



kubernetes

목차

1. 컨테이너

- 1-1. 컨테이너 환경의 탄생.
- 1-2. Containerization.
- 1-3. 컨테이너의 특징.

2. 많은 컨테이너를 관리하면서 생기는 문제점.

- 1-1. 서비스 배포
- 1-2. 서비스 검색
- 1-3. 서비스 관리

3. Container Orchestration

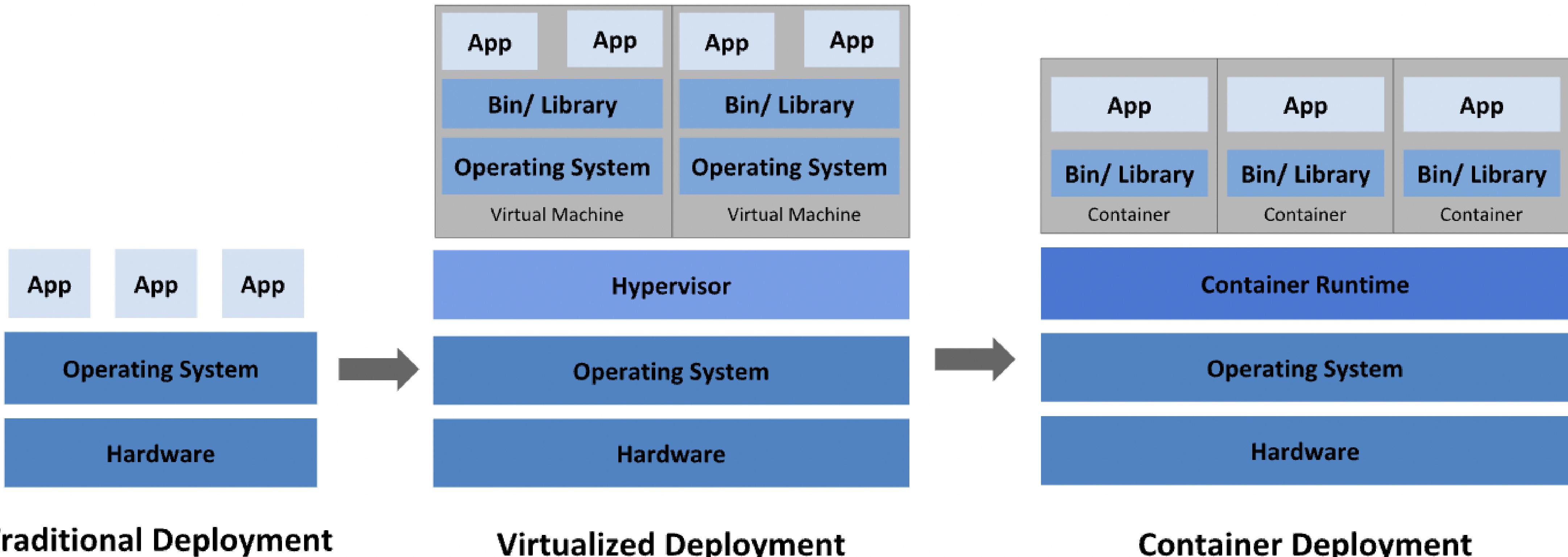
- 1-1. Container Orchestration의 특징

4. 왜 쿠버네티스?

- 1-1. 쿠버네티스란?



1-1. 컨테이너 환경의 탄생

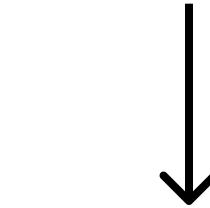




1-2. Containerization



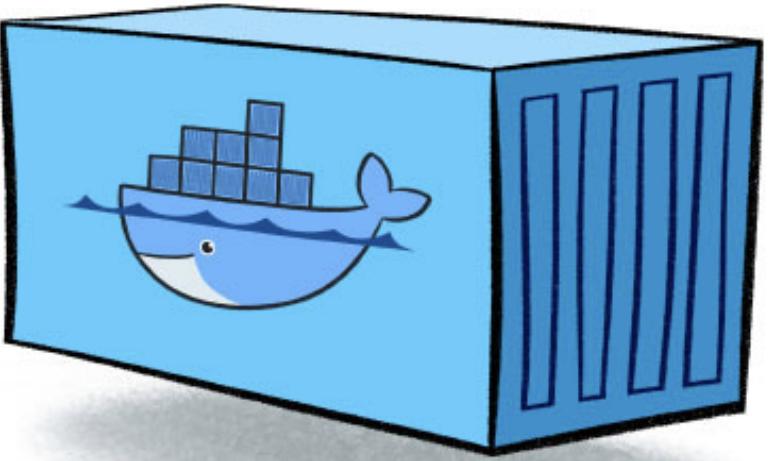
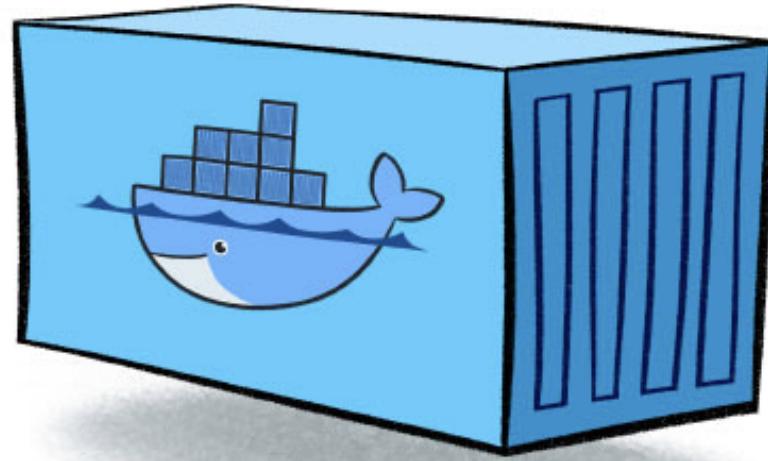
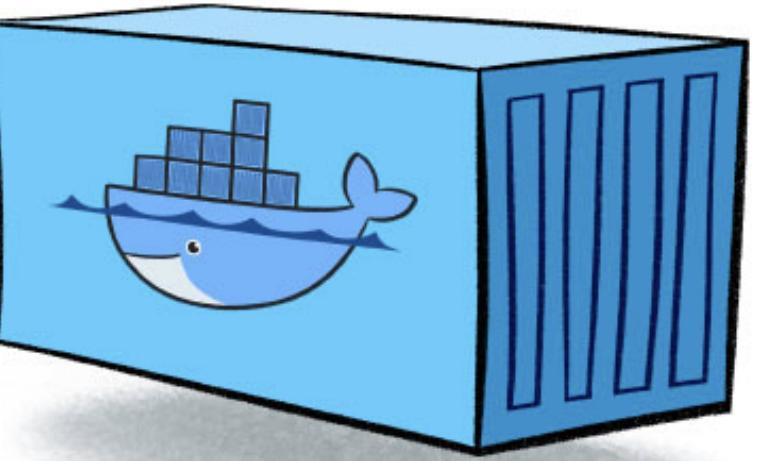
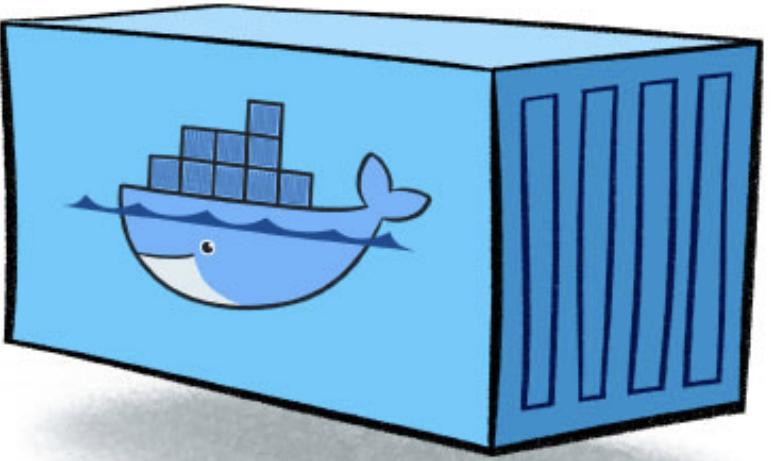
Flask



redis

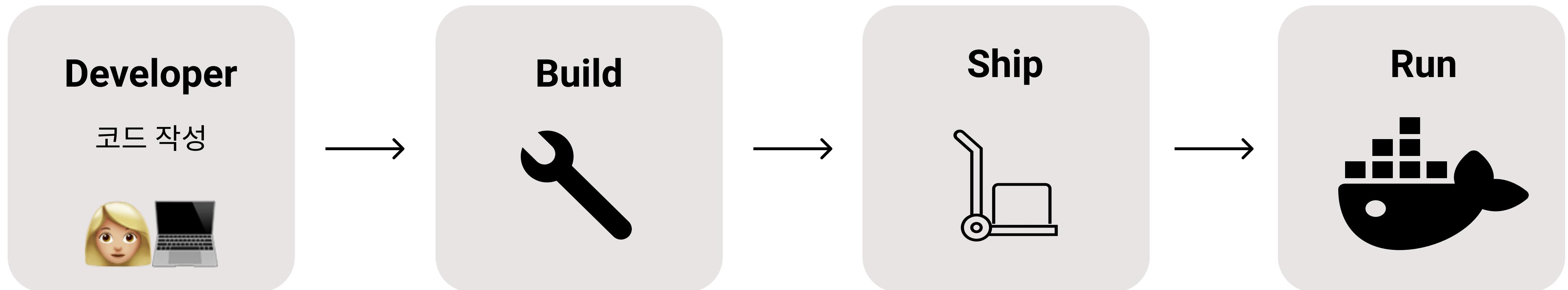


MySQL®





1-2. Containerization

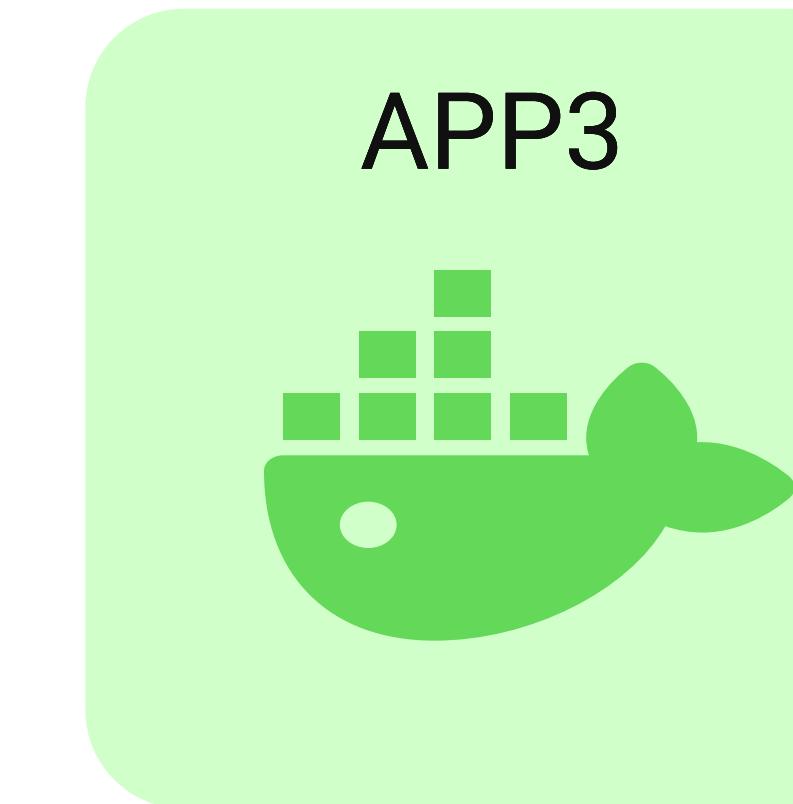
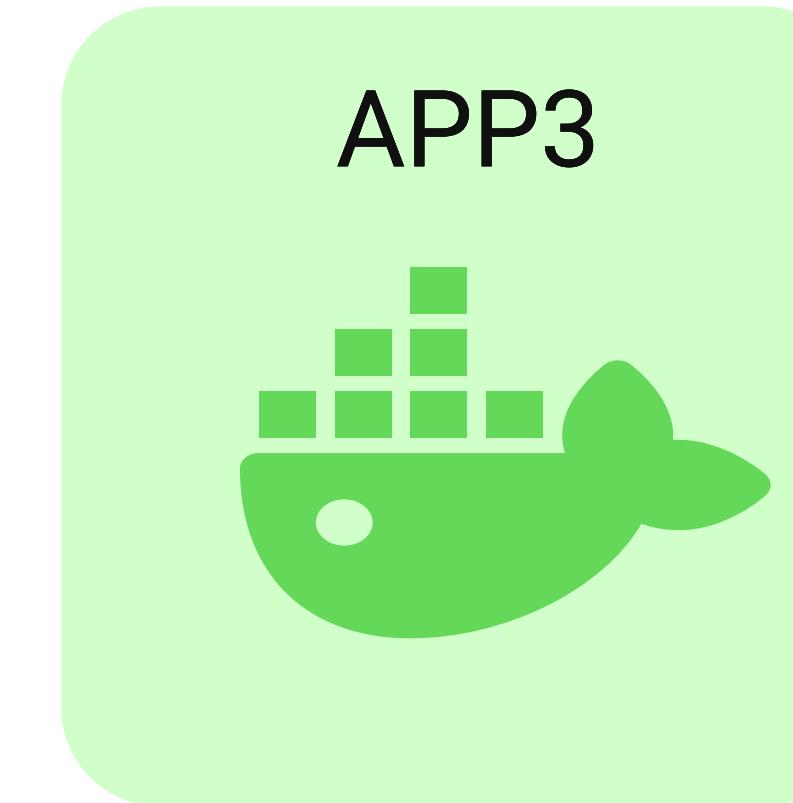




1-3. 컨테이너의 특징

- 가상머신과 비교하여 **생산이 쉽고 효율적**
- 컨테이너 이미지를 이용한 **배포와 롤백이 간단**
- 언어 & 프레임워크에 관계없이 동일한 방식으로 관리
- 개발, 테스팅, 운영 환경은 물론 로컬 PC와 Cloud까지 **동일한 환경**
- 특정 클라우드에 종속적이지 않음.

(ex. AWS -> Google Cloud로 쉽게 서버 이동)



컨테이너수의 증가 이유

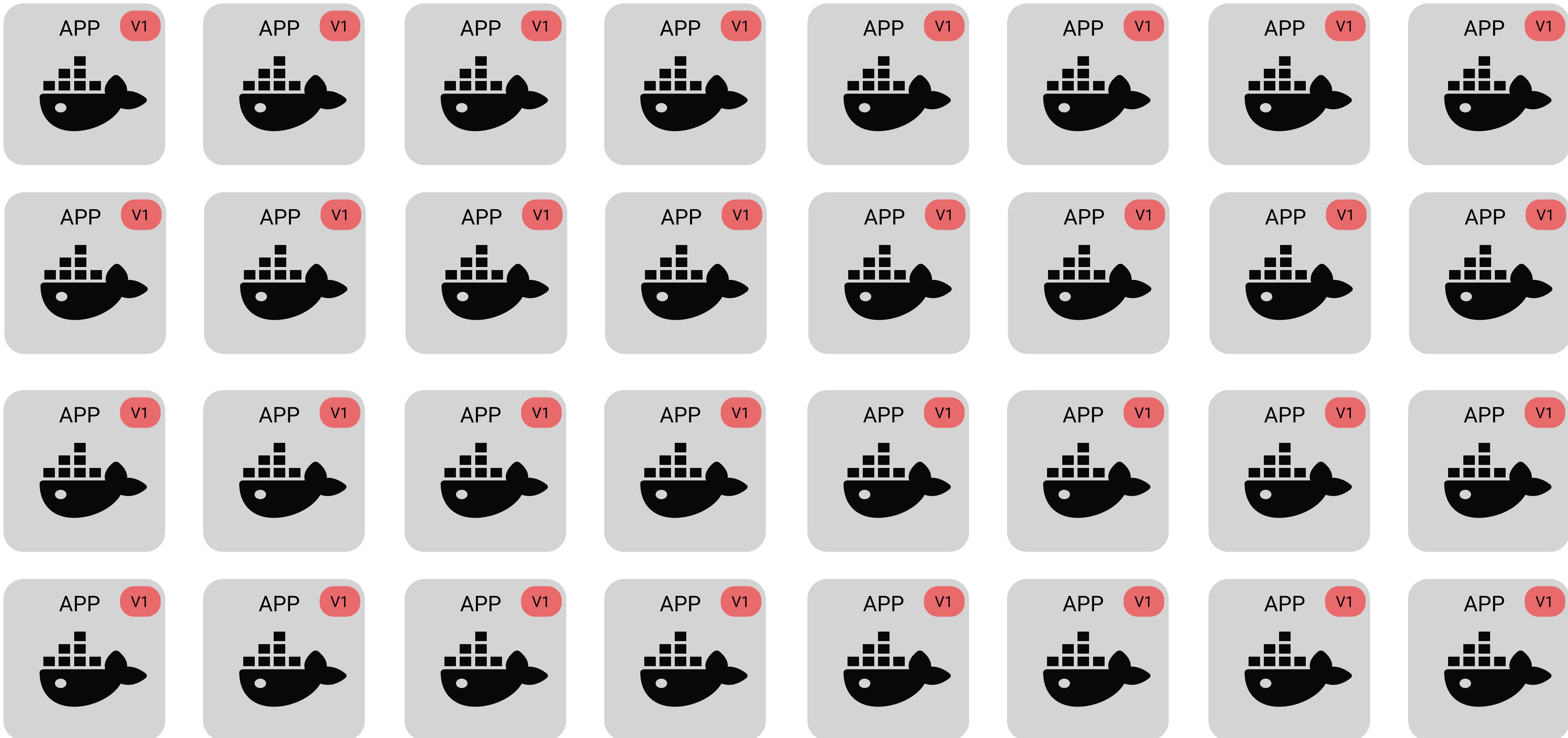
- 요청의 증가
- 서비스 크기의 증가
- 마이크로 서비스

컨테이너 배포

버전 관리, Rollout & Rollback...

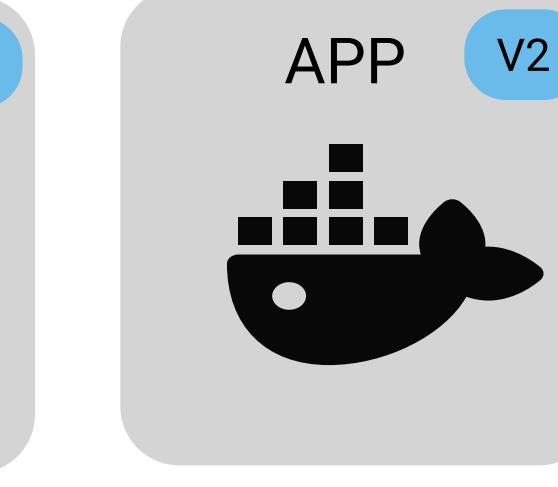
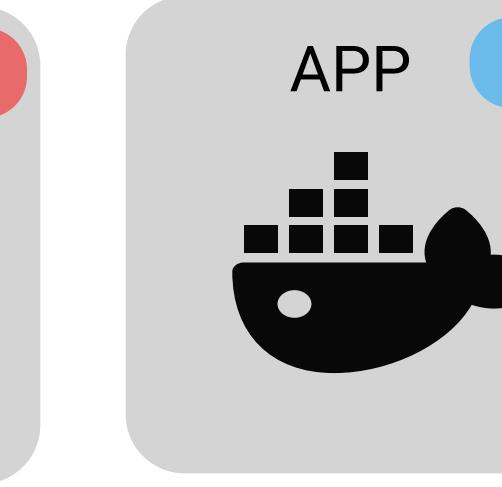
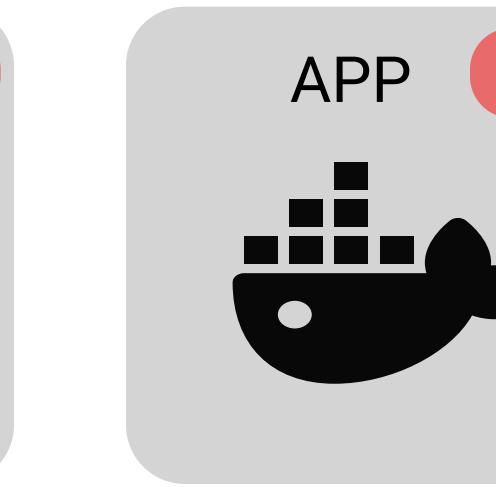
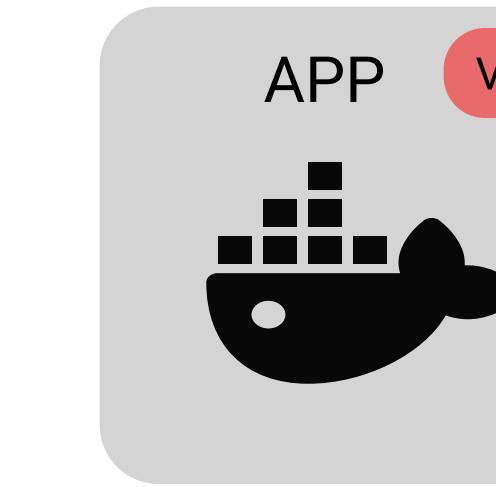
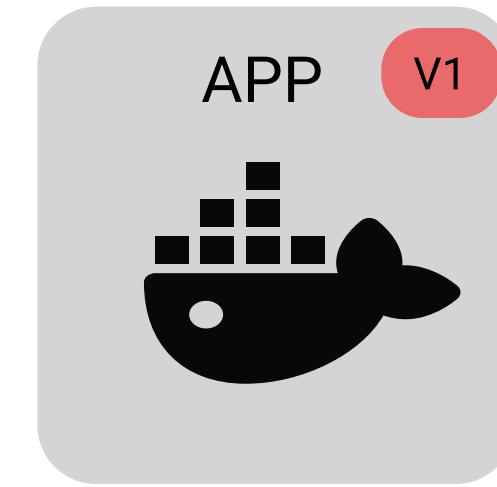
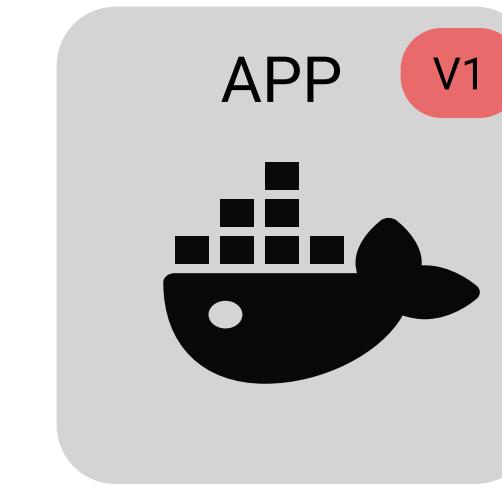
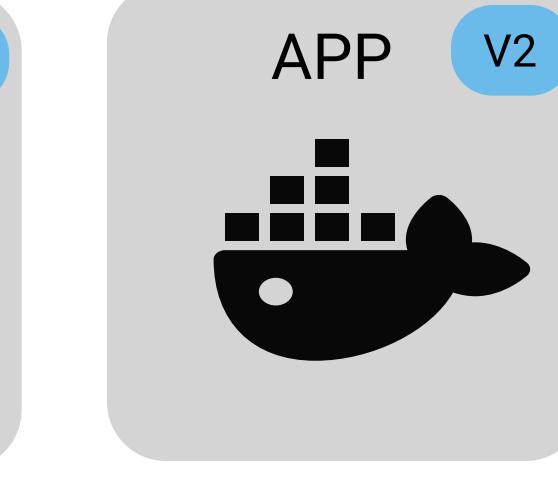
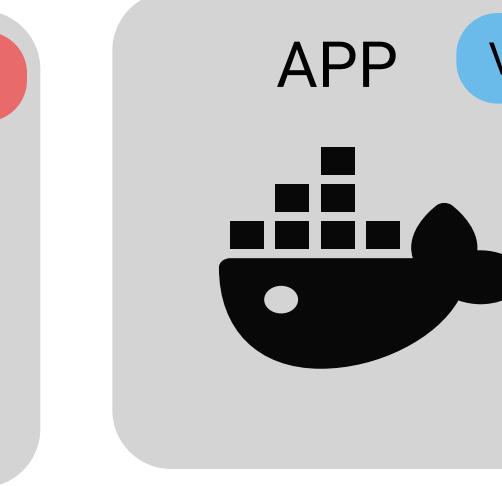
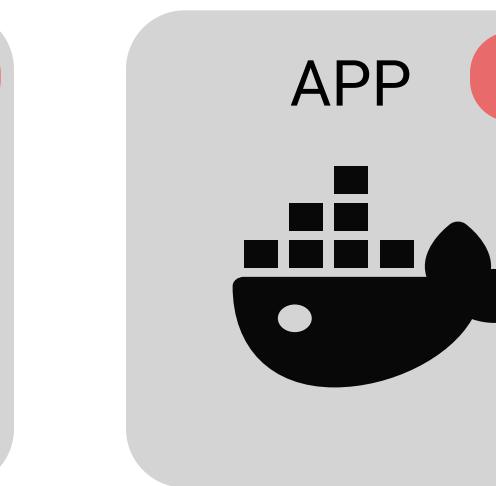
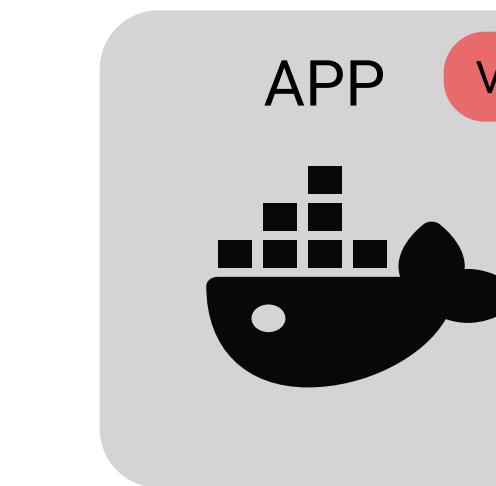
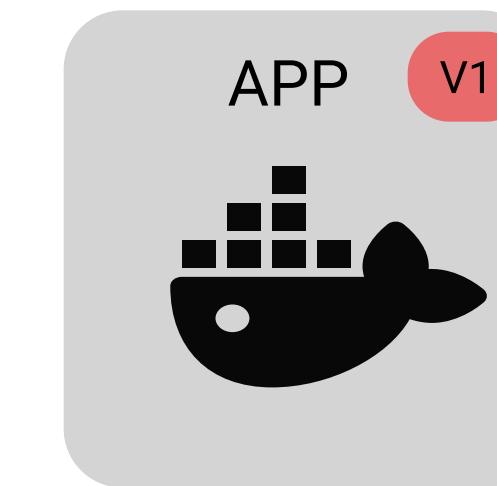
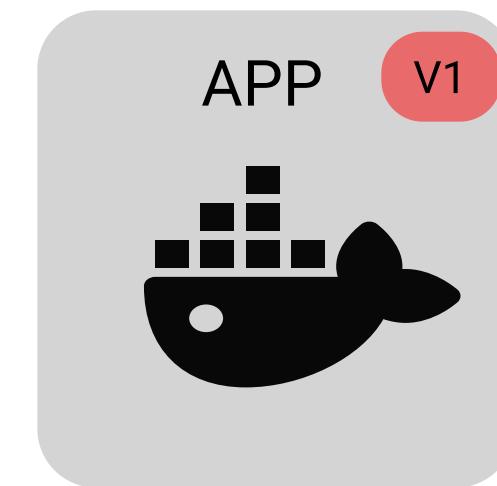
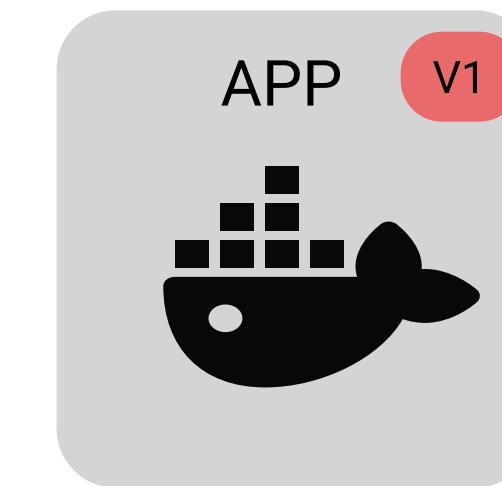
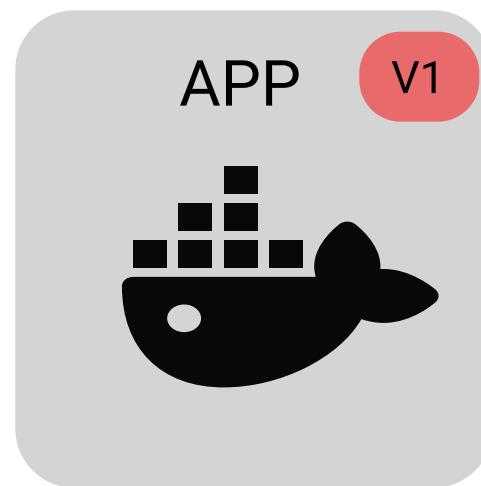
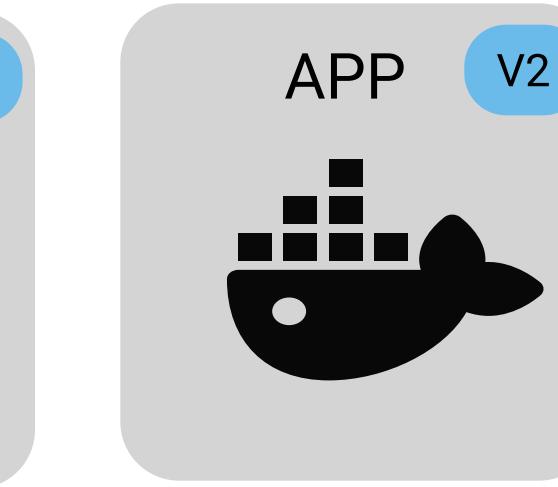
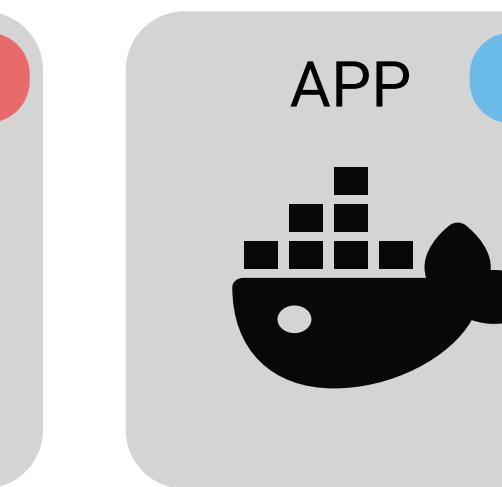
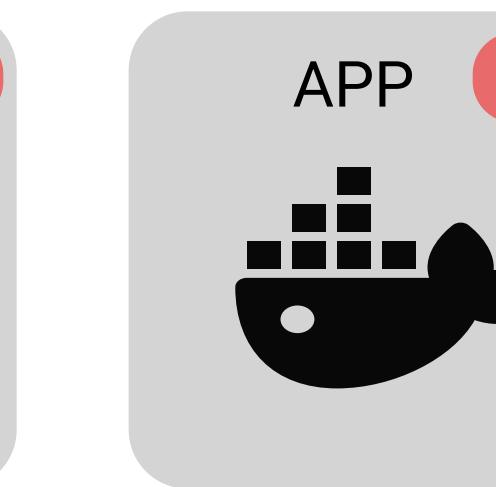
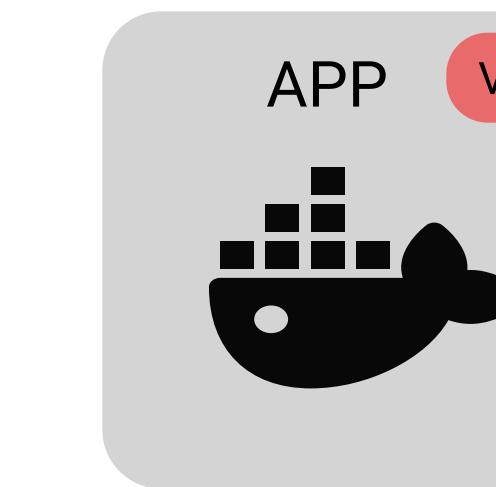
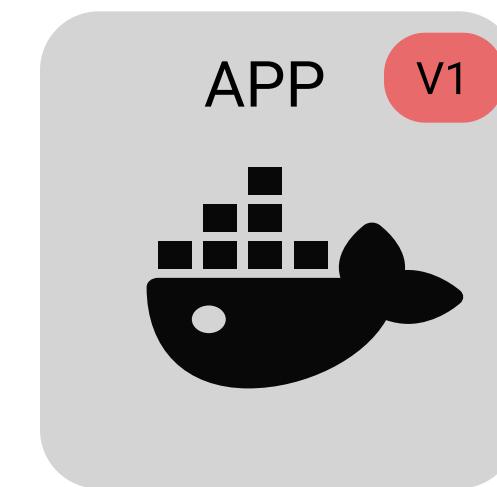
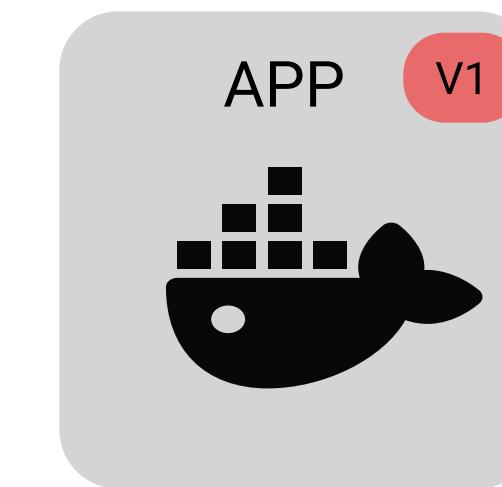
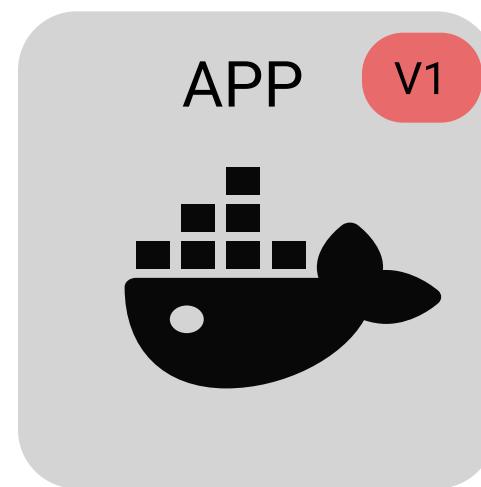
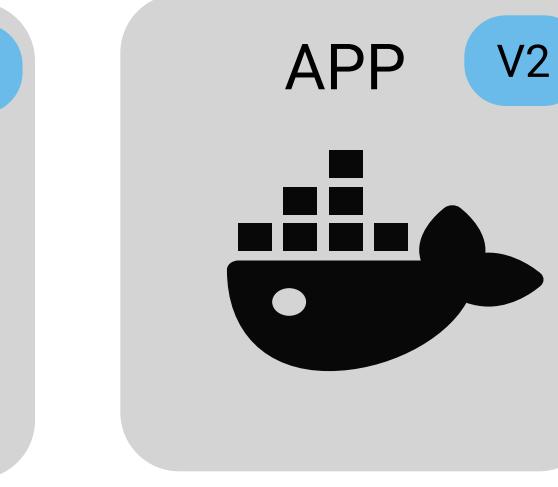
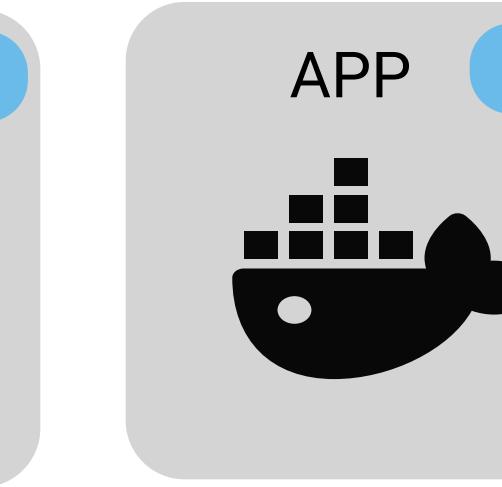
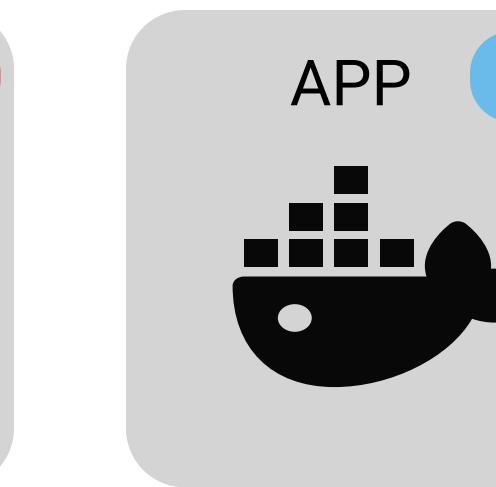
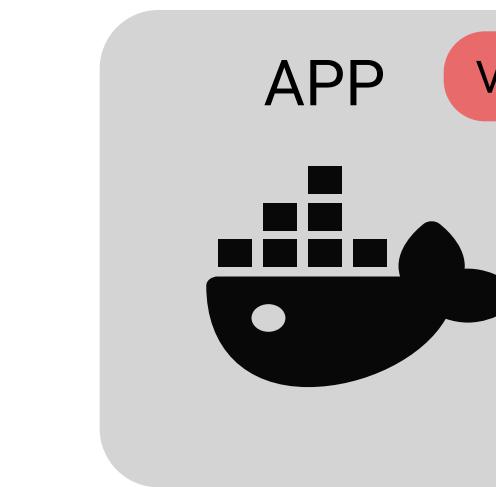
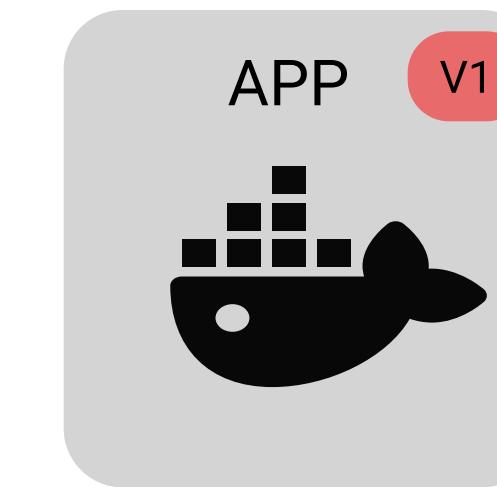
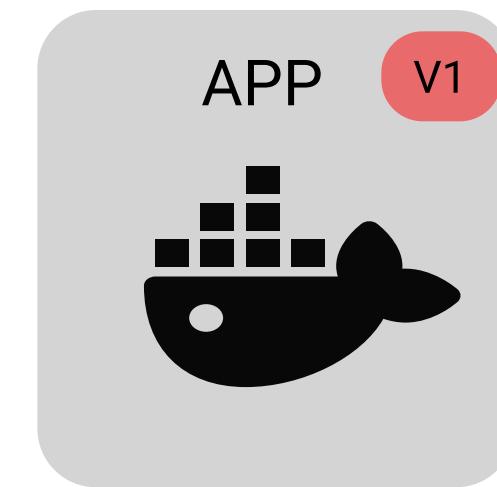
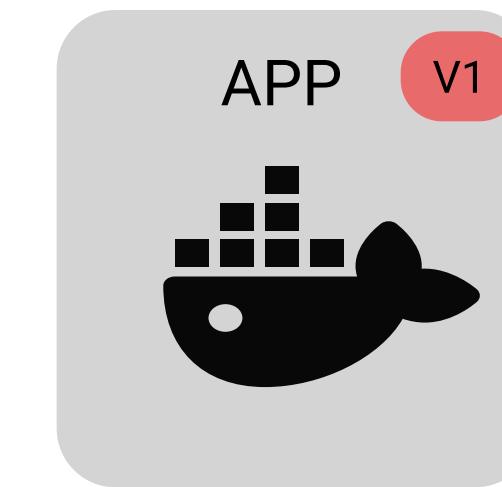
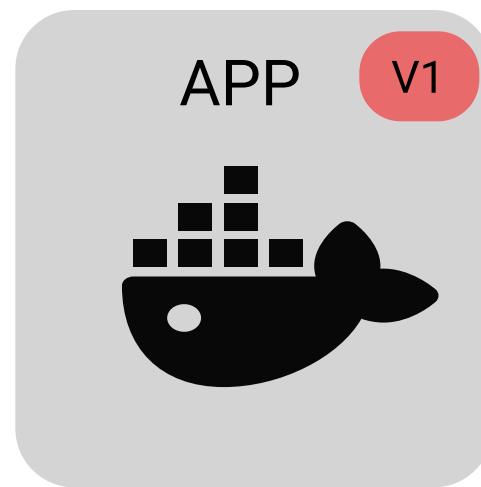


2-1. 컨테이너 배포



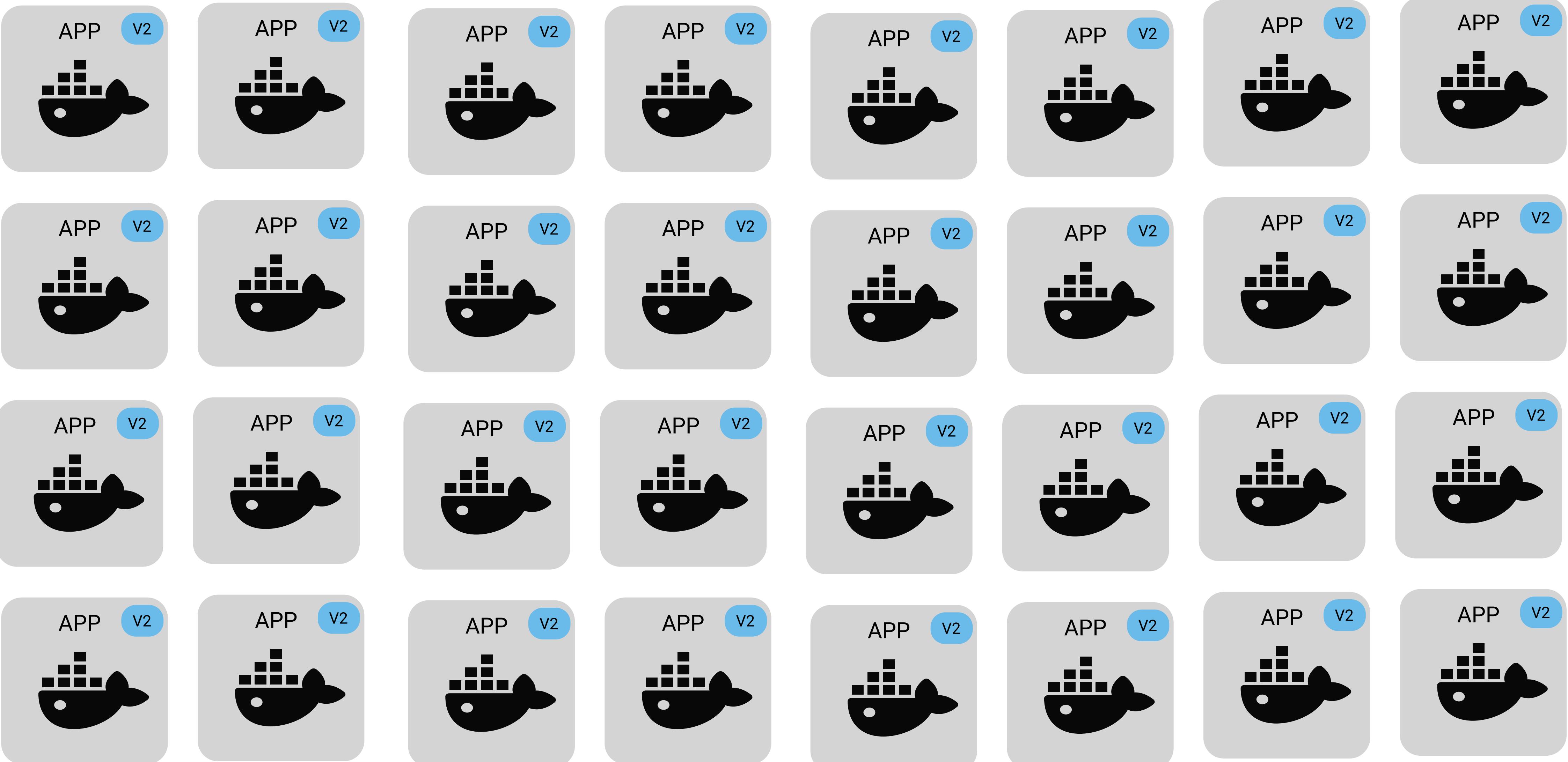


2-1. 컨테이너 배포





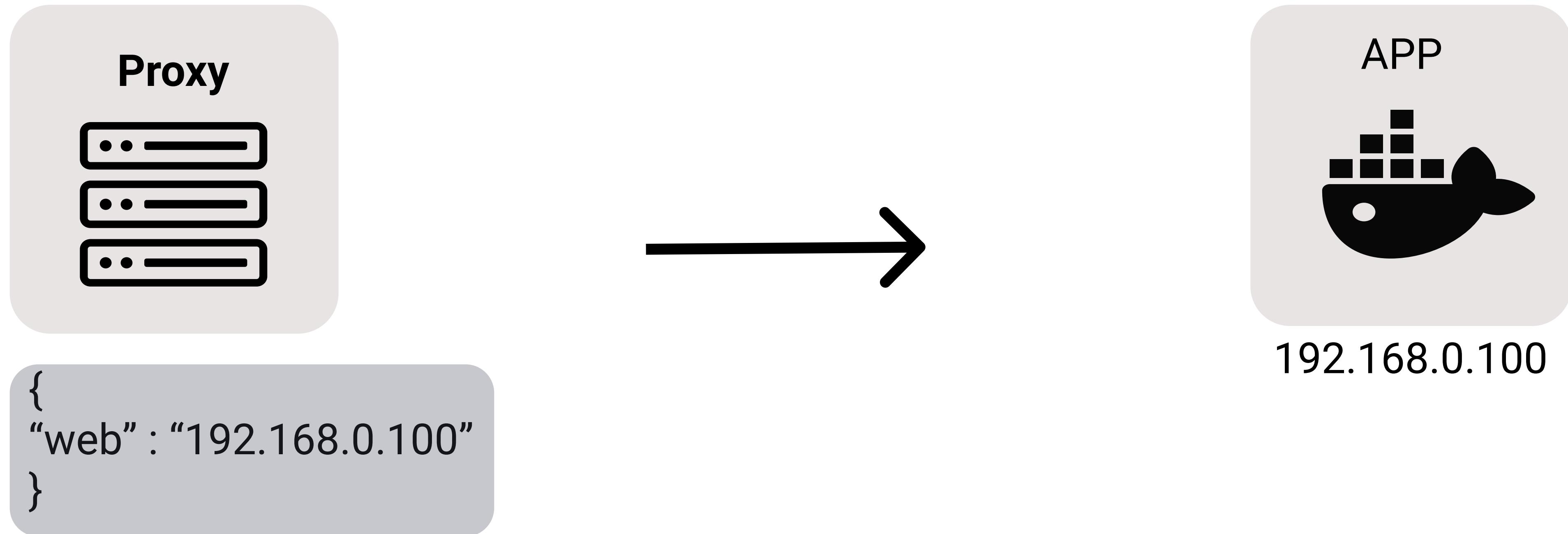
2-1. 컨테이너 배포



서비스 검색

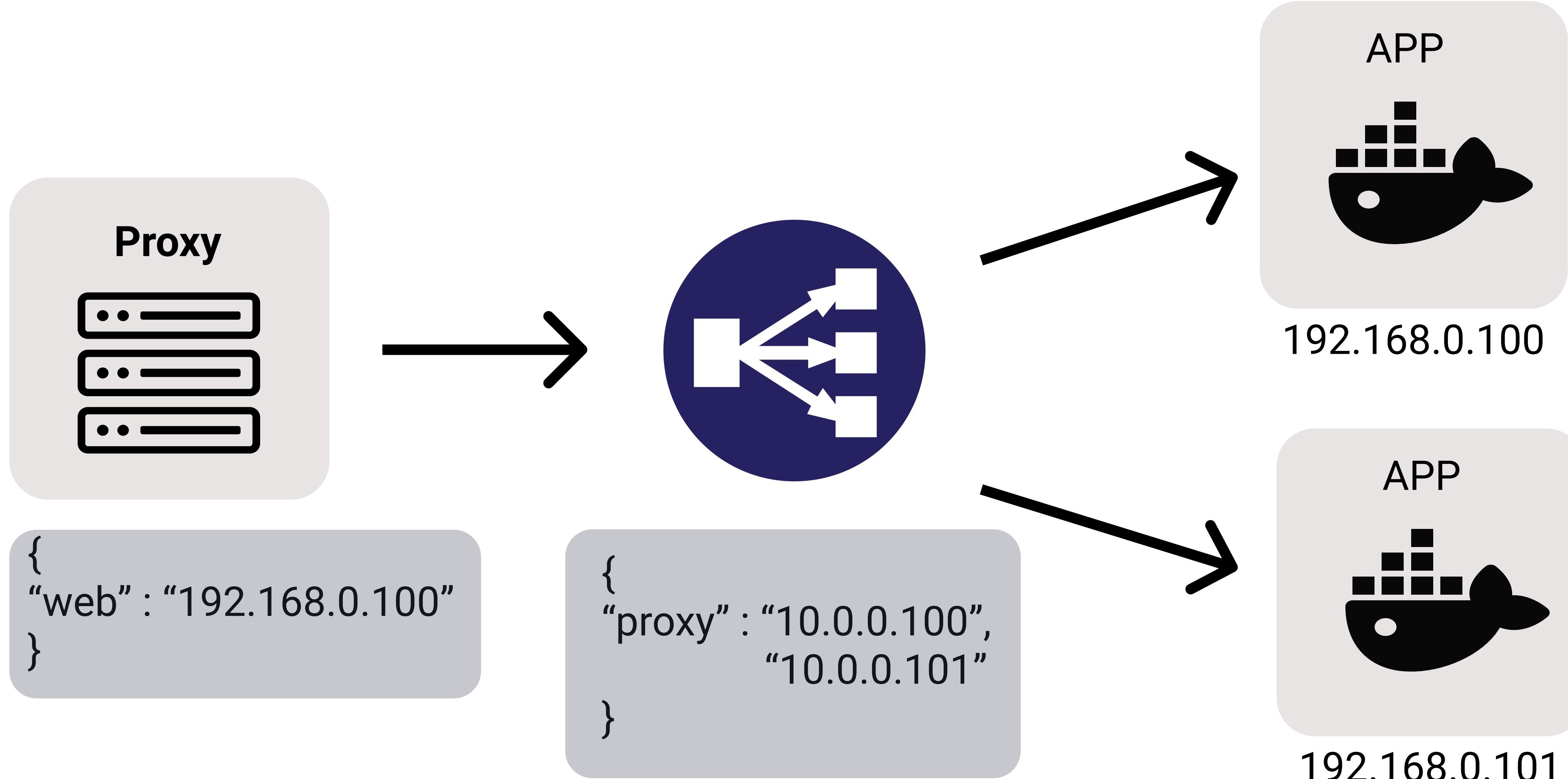


2-2. 서비스 검색



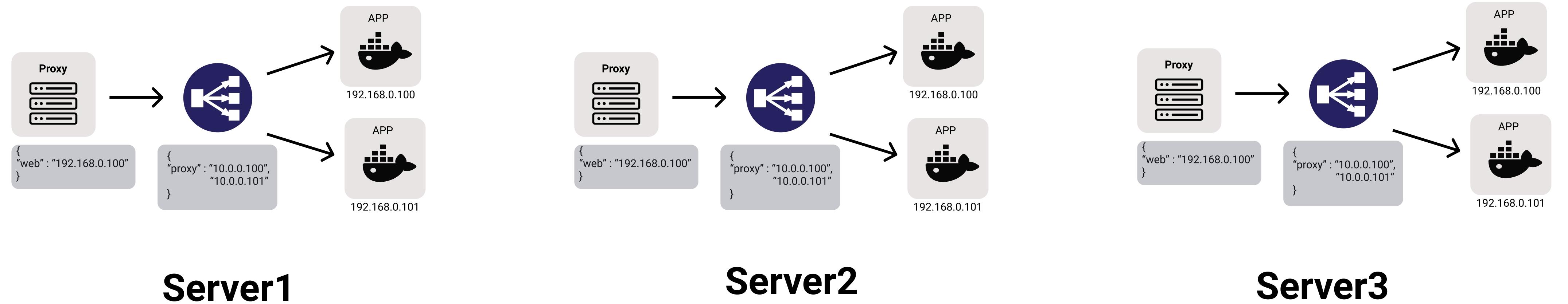


2-2. 서비스 검색





2-2. 서비스 검색



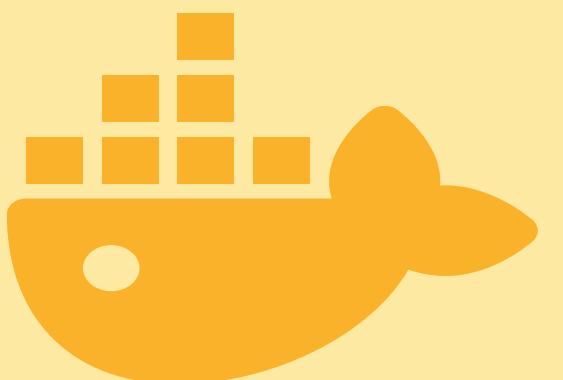
서비스 관리

부하 모니터링, 환경 체크...

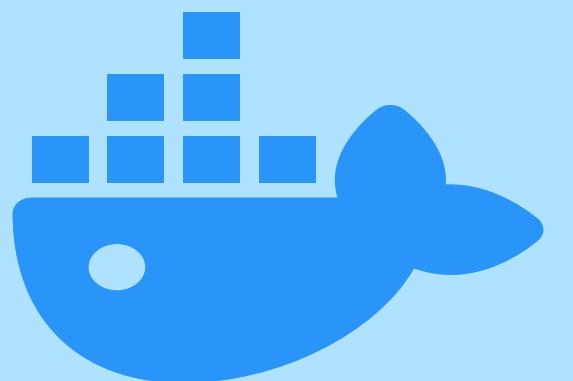


2-3. 서비스 관리

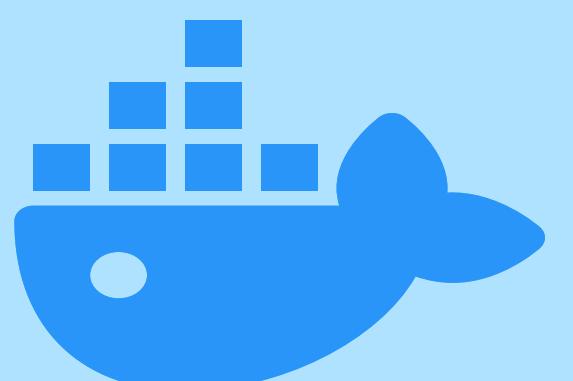
APP1



APP2



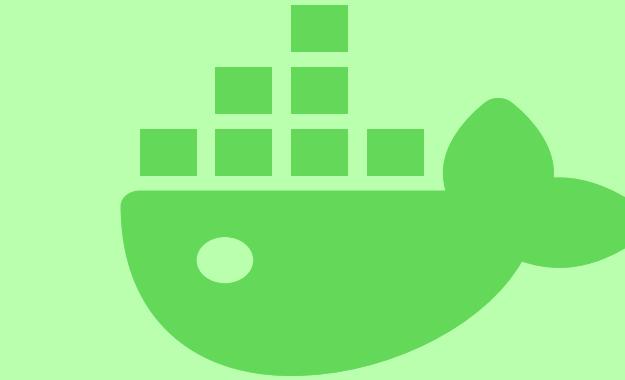
APP2



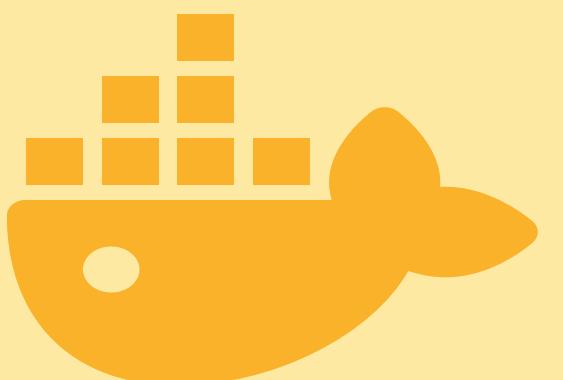
APP3



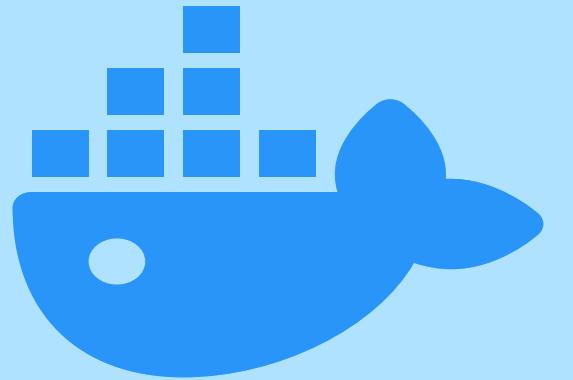
APP3



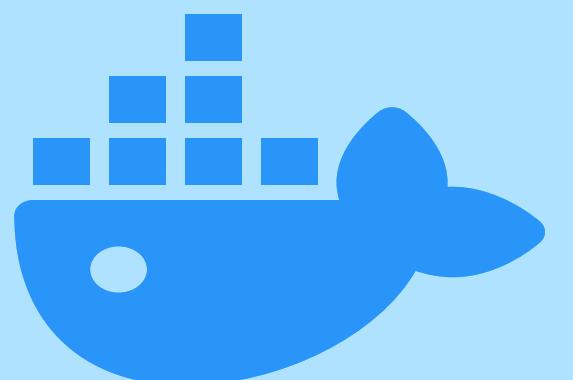
APP1



APP2



APP2



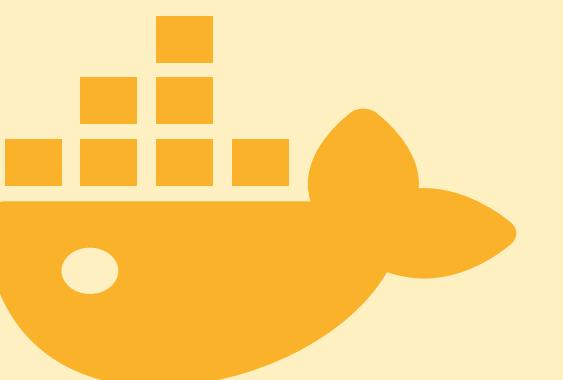
APP3



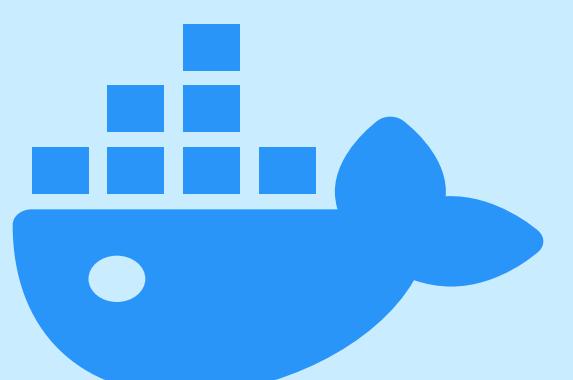
APP3



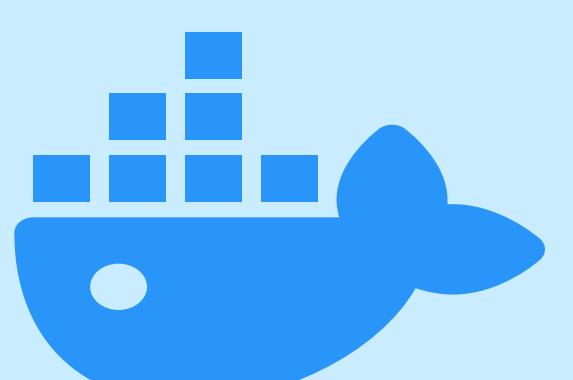
APP1



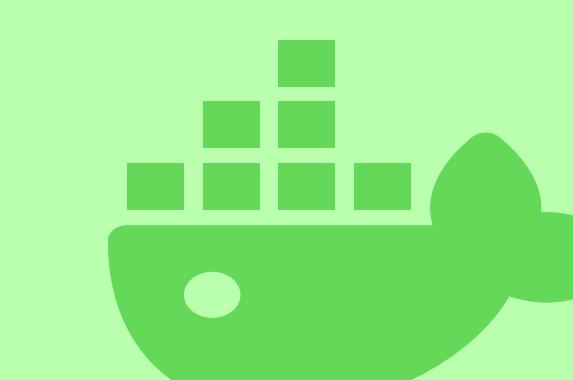
APP2



APP2



APP3

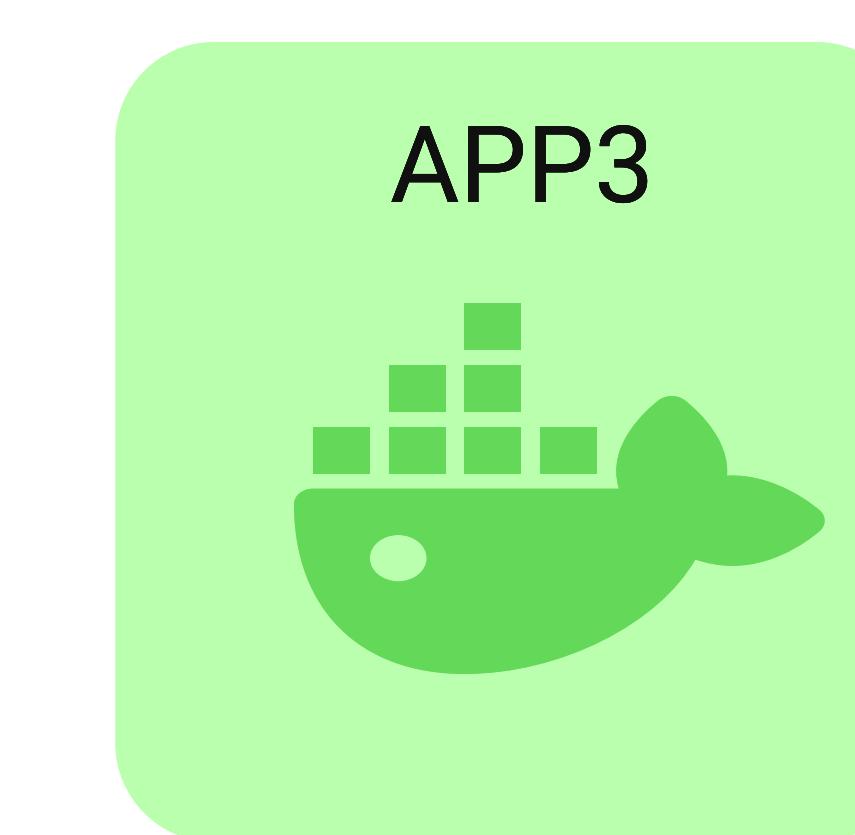
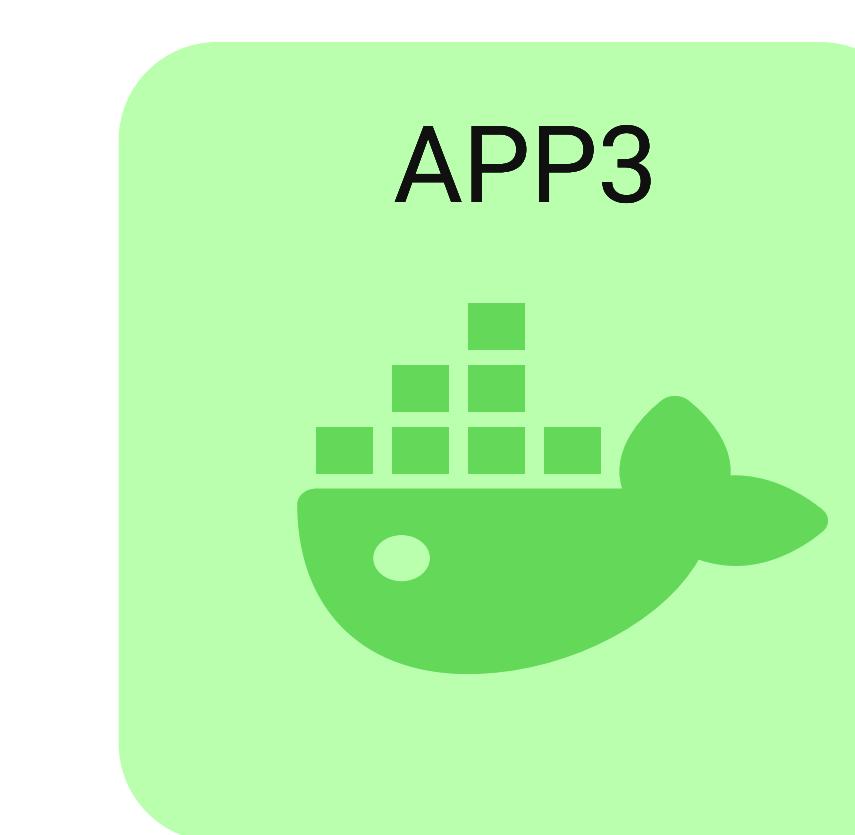
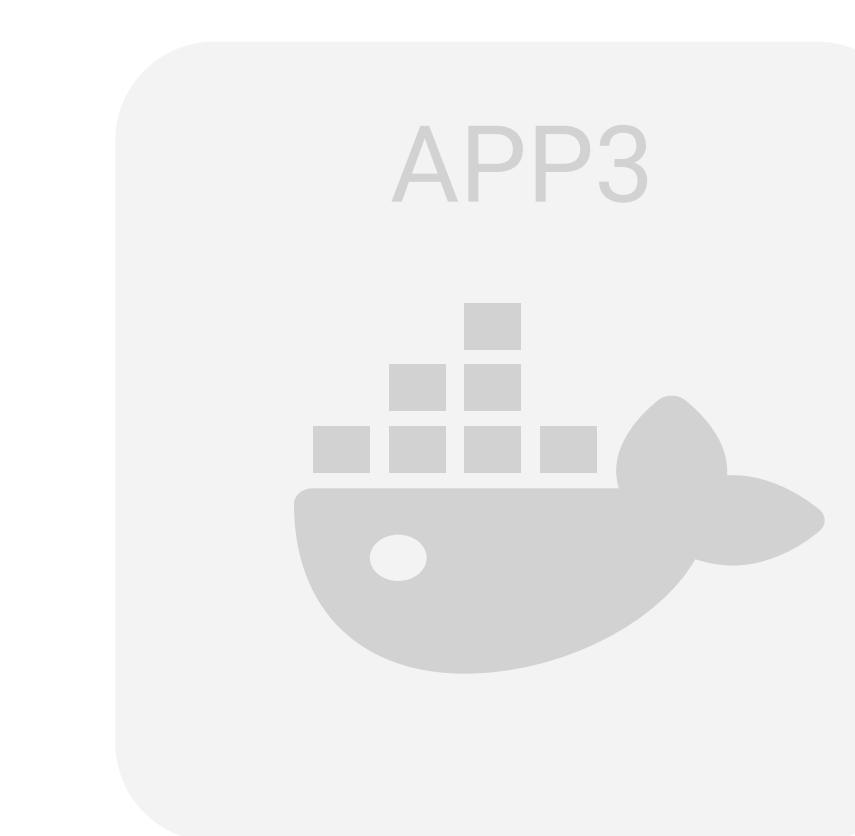
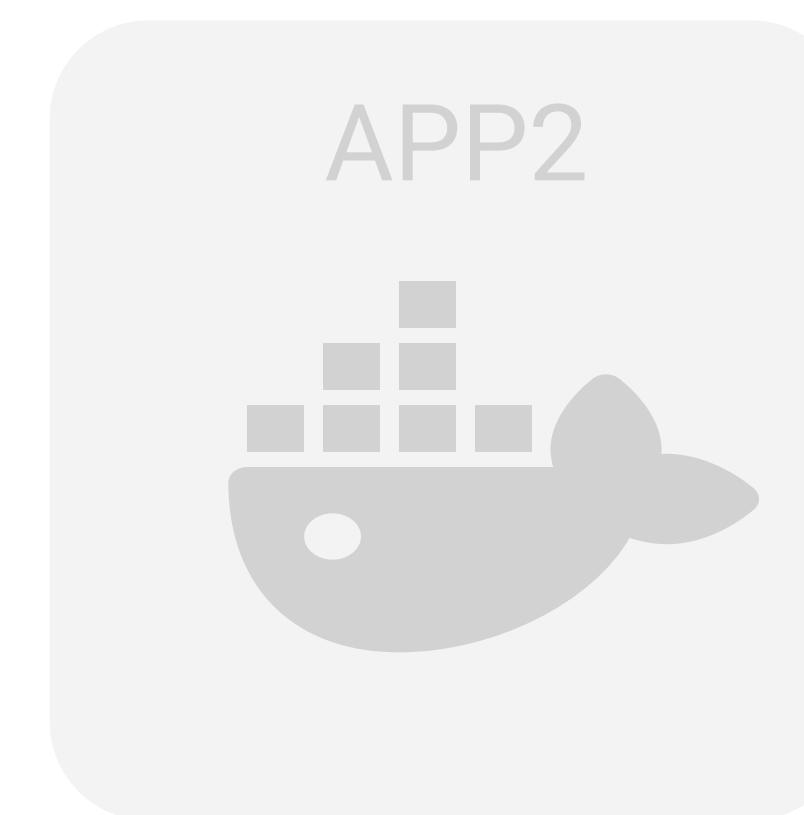
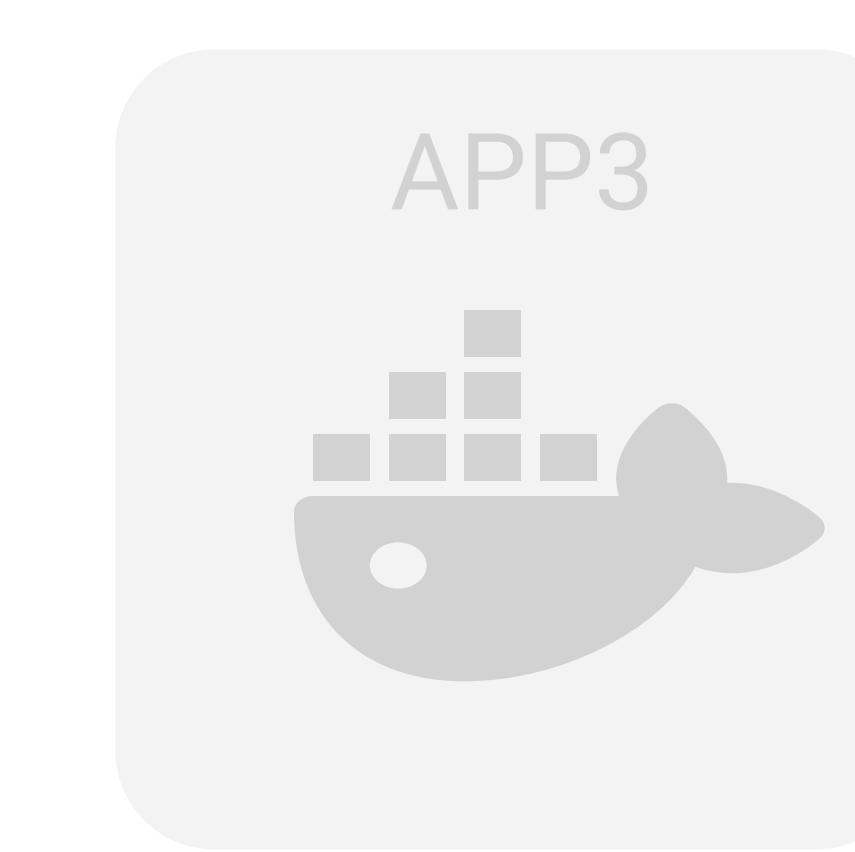


APP3





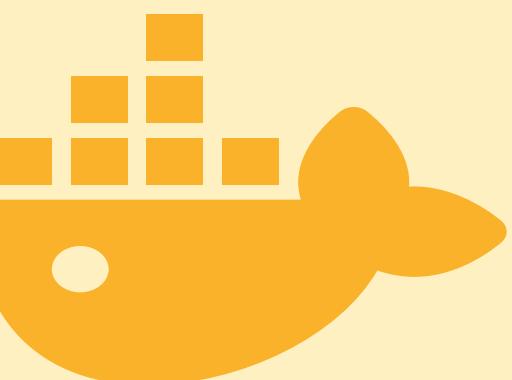
2-3. 서비스 관리



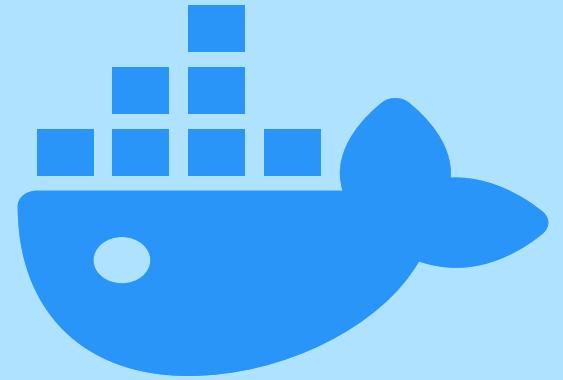


2-3. 서비스 관리

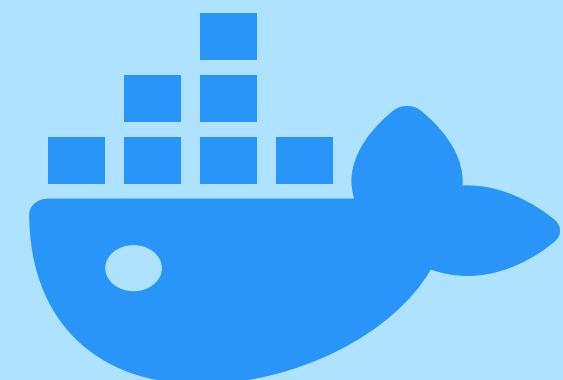
APP1



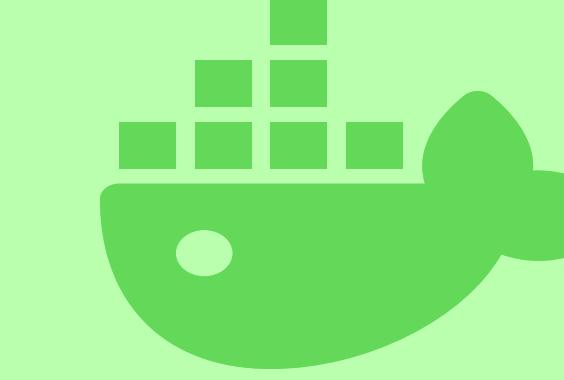
APP2



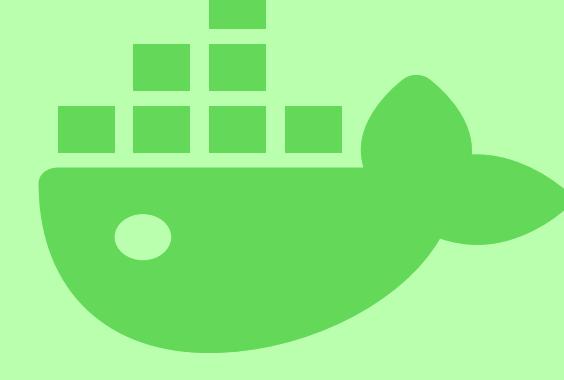
APP2



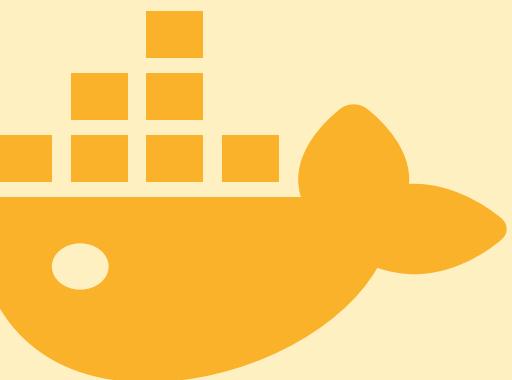
APP3



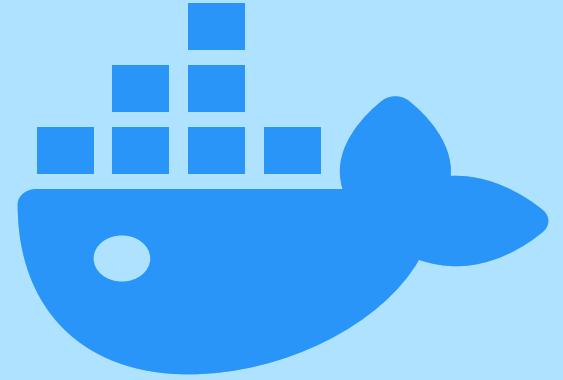
APP3



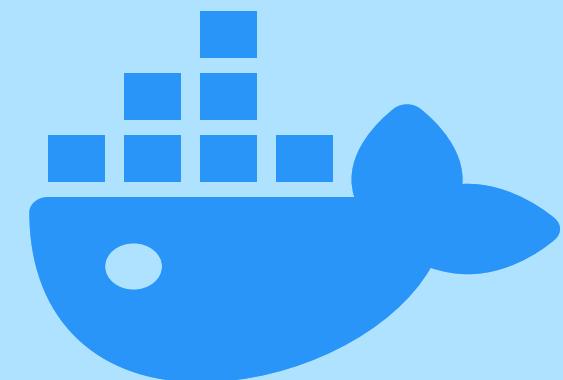
APP1



APP2



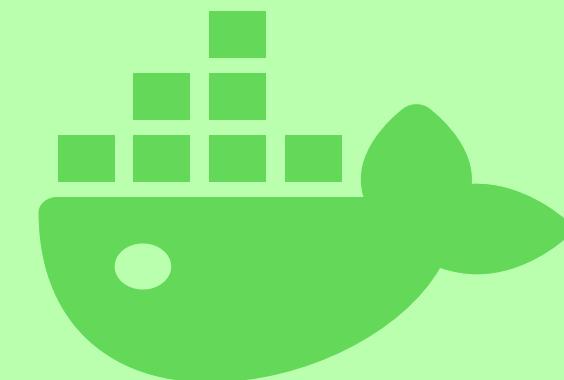
APP2



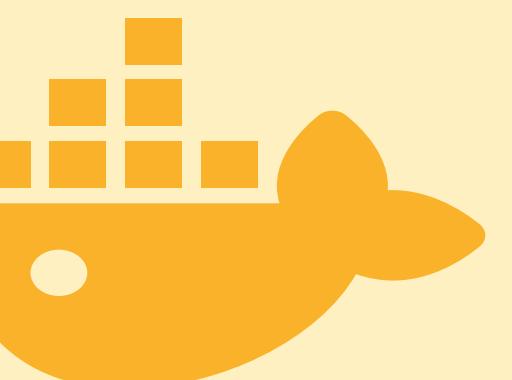
APP3



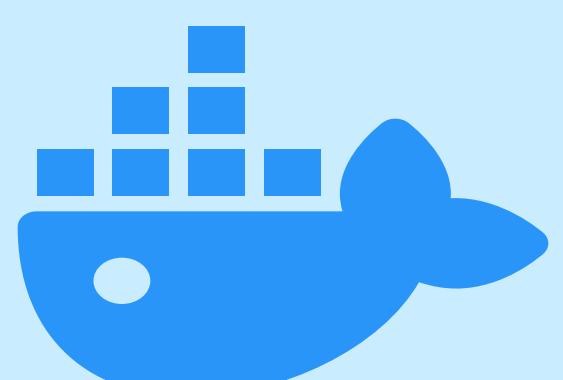
APP3



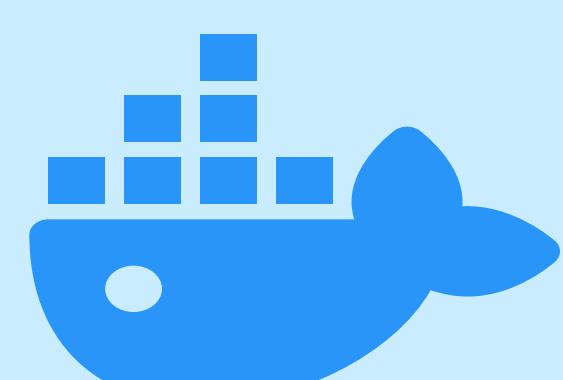
APP1



APP2



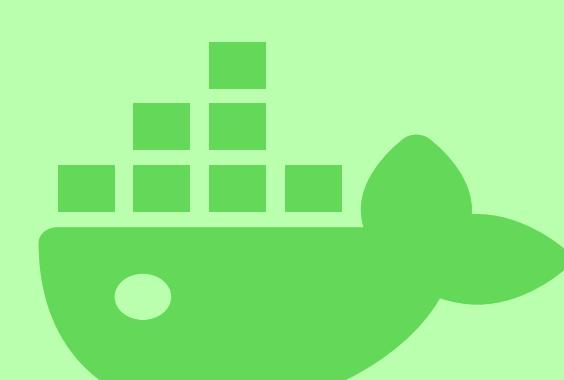
APP2



APP3



APP3

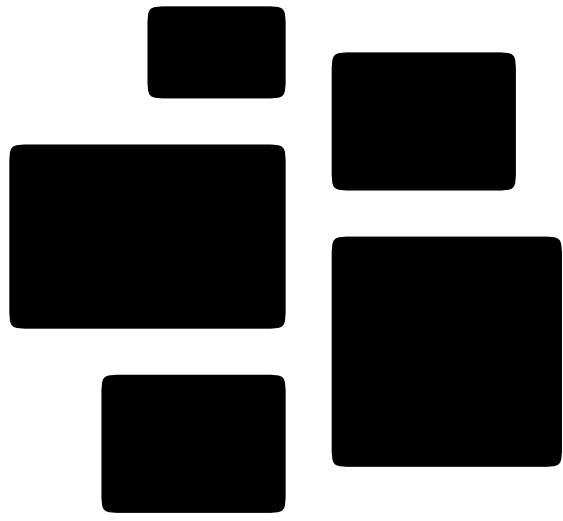


Container Orchestration

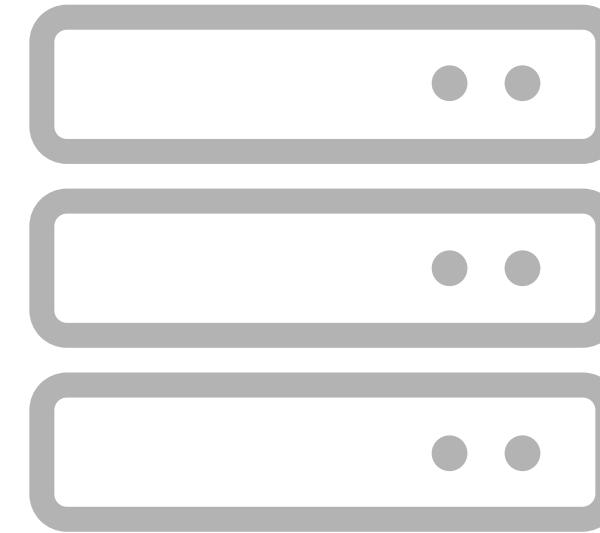
- 복잡한 컨테이너 환경을 효과적으로 관리하기 위한 도구
- 서버 관리자의 역할을 대신 할 프로그램을 만드는 도구



3-1. Container Orchestration의 기능



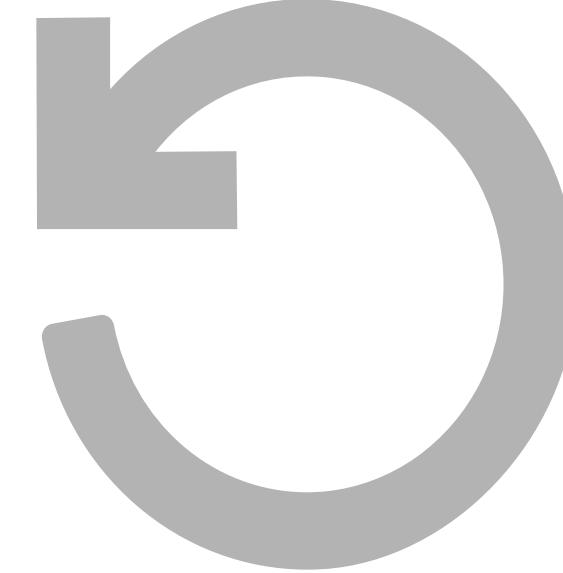
Cluster



State

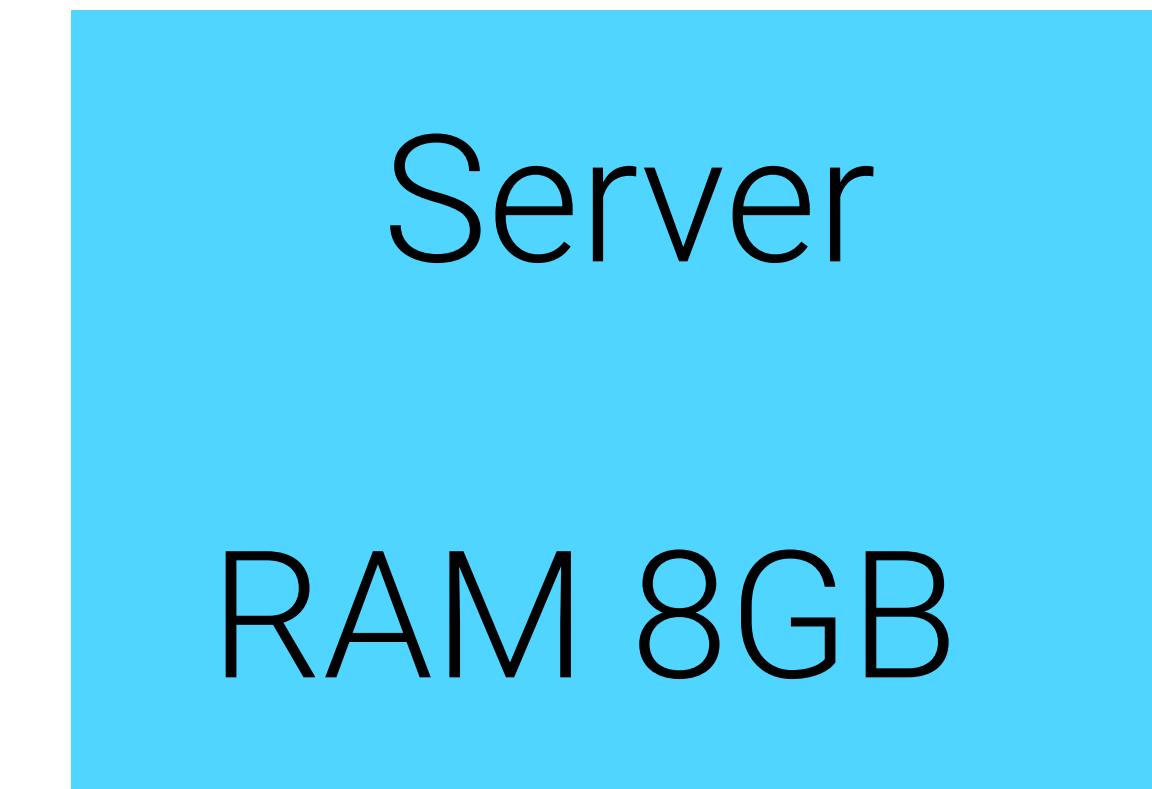
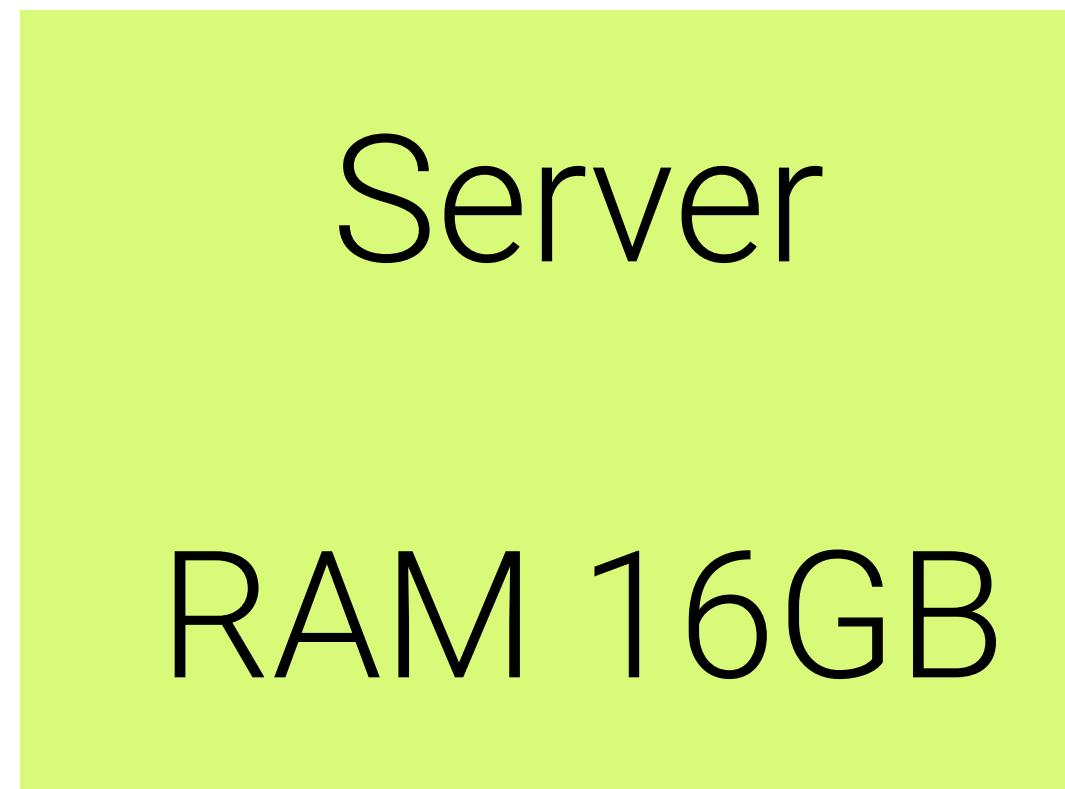


Scheduling



Rollout & Rollback

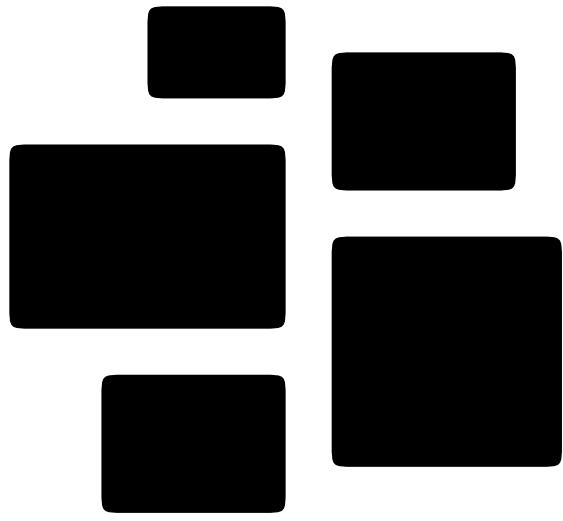
- 클러스터



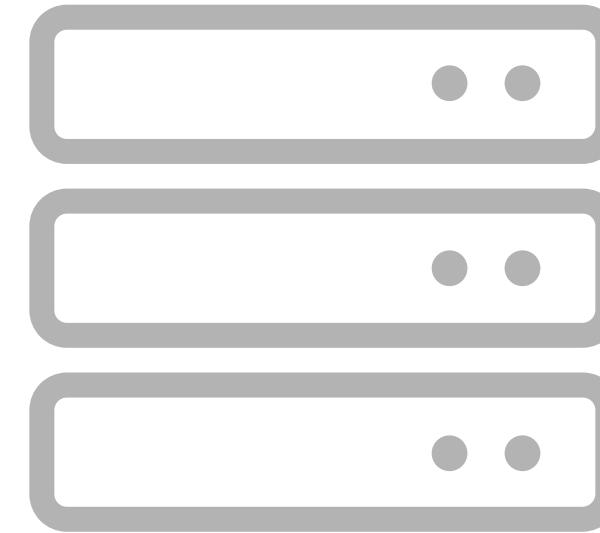
서버마다 CPU, RAM이 다른데 서버 관리자가 하나하나 알고 설정함.



3-1. Container Orchestration의 기능



Cluster



State

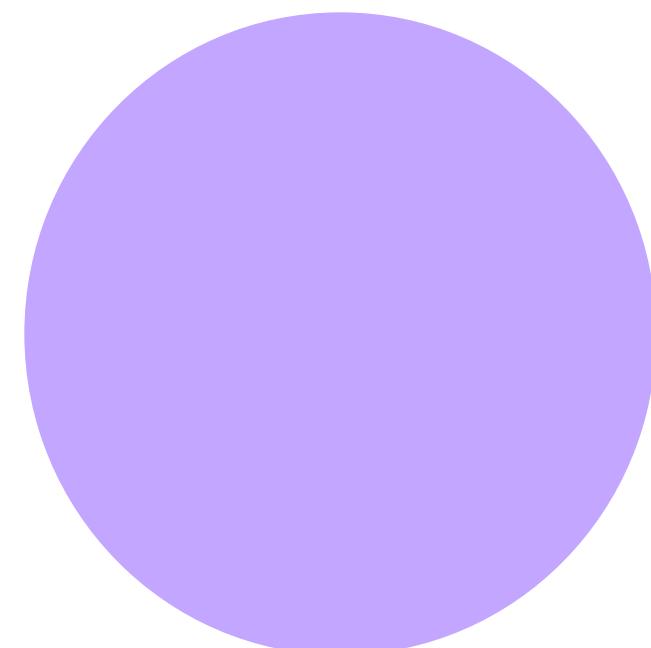


Scheduling

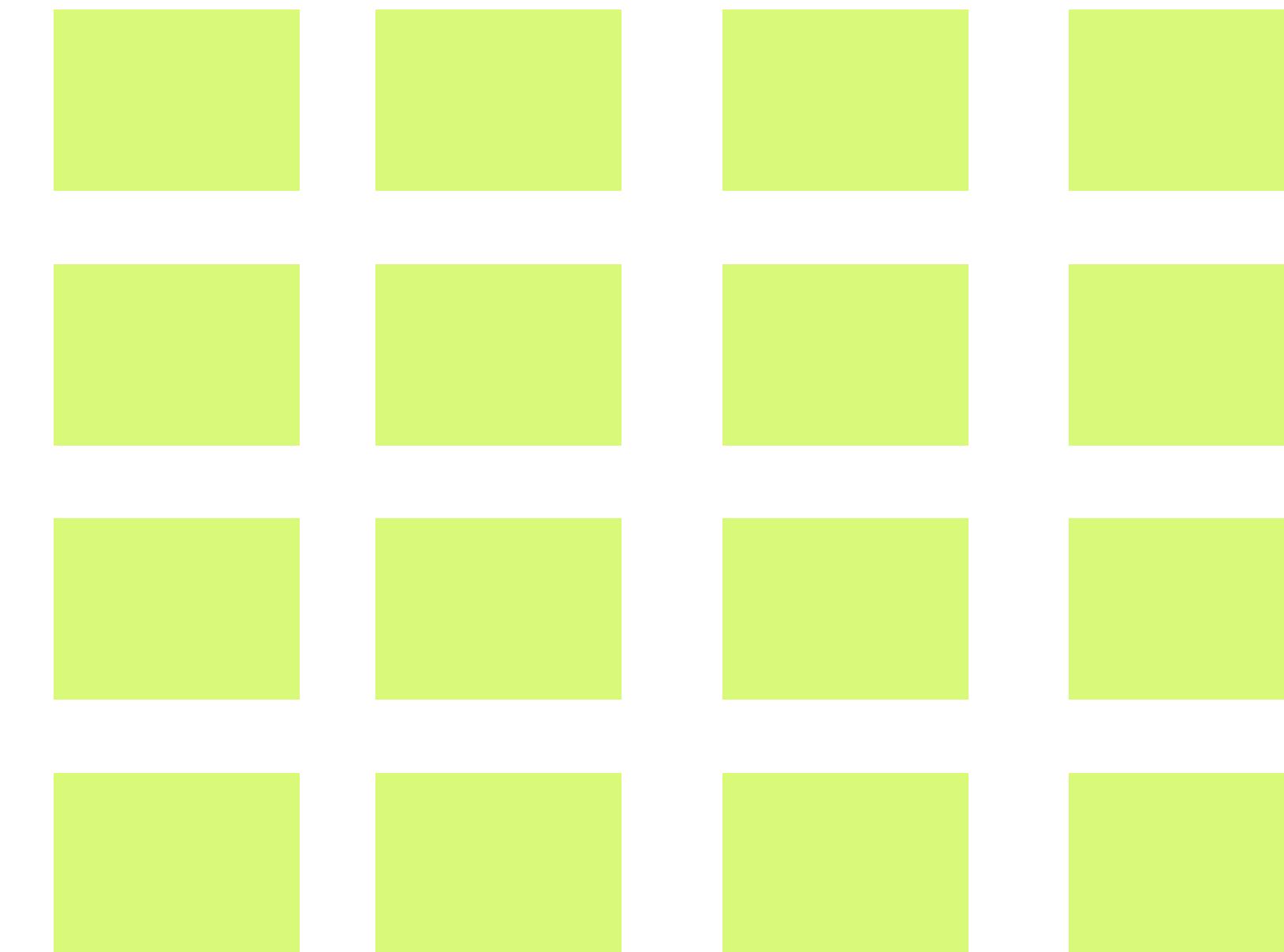


Rollout & Rollback

- 클러스터



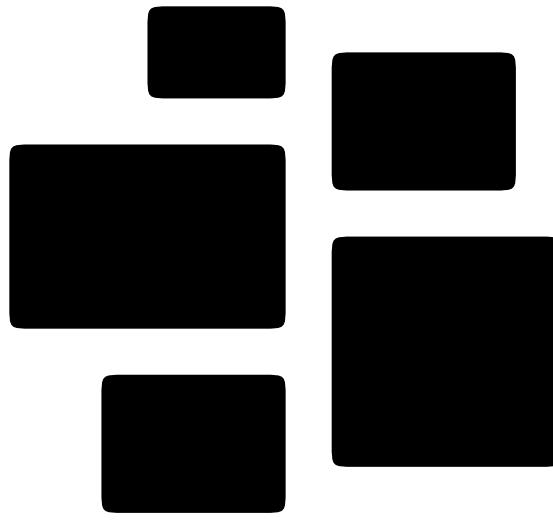
master node



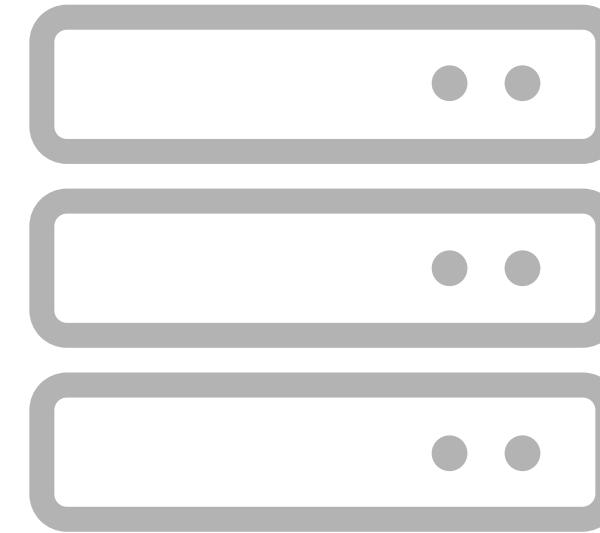
컨테이너 오케스트레이션에서는 서버를 클러스터링 해서 master node로 관리



3-1. Container Orchestration의 기능



Cluster



State

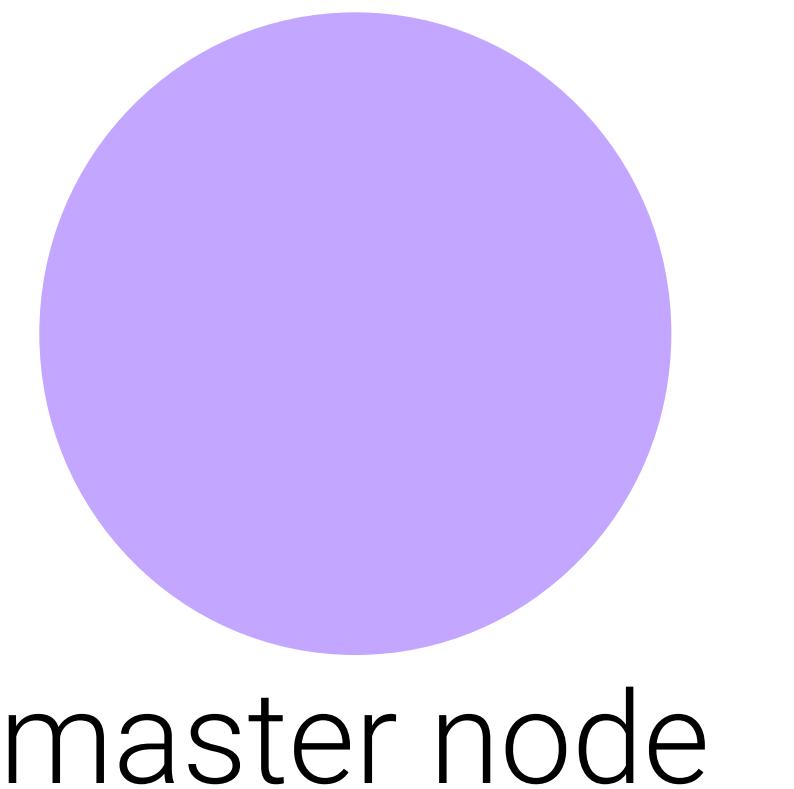


Scheduling

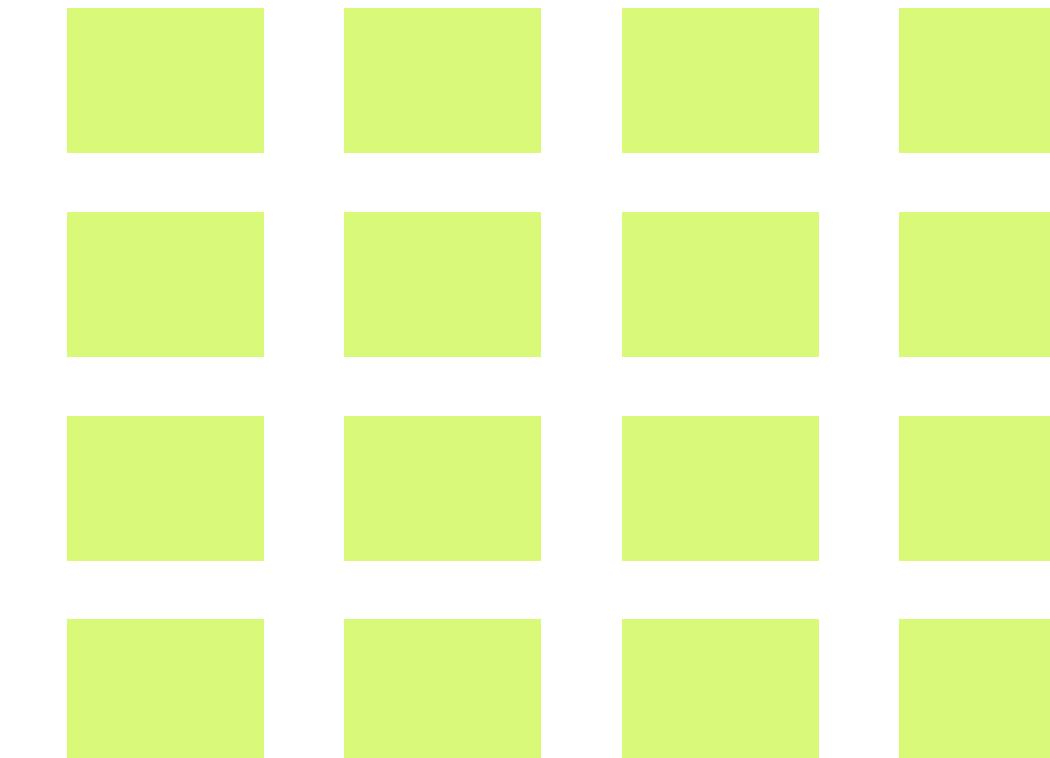


Rollout & Rollback

- 클러스터



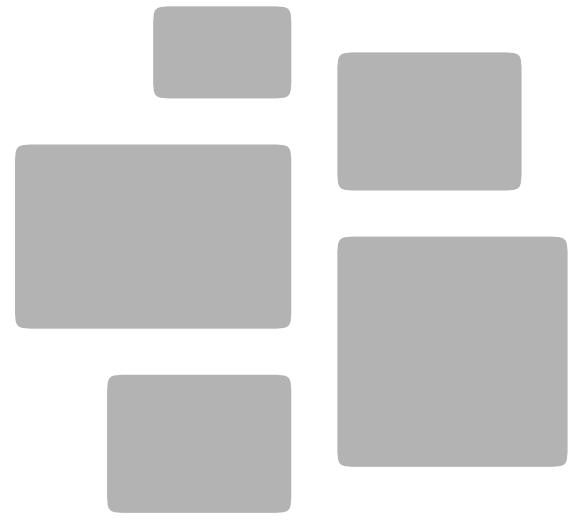
master node



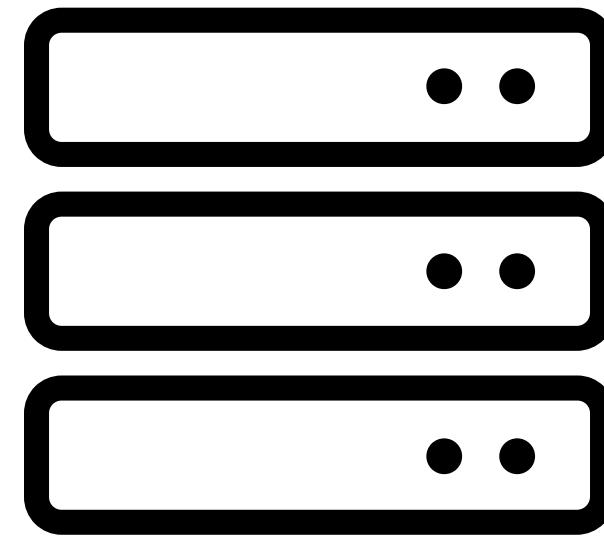
x 1000

노드의 갯수가 수천개 수만개가 되더라도 잘 돌아가야함.

3-1. Container Orchestration의 기능



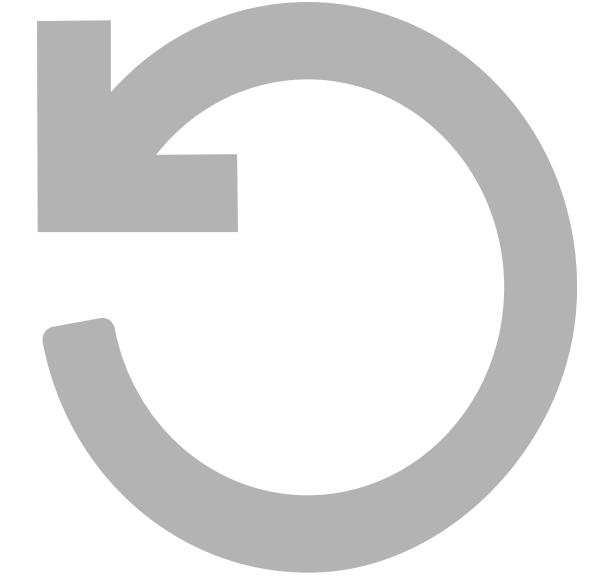
Cluster



State



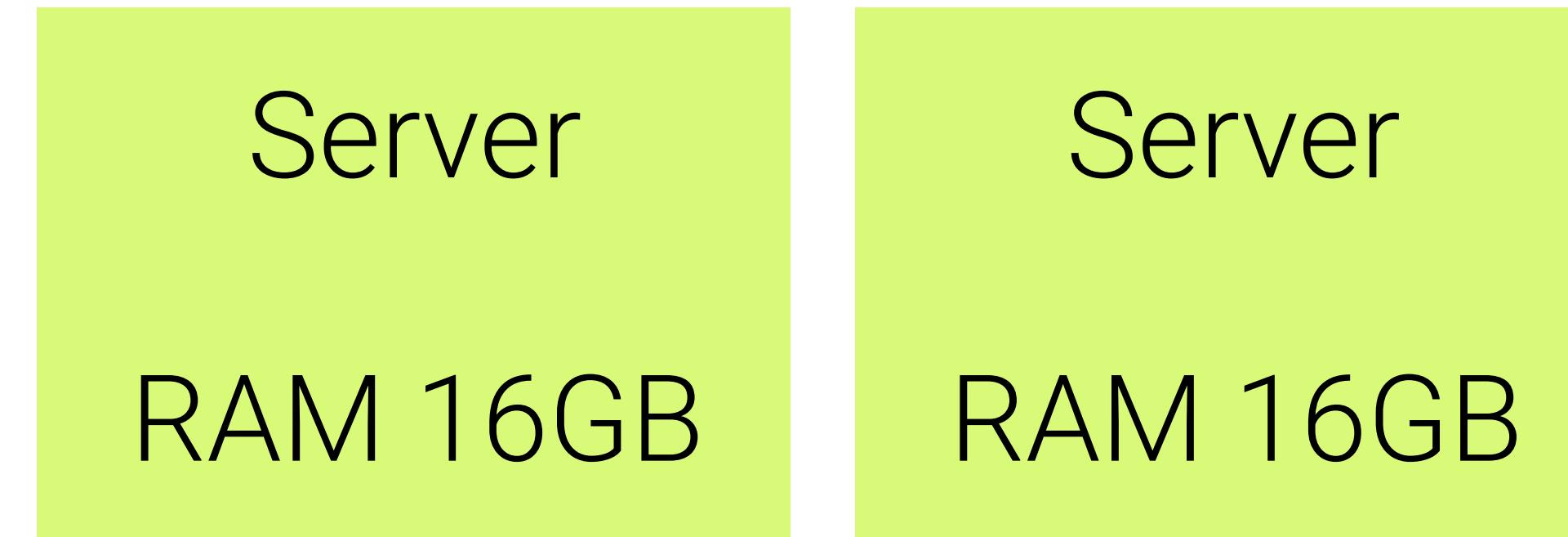
Scheduling



Rollout & Rollback

- 상태

설정 파일
replicas 2



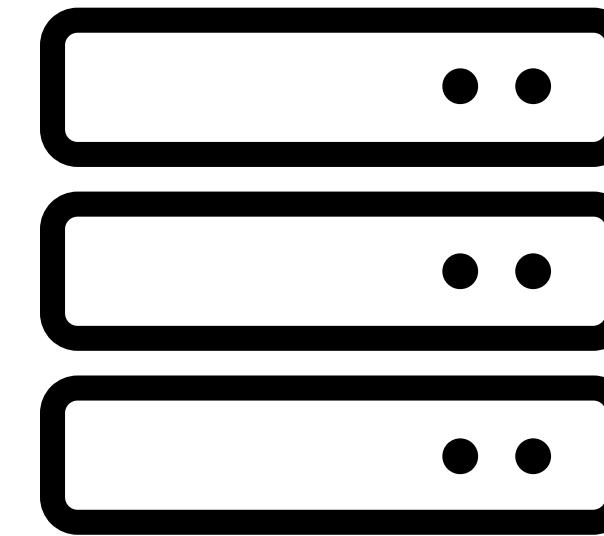
컨테이너 수량을 두개로 설정 했다면, 두개의 복제 컨테이너가 있어야 함.



3-1. Container Orchestration의 기능



Cluster



State



Scheduling



Rollout & Rollback

- 상태

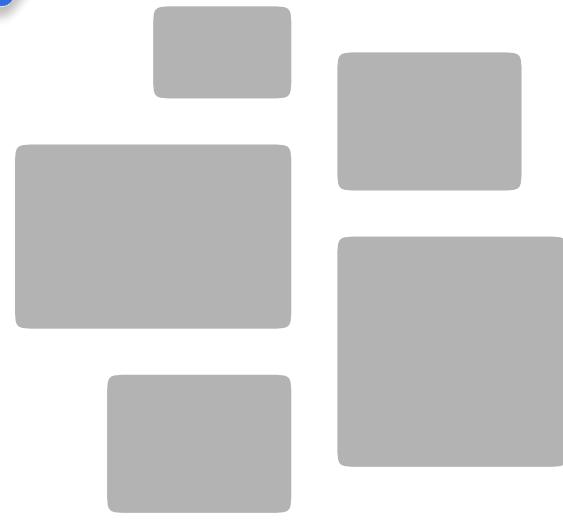
설정 파일
replicas 2



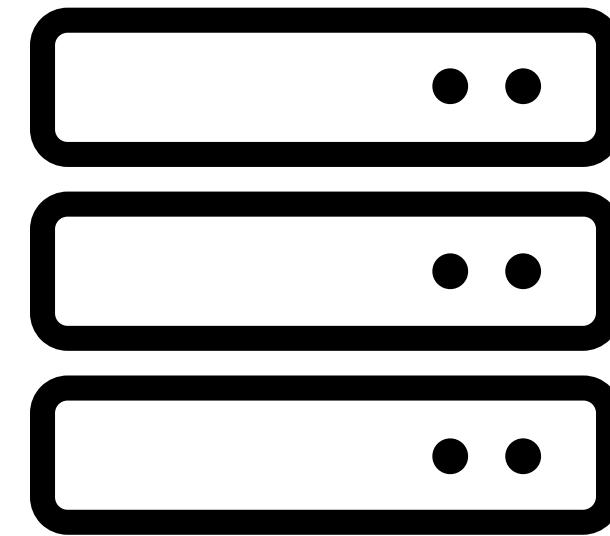
장애로 인해 상태가 변경 된 경우 설정 된 상태를 유지하기 위한 명령이 자동으로 실행 되어야 함.



3-1. Container Orchestration의 기능



Cluster



State



Scheduling

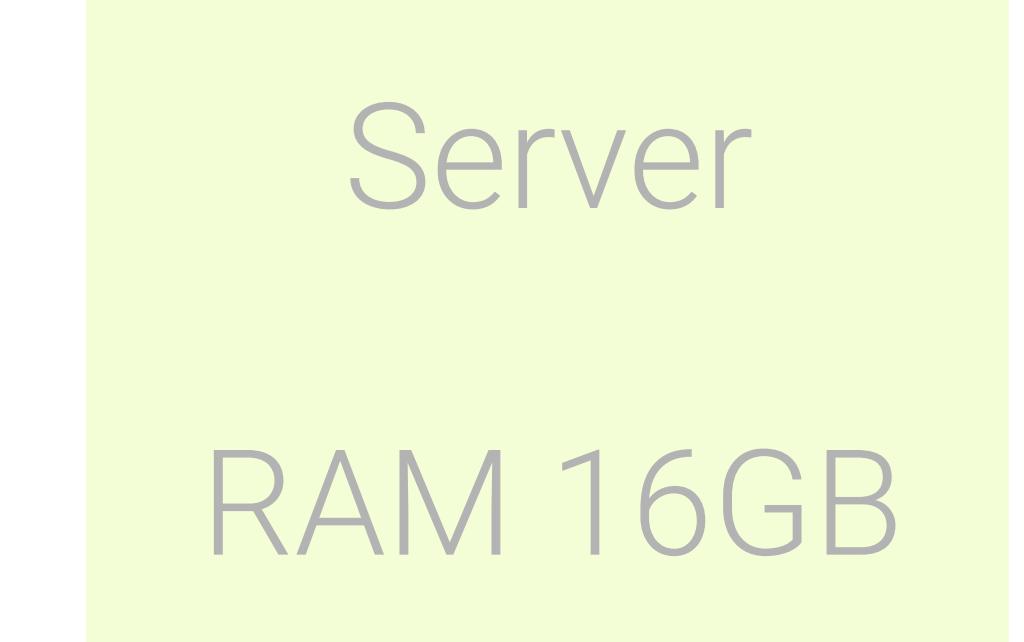
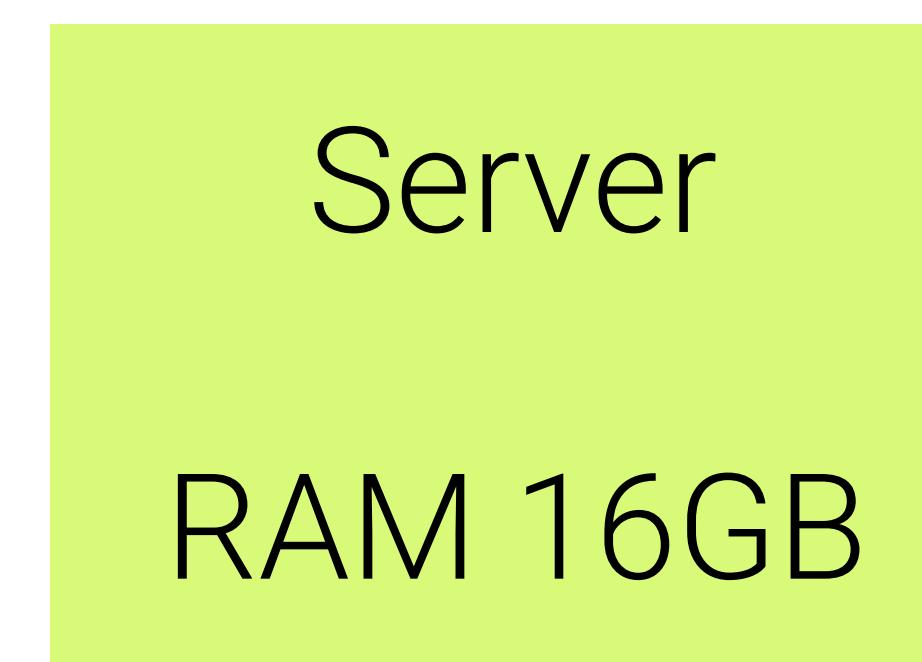


Rollout & Rollback

- 상태

설정 파일

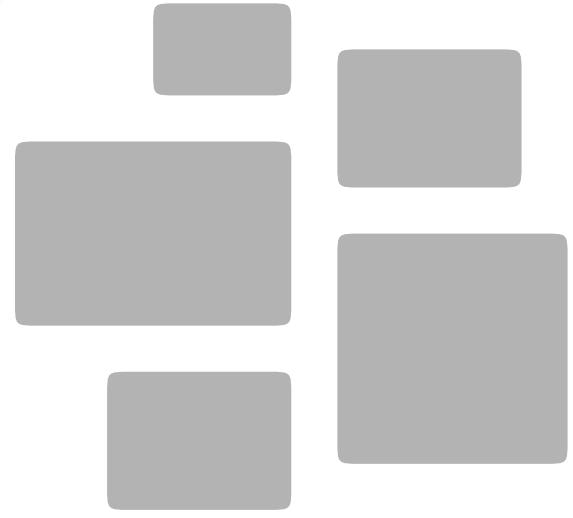
replicas 2 -> 3



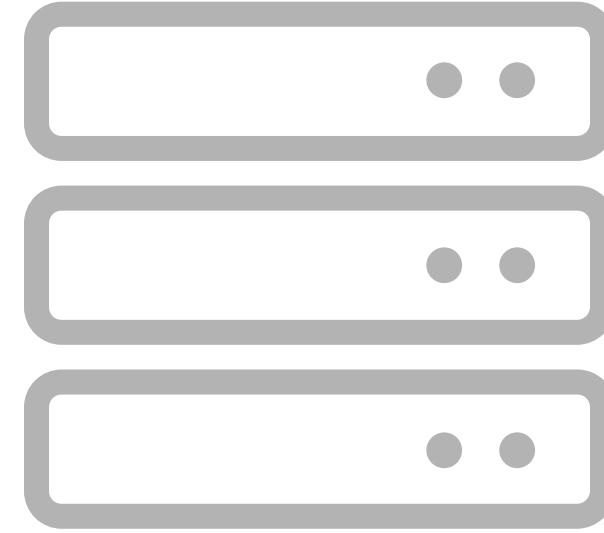
설정을 변경하면 별도의 명령없이 해당 상태로 변경될 수 있어야 함.



3-1. Container Orchestration의 기능



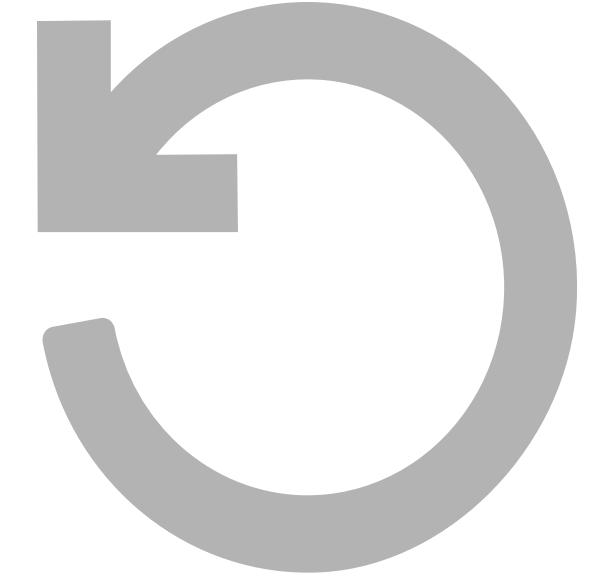
Cluster



State

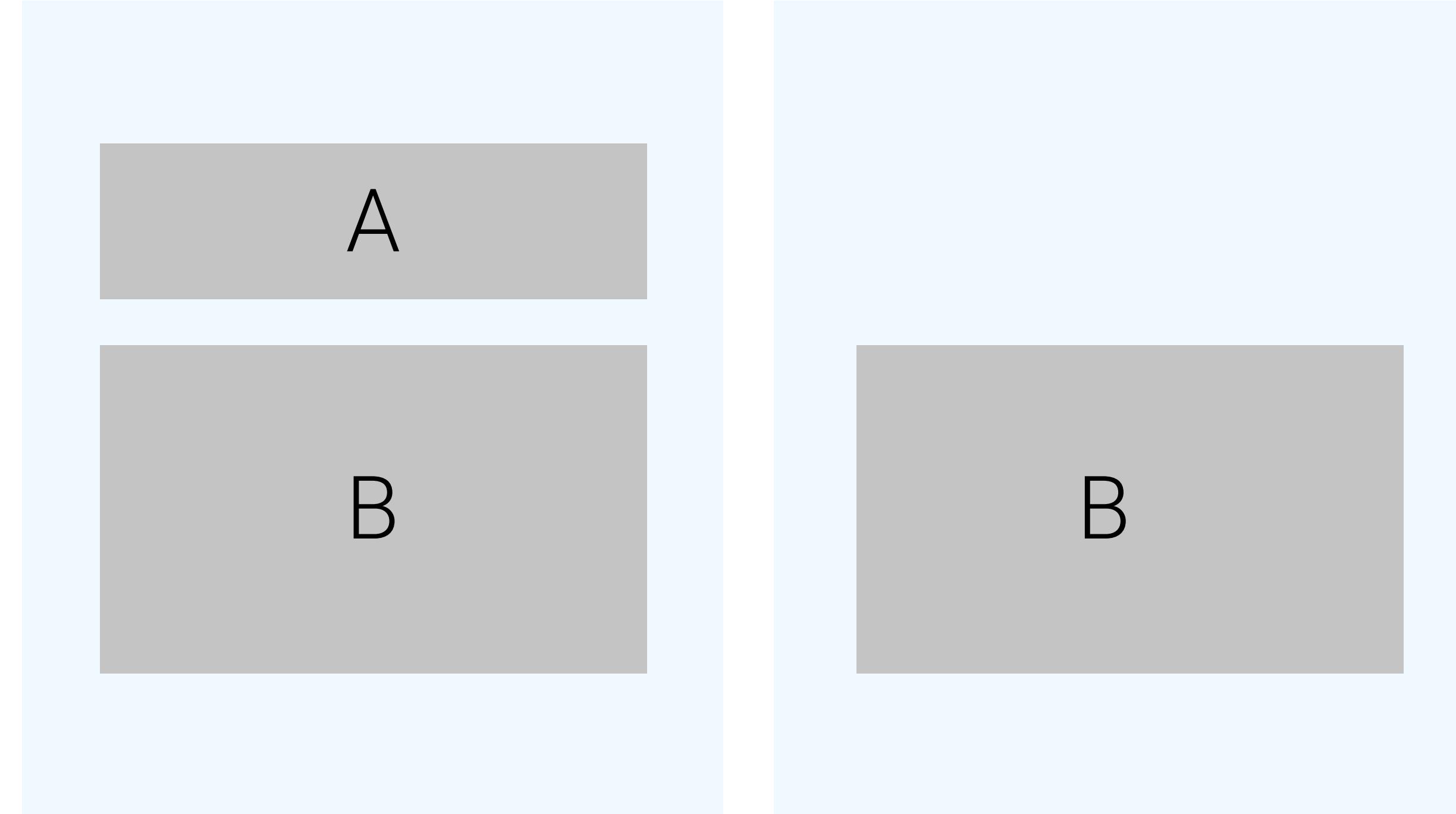


Scheduling



Rollout & Rollback

- 배포 관리



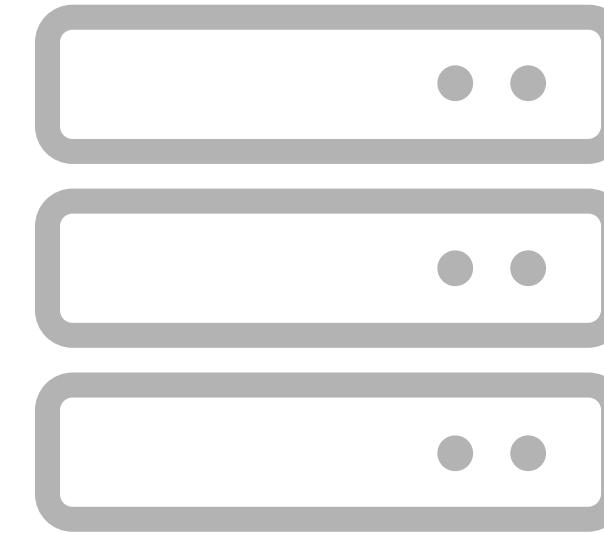
유휴 리소스를 관리하여 새로운 컨테이너 생성 시 알맞은 서버에 배치 시킬 수 있어야 함.



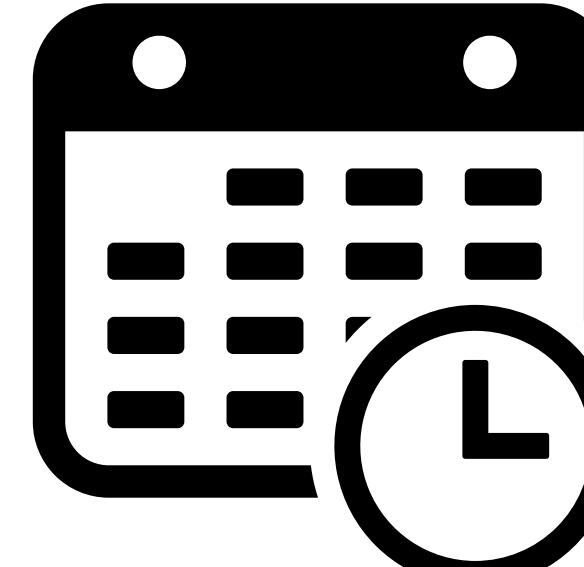
3-1. Container Orchestration의 기능



Cluster



State

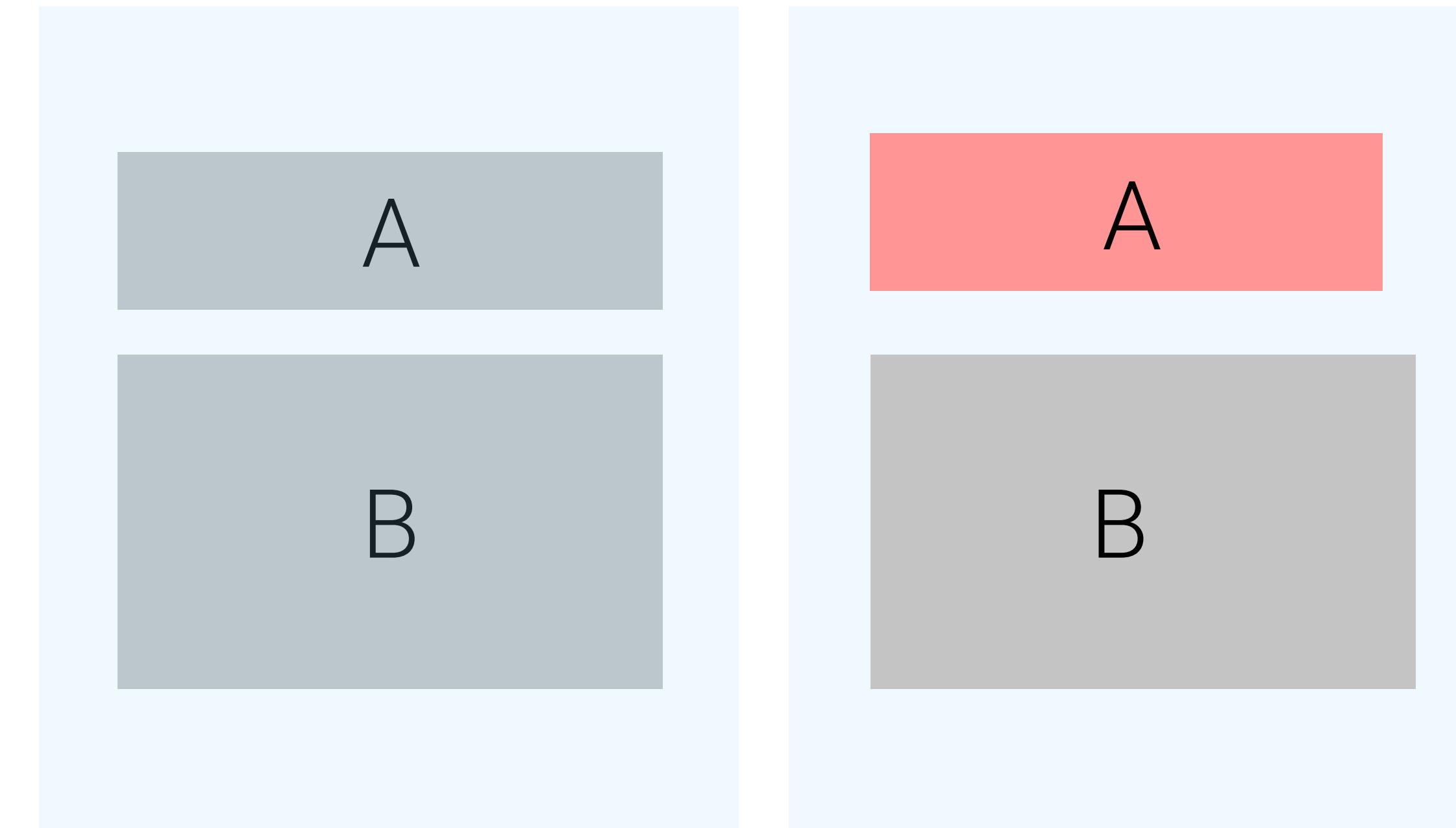


Scheduling



Rollout & Rollback

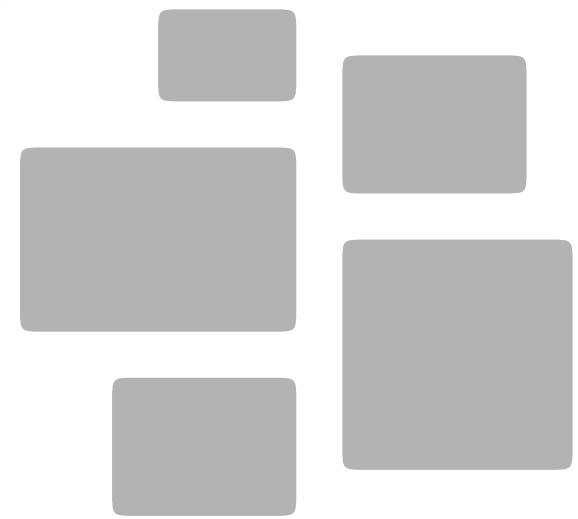
- 배포 관리



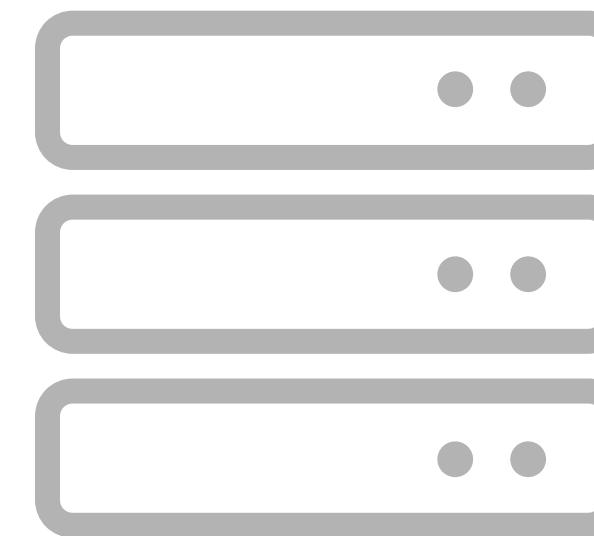
유휴 리소스를 관리하여 새로운 컨테이너 생성 시 알맞은 서버에 배치 시킬 수 있어야 함.



3-1. Container Orchestration의 기능



Cluster



State

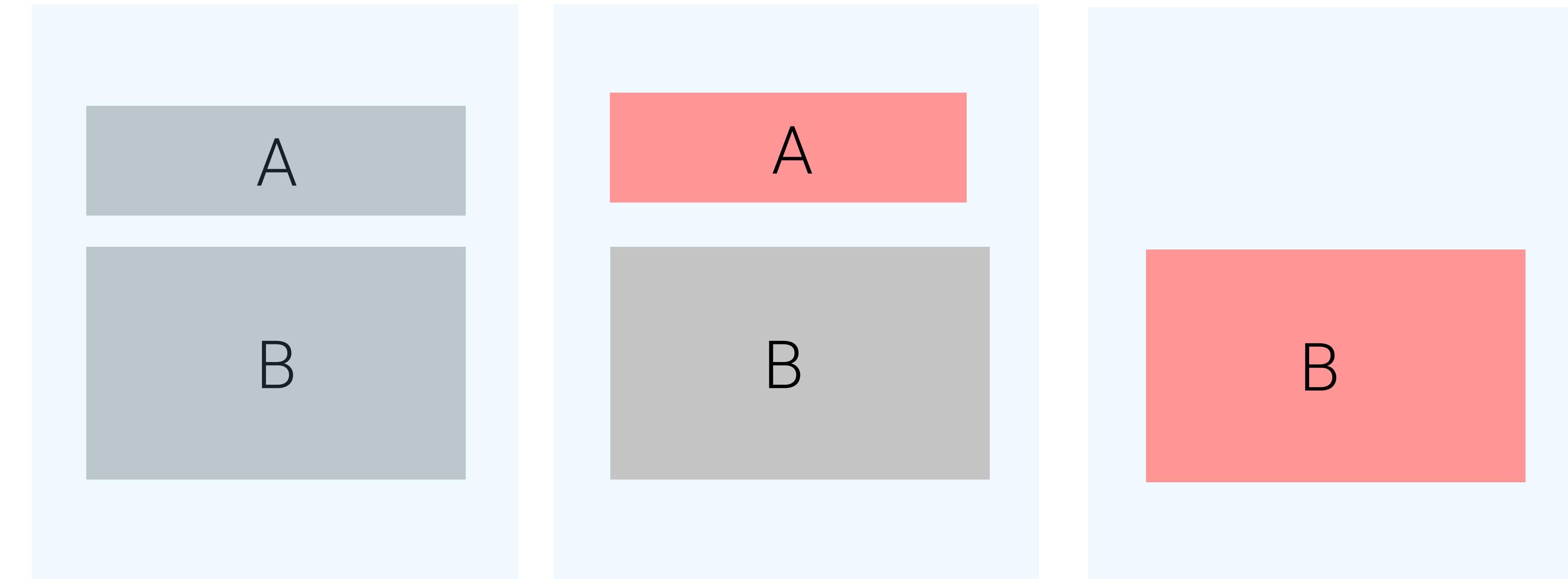


Scheduling



Rollout & Rollback

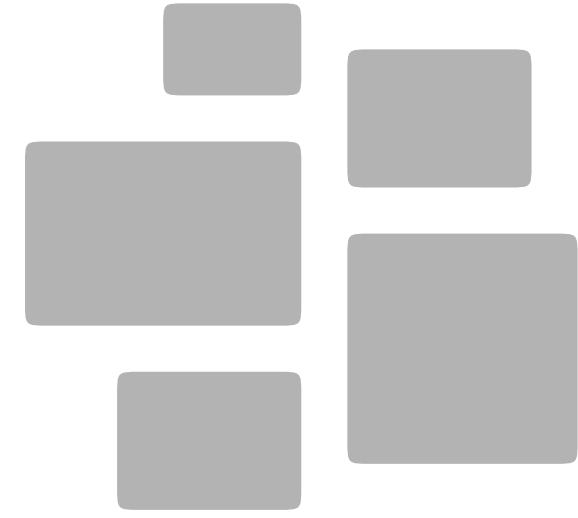
- 배포 관리



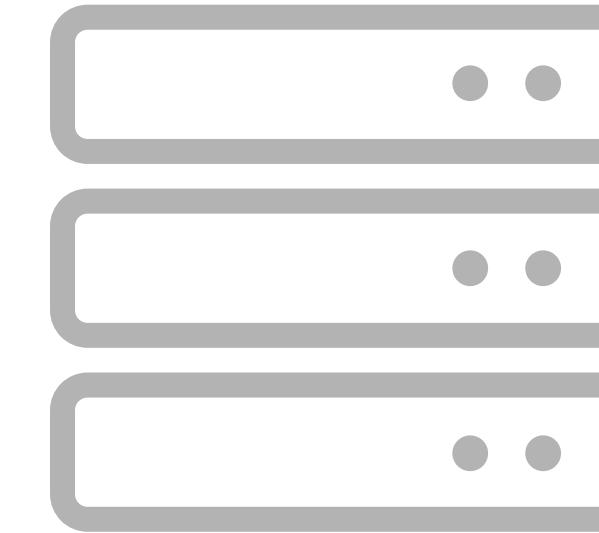
추가 리소스가 필요 한 경우 리소스를 할당하여 컨테이너를 배치 시킬 수 있어야 함.



3-1. Container Orchestration의 기능



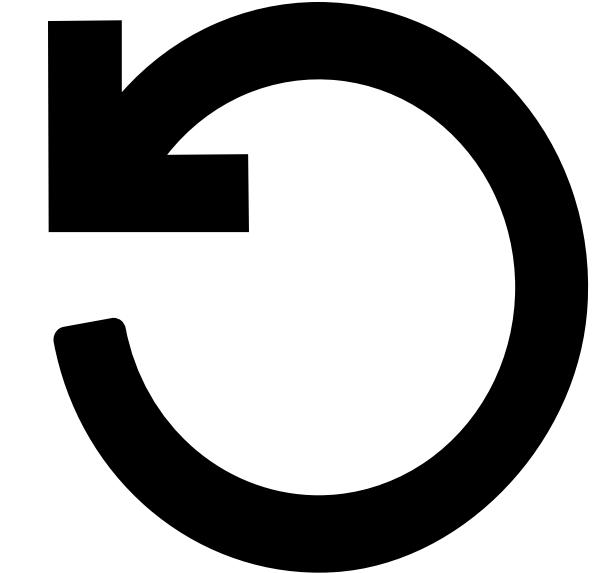
Cluster



State



Scheduling

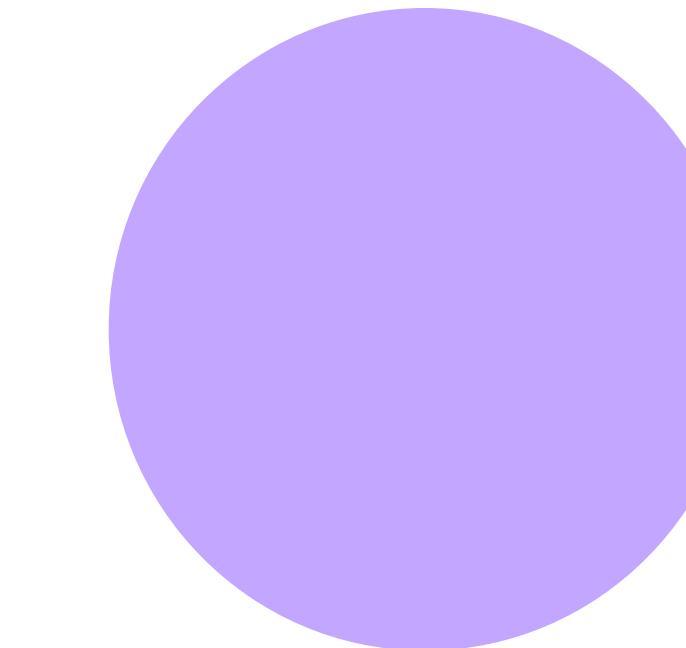


Rollout & Rollback

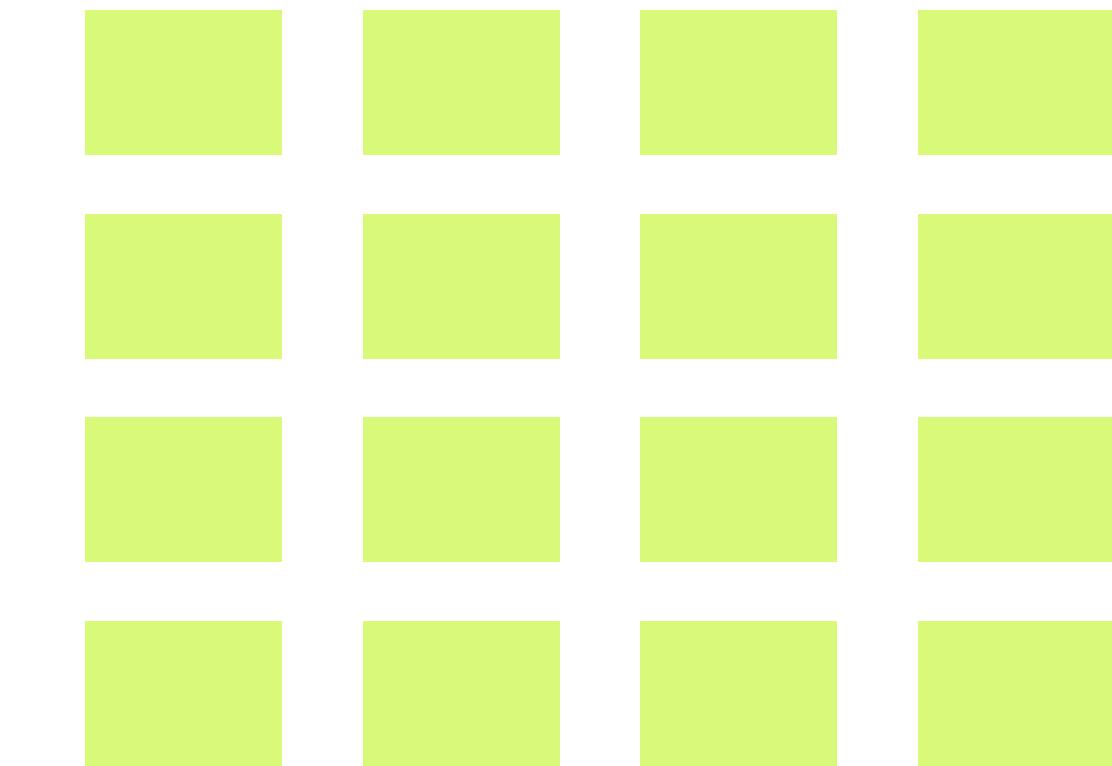
- 배포 버전관리



update!!



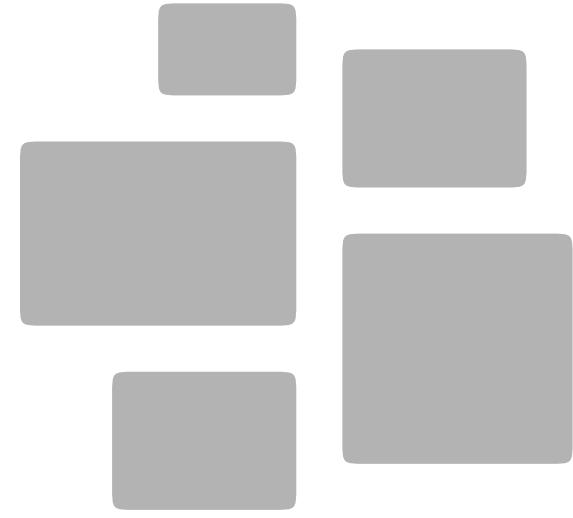
master node



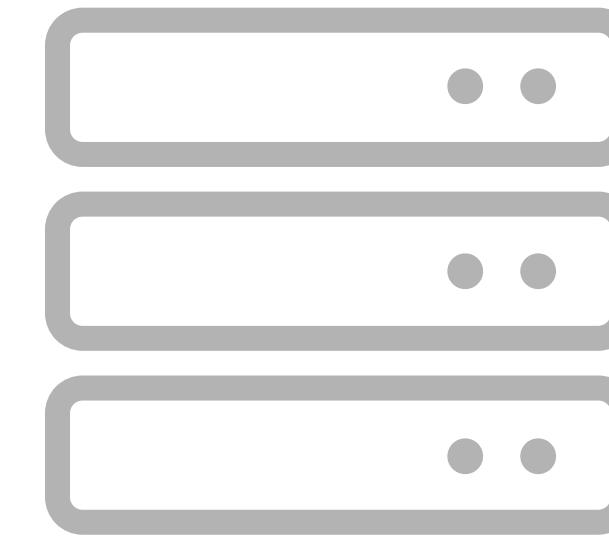
배포 시 개별 배포가 아닌 중앙에서 자동 배포가 진행될 수 있어야 함.



3-1. Container Orchestration의 기능



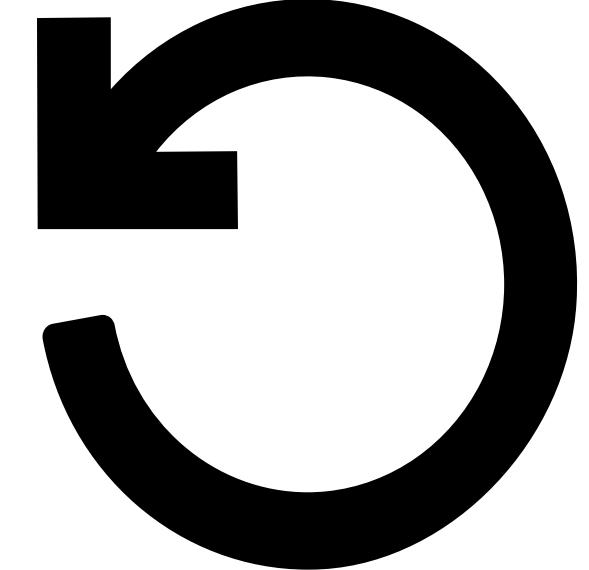
Cluster



State



Scheduling

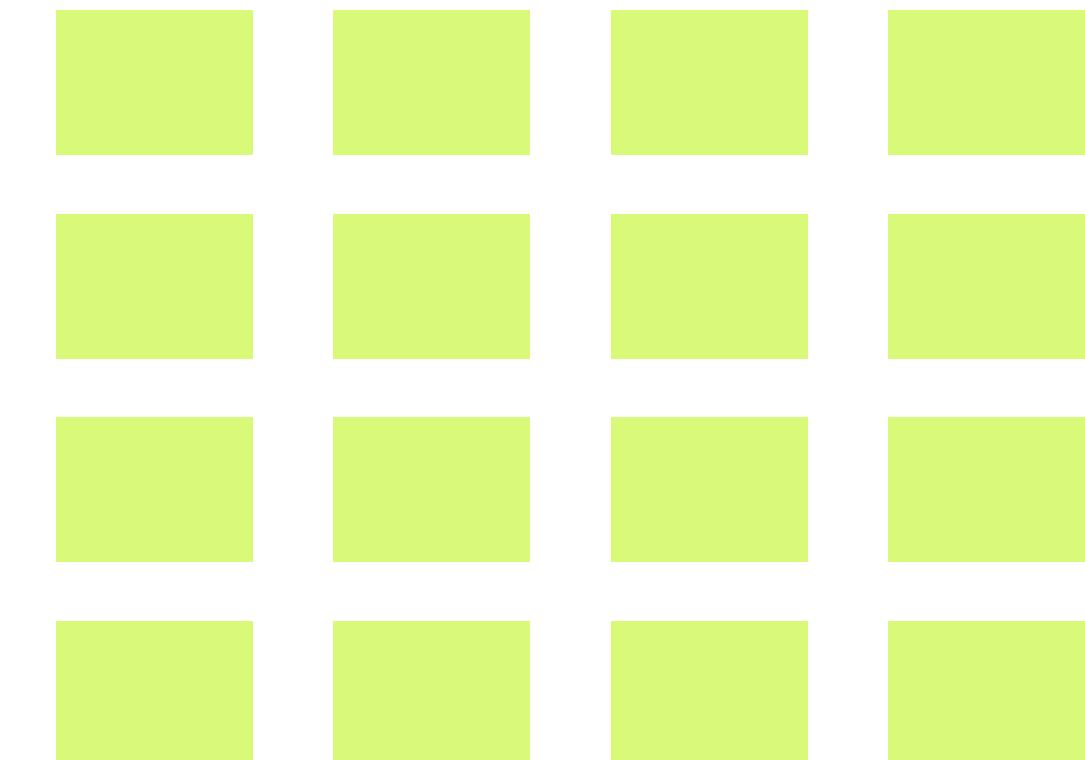


Rollout & Rollback

- 배포 버전관리



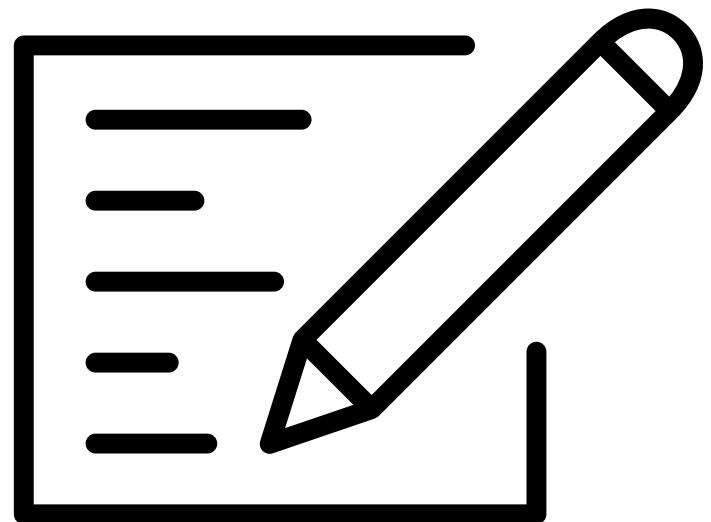
Rollback!!



롤백의 경우에도 중앙에서 관리하여 일괄적으로 롤백이 진행될 수 있어야 함.

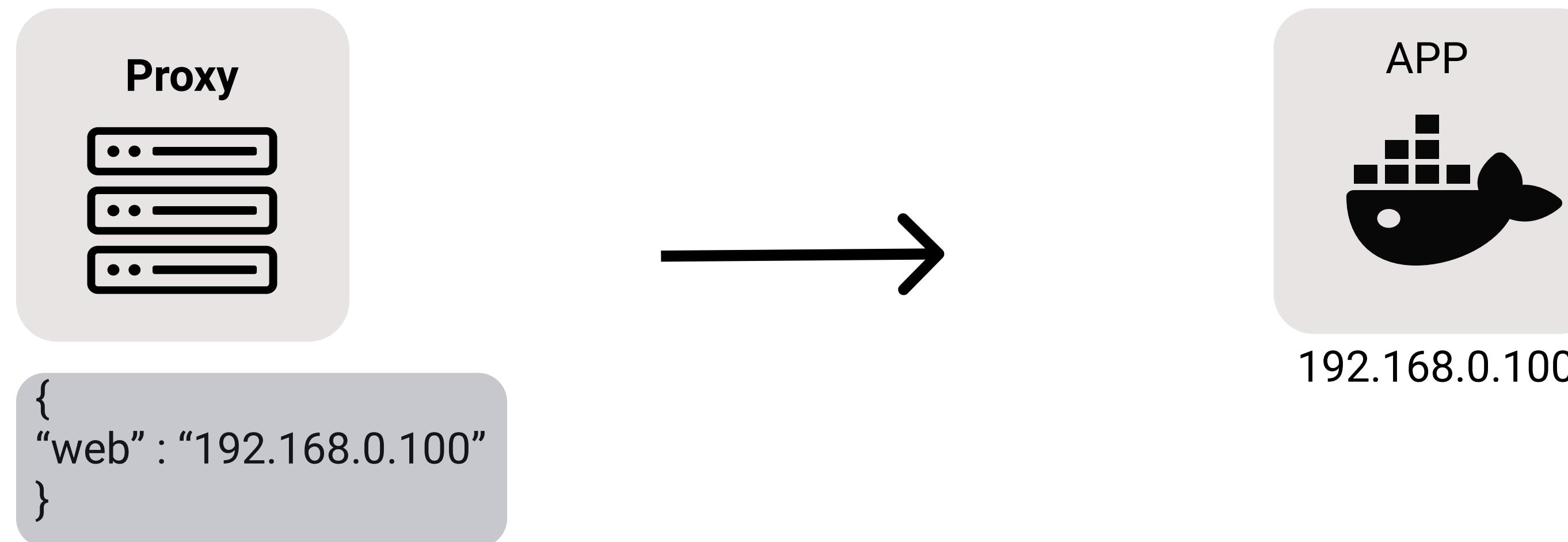


3-1. Container Orchestration의 기능



Service Discovery

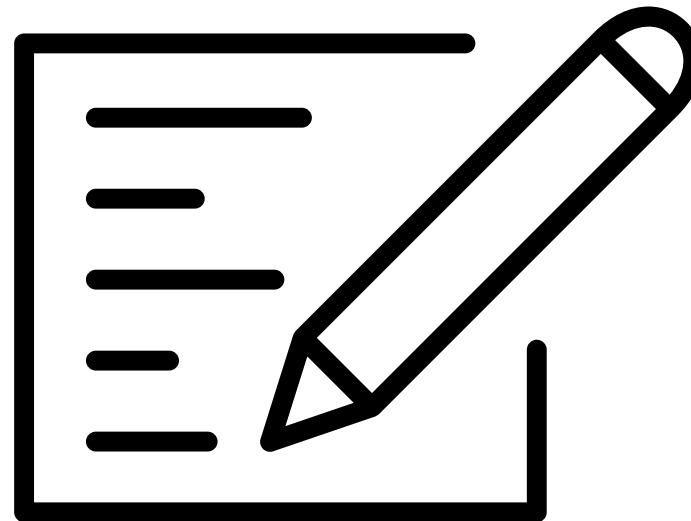
- 서비스 등록 조회



새로운 서비스가 추가 된 경우 자동으로 해당 서비스에 대한 설정이 추가 되어야 함.



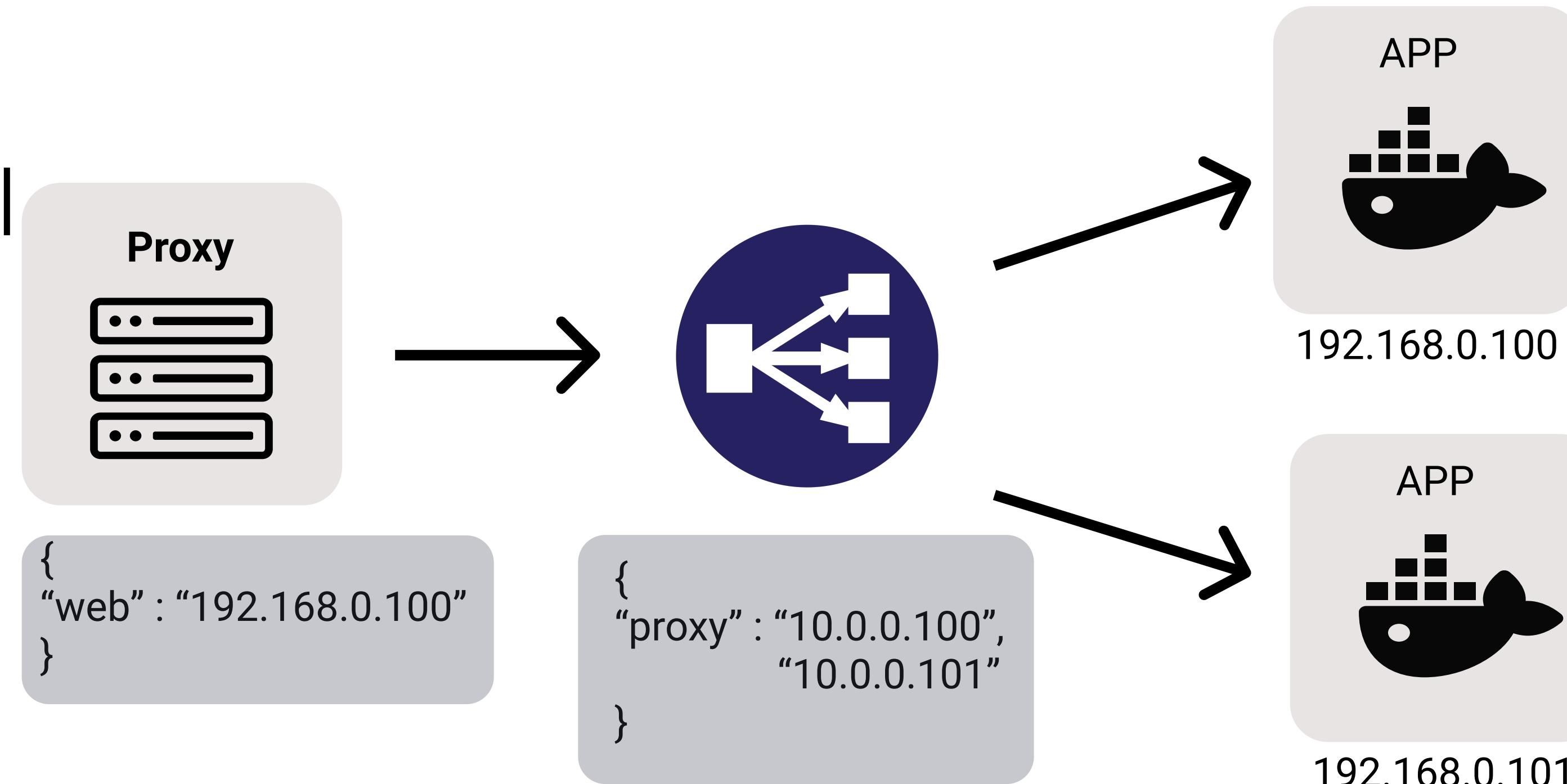
3-1. Container Orchestration의 기능



Service Discovery

- 서비스 등록 조회

난 암것두 안함..!



새로운 서비스가 추가 된 경우 자동으로 해당 서비스에 대한 설정이 추가 되어야 함.

왜



kubernetes

일까?



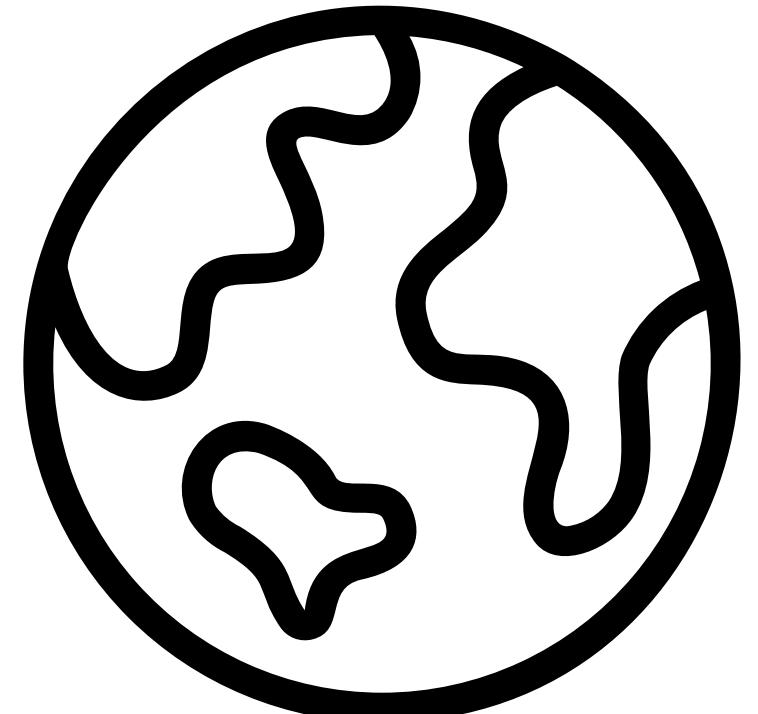
kubernetes

- 컨테이너를 쉽고 빠르게 배포/확장하고 관리를 자동화 해주는 오픈소스 플랫폼
- 2015년 구글에 의해 공개(v1.0 release)
 - 1주일에 20억개의 컨테이너를 생성
 - 컨테이너 배포/관리를 위해 사용하던 brog를 기반으로 만든 오픈소스

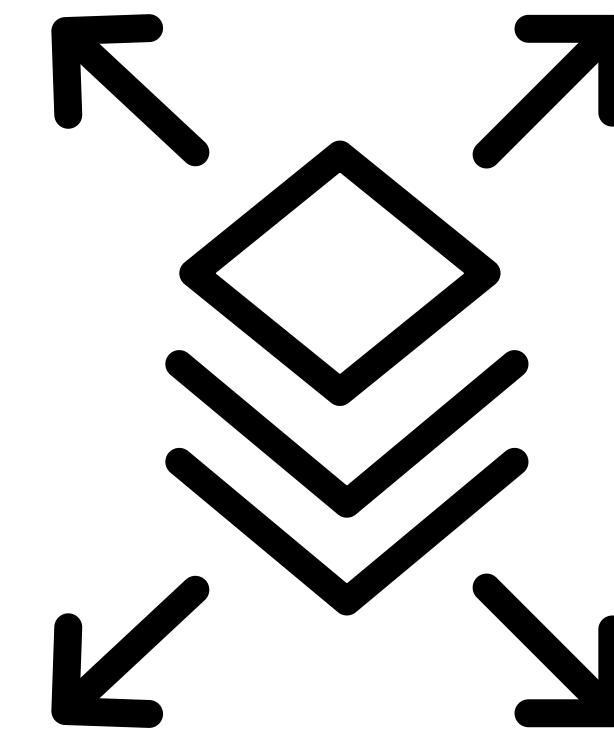


kubernetes

- 정의
 - 운영 환경에서 사용 가능한 컨테이너 오케스트레이션
 - 자동화된 컨테이너 배포, 스케일링과 관리



Planet Scale



Never Outgrow



Run AnyWhere



kubernetes

엄청난
인기!!

star: 80.9k

The screenshot shows the GitHub repository page for 'kubernetes / kubernetes'. The top navigation bar includes links for Code, Issues (1.7k), Pull requests (1k), Actions, Projects (6), Security, and Insights. On the right, there are buttons for Watch (3.3k), Star (80.9k), Fork (29.5k), and a dropdown for Code. Below the navigation, a list of recent commits is displayed under the 'master' branch, with 47 branches and 827 tags available. The commits are from various contributors, including 'k8s-ci-robot', and cover updates like 'update enhancement issue template to point to KEPs' and 'Merge pull request #104399'. To the right, the 'About' section describes Kubernetes as a 'Production-Grade Container Scheduling and Management' system, linking to 'kubernetes.io'. It also lists 'Readme', 'Apache-2.0 License', and 'Releases' (827). The latest release is 'Kubernetes v1.22.1'.

Author	Commit Message	Time Ago
k8s-ci-robot	Merge pull request #105034 from pacoxu/kubeadm-insecure...	73d51a2 4 hours ago
.github	.github: update enhancement issue template to point to KEPs	7 months ago
CHANGELOG	CHANGELOG: Update directory for v1.23.0-alpha.2 release	16 hours ago
LICENSES	klog 2.20.0, logr v1.1.0, zapr v1.1.0	6 days ago
api	Merge pull request #104399 from tkashem/apf-v1beta2	2 days ago
build	Merge pull request #104749 from cpanato/GH-102822	22 hours ago
cluster	Merge pull request #102592 from pacoxu/patch-11	6 hours ago
cmd	kubeadm: remove --port from kube-scheduler manifest	10 hours ago



kubernetes

엄청난 확장성

- 쿠버네티스를 기반으로 동작하는 플랫폼들이 많아지고 있음.

- ML : Kubeflow
- CI/CD : TEKTION
- Service Mesh : Istio
- Serverless : Knative



Kubeflow





kubernetes

컨테이너 오케스트레이션의

(사실상) 표준!!