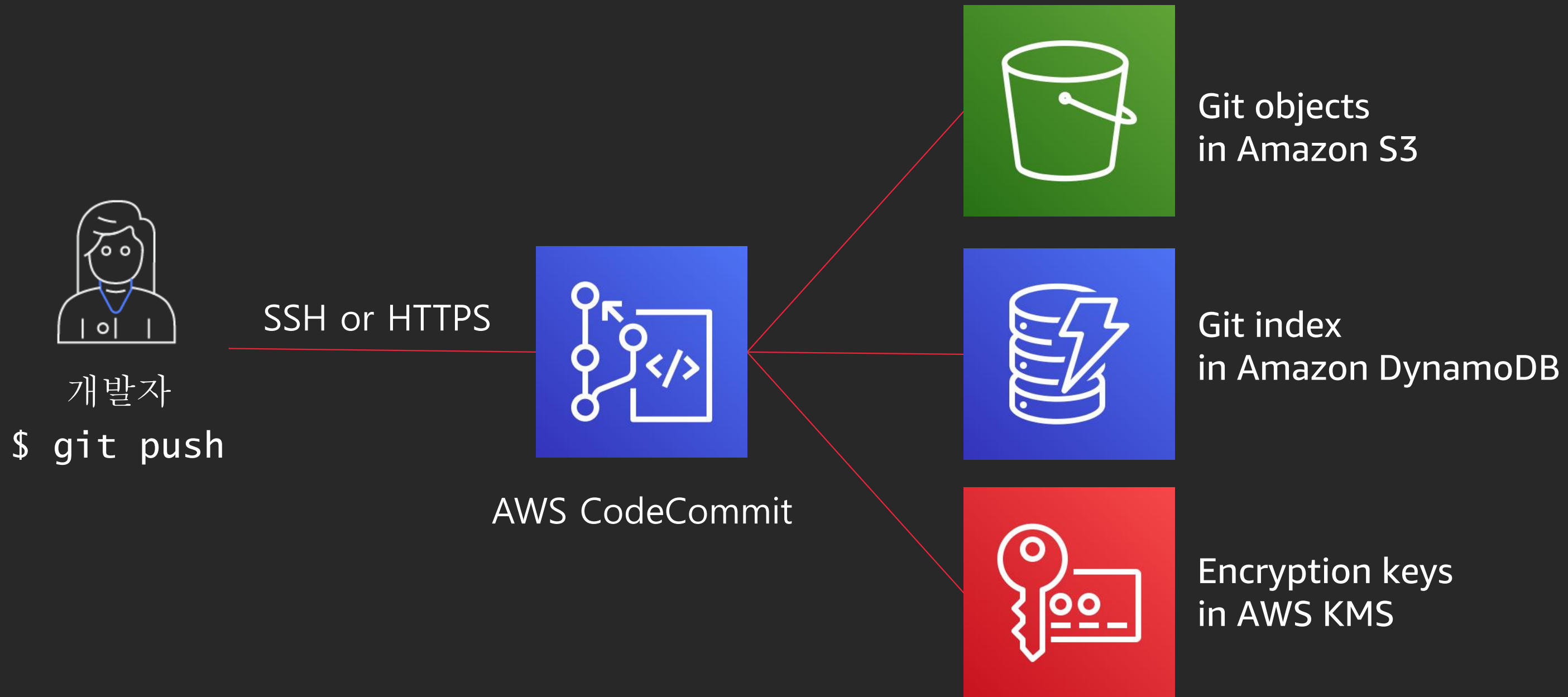


AWS Toolkit  
for IntelliJ



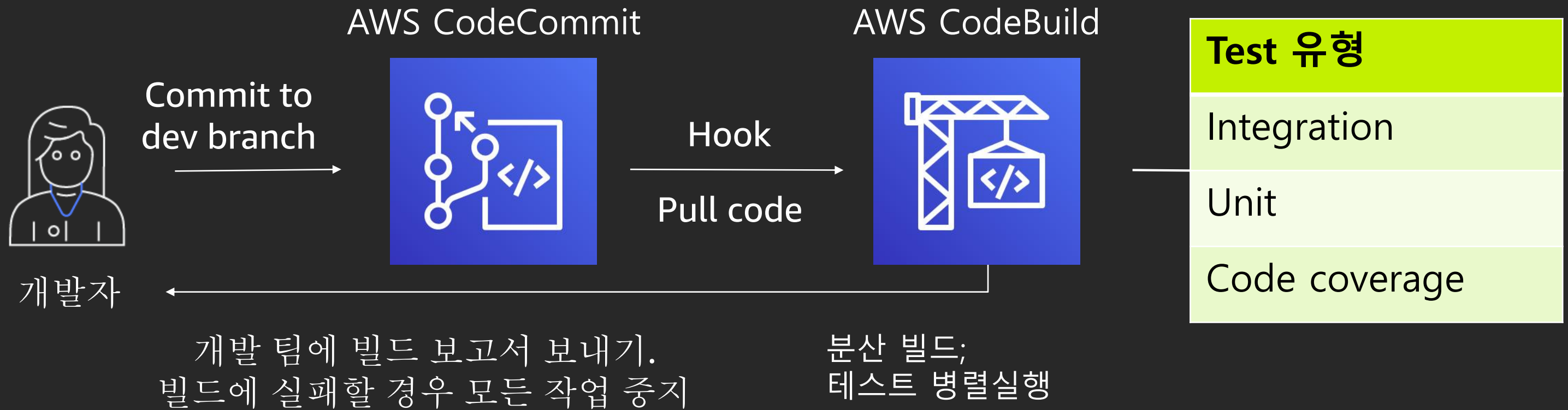
AWS Toolkit  
for Visual Studio  
Code

# AWS 서비스를 활용하여 DevOps 2단계



안전하고, 뛰어난 확장성의 프라이빗 Git 리포지토리

# AWS 서비스를 활용하여 DevOps 3단계



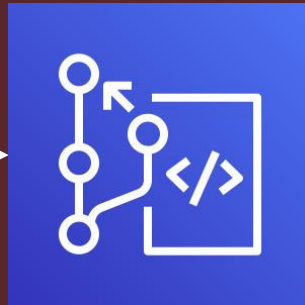
# AWS 서비스를 활용하여 DevOps 4단계

AWS CodePipeline

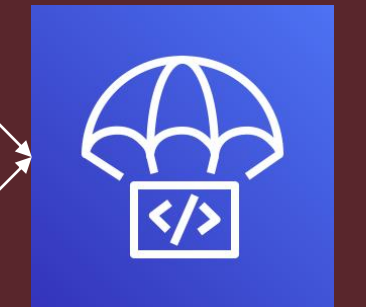
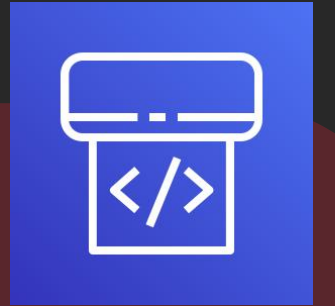
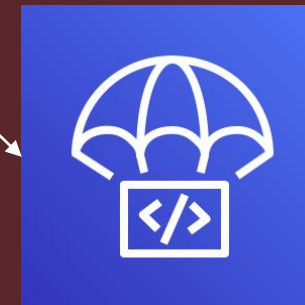
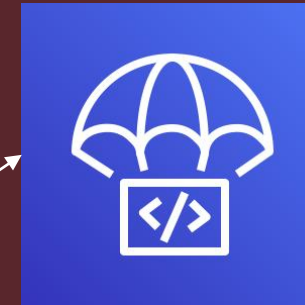
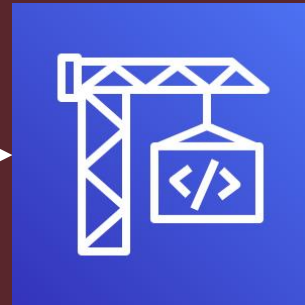


개발자

Merge PR  
into trunk



Hook  
pull code



AWS CodeDeploy

AWS CodeCommit

AWS CodeBuild

Testing  
environments

Production  
environments



# AWS DevOps 도구들 – CI/CD



AWS CodePipeline

소스

빌드

테스트

배포

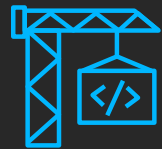
모니터링



AWS CodeCommit



AWS CodeBuild



AWS CodeBuild +  
Third Party



AWS CodeDeploy

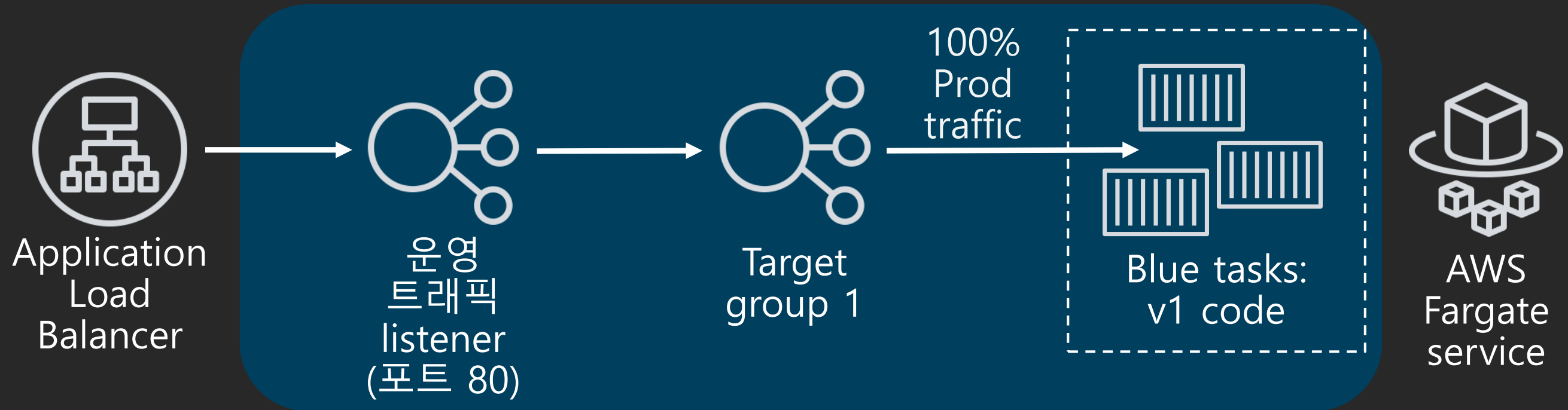


AWS X-Ray

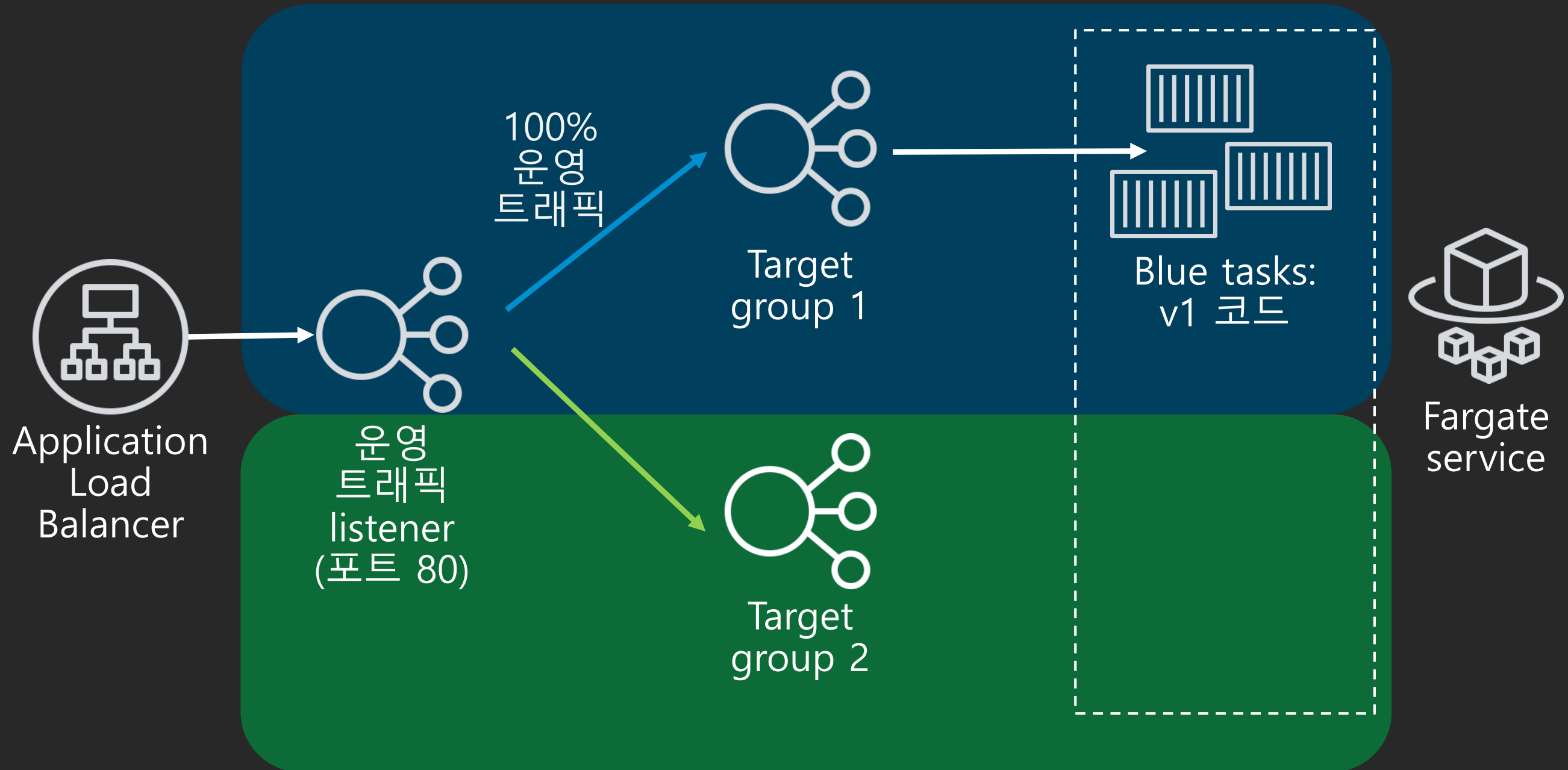


Amazon CloudWatch

# AWS Blue/Green 배포 예제

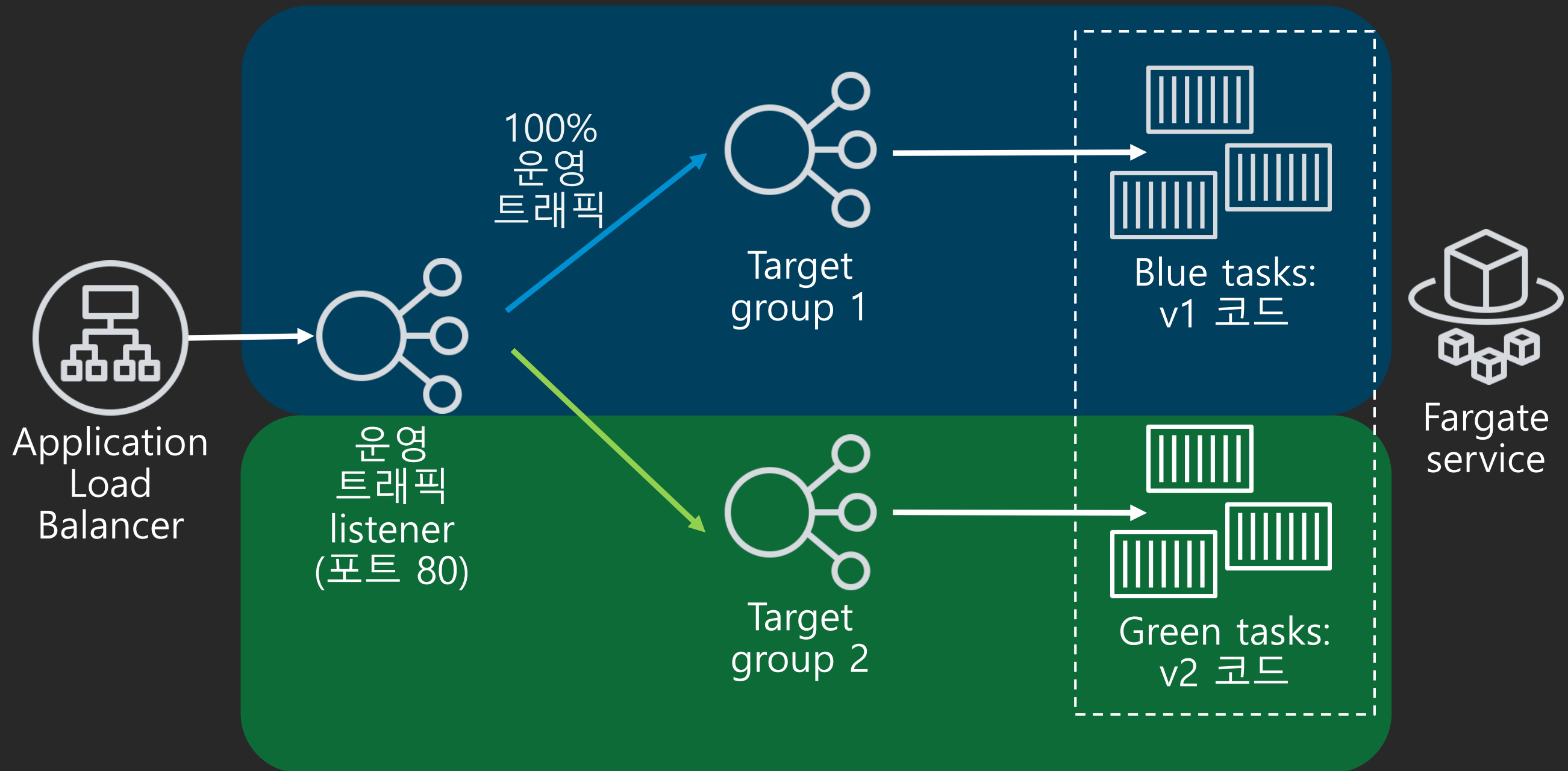


# AWS Blue/Green 배포 예제



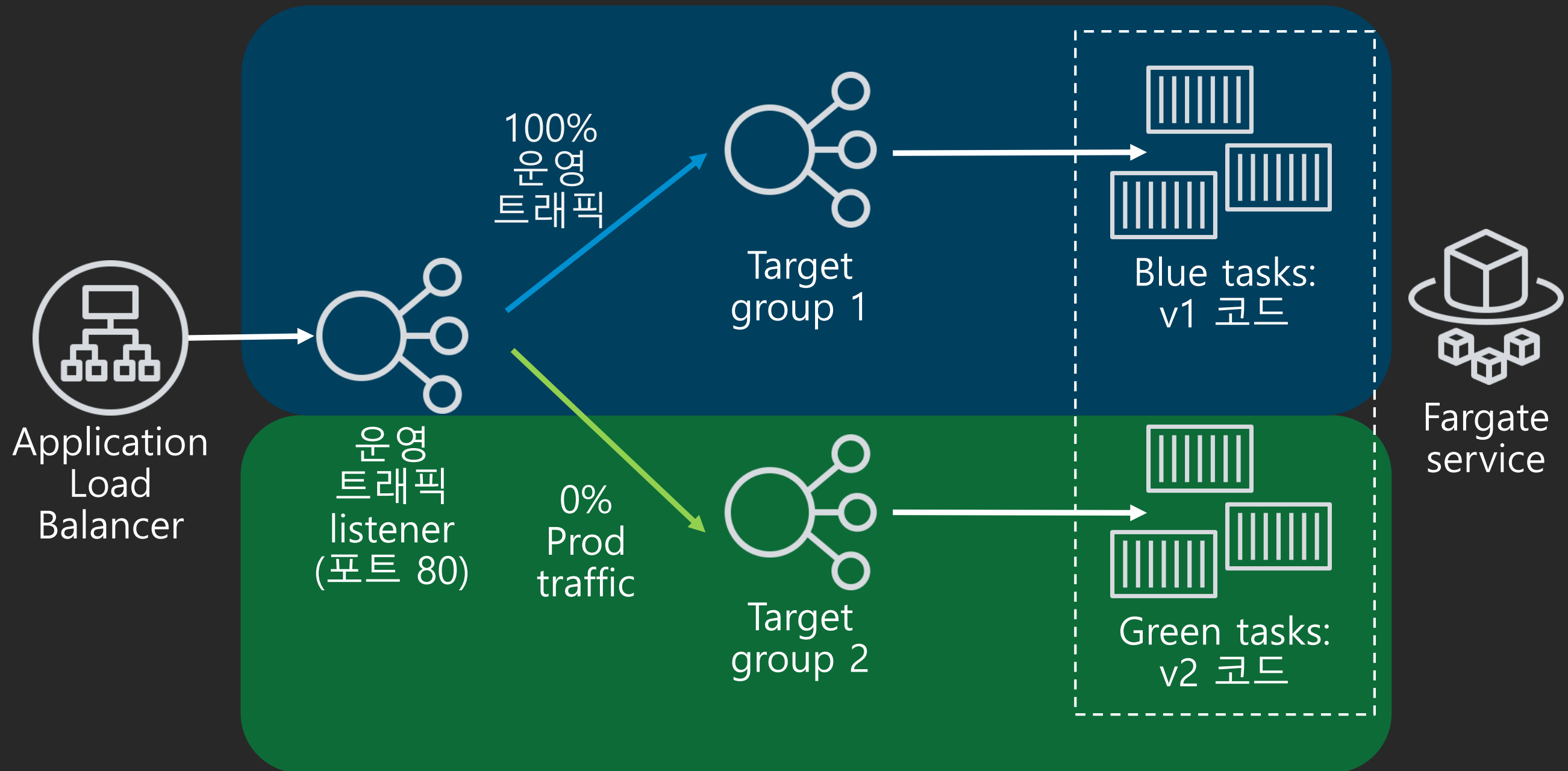
# AWS Blue/Green 배포 예제

## Green tasks 배포



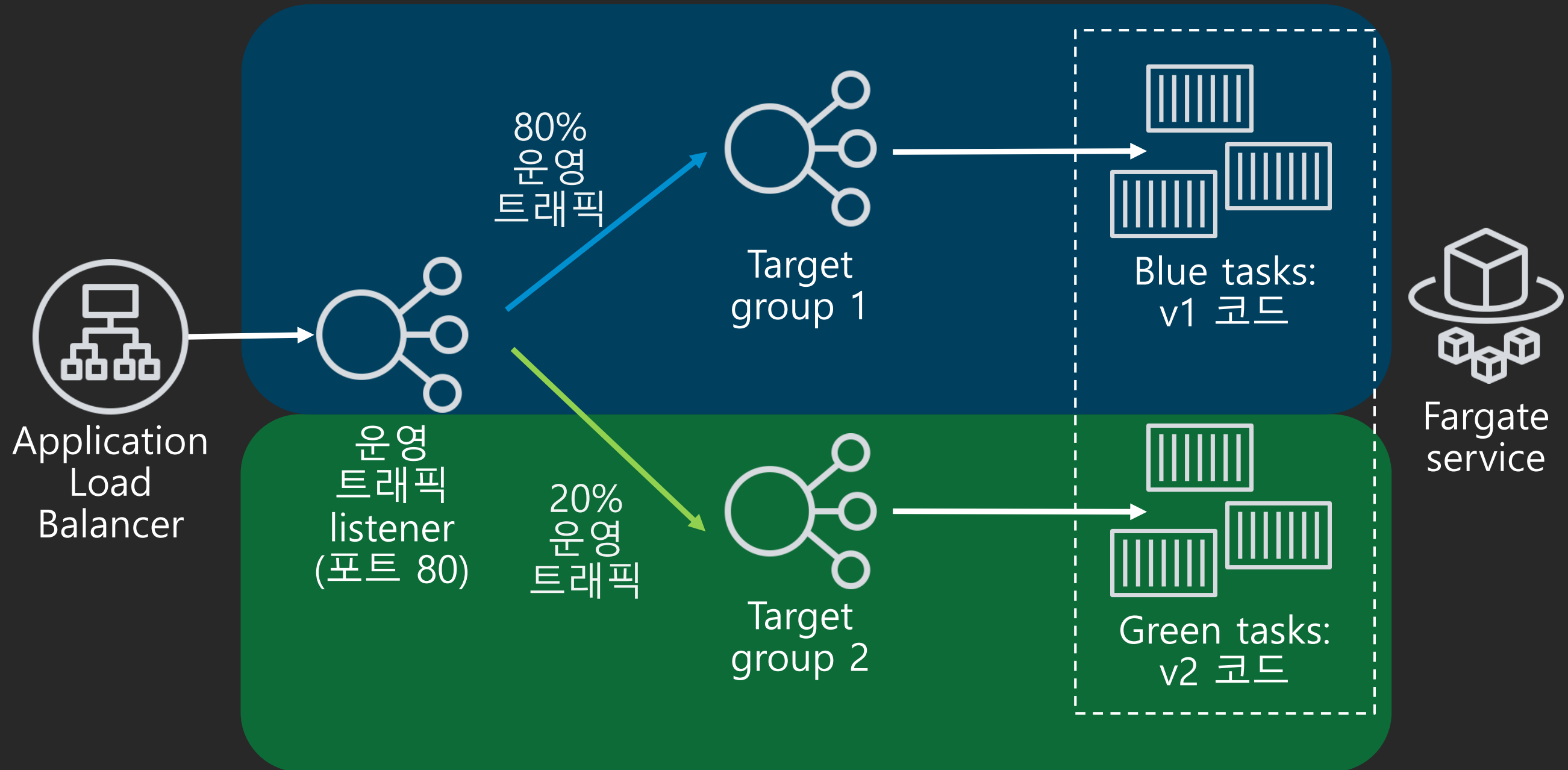
# AWS Blue/Green 배포 예제

새로운 타겟 그룹에 이벤트 후 **Green task**로 트래픽 부분 이동



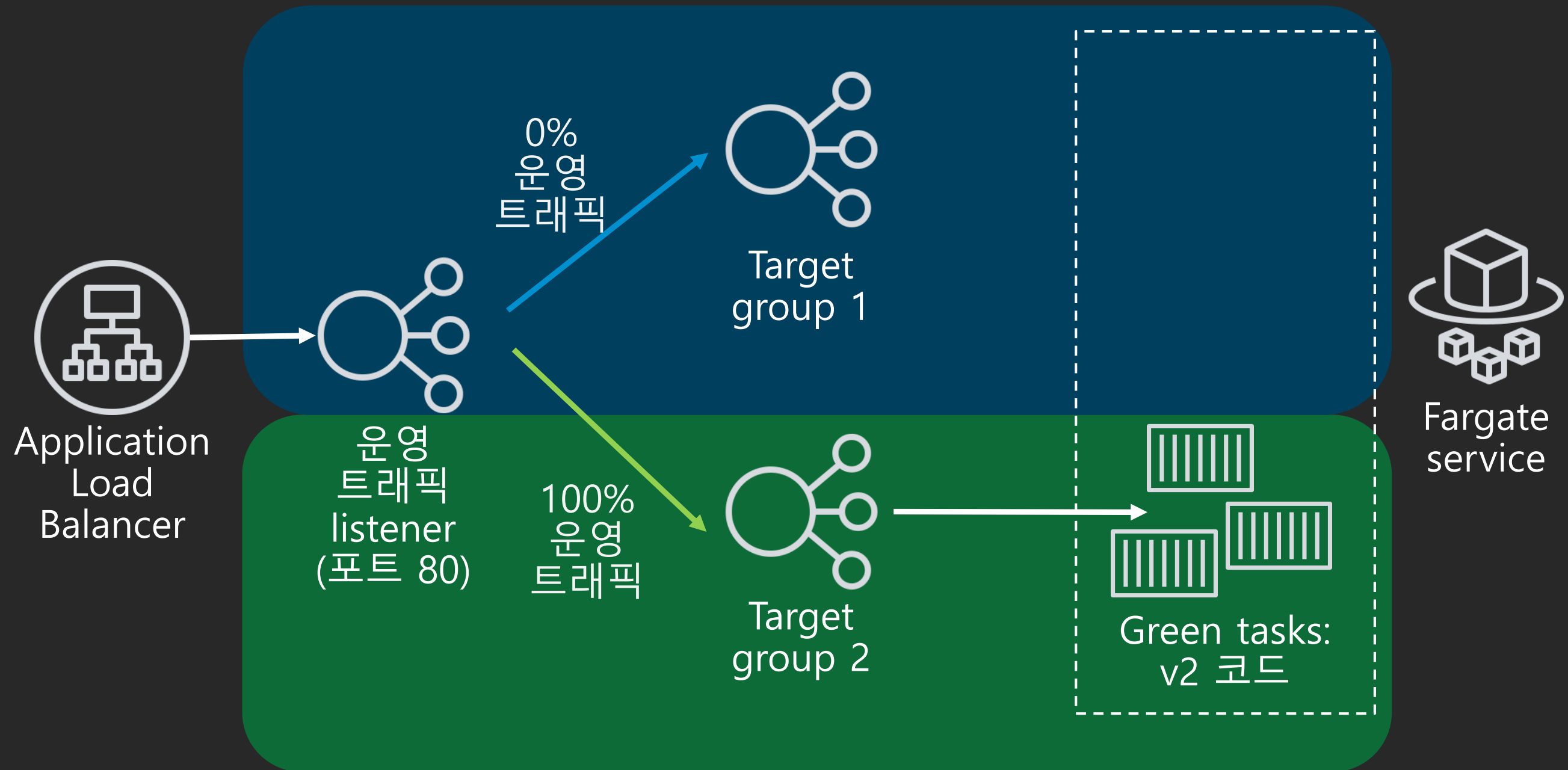
# AWS Blue/Green 배포 예제

Green tasks로 트래픽 부분 이관; 알람 발생시 Roll-back



# AWS Blue/Green 배포 예제

## Blue tasks 드레인시키기



정리해보자면...



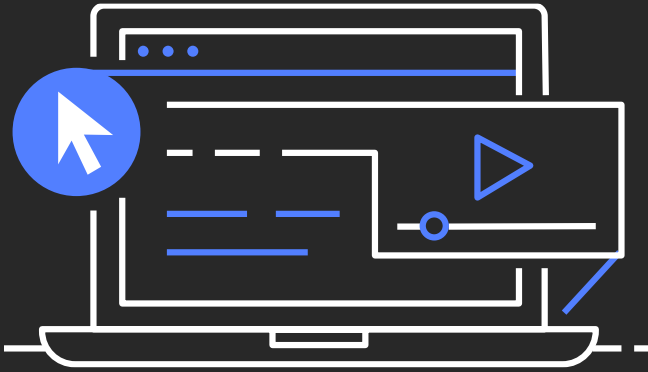
문화

방법론

도구



# AWS 온라인 교육 과정



자신의 속도에 맞춰 학습하세요.

무료 AWS 디지털 교육을 통해  
편한 시간에 원하는 장소에서  
최신 클라우드 기술을  
학습할 수 있습니다.

- AWS Cloud Practitioner Essentials  
AWS 클라우드의 기초를 배우고, AWS Certified Cloud Practitioner 공인 자격 시험을 준비할 수 있는 과정입니다.  
<https://www.aws.training/Details/Curriculum?id=32442>
- AWS 클라우드 보안 기초  
AWS 액세스 제어 및 관리, 거버넌스, 로깅 및 암호화 방법 등 AWS의 보안 개념을 소개합니다.  
<https://www.aws.training/Details/Curriculum?id=11048>
- Amazon Elastic Block Storage (EBS) 소개  
AWS 클라우드의 Amazon EC2 인스턴스에 사용할 블록 스토리지 볼륨을 제공하는 Amazon Elastic Block Store(EBS)를 소개합니다.  
<https://www.aws.training/Details/Video?id=37393>

# AWS Builders Online Series에 참석해주셔서 대단히 감사합니다.

저희가 준비한 내용, 어떻게 보셨나요?  
더 나은 세미나를 위하여 **설문을 꼭 작성해 주시기 바랍니다.**



[aws-korea-marketing@amazon.com](mailto:aws-korea-marketing@amazon.com)



[twitter.com/AWSKorea](https://twitter.com/AWSKorea)



[facebook.com/amazonwebservices.ko](https://facebook.com/amazonwebservices.ko)



[youtube.com/user/AWSKorea](https://youtube.com/user/AWSKorea)



[slideshare.net/awskorea](https://slideshare.net/awskorea)



[twitch.tv/aws](https://twitch.tv/aws)



# Builders Online Series

# Thank you