2nd Week

Today's contents

▷ Kubernetes Overview

▷ Pod & Namespace

///

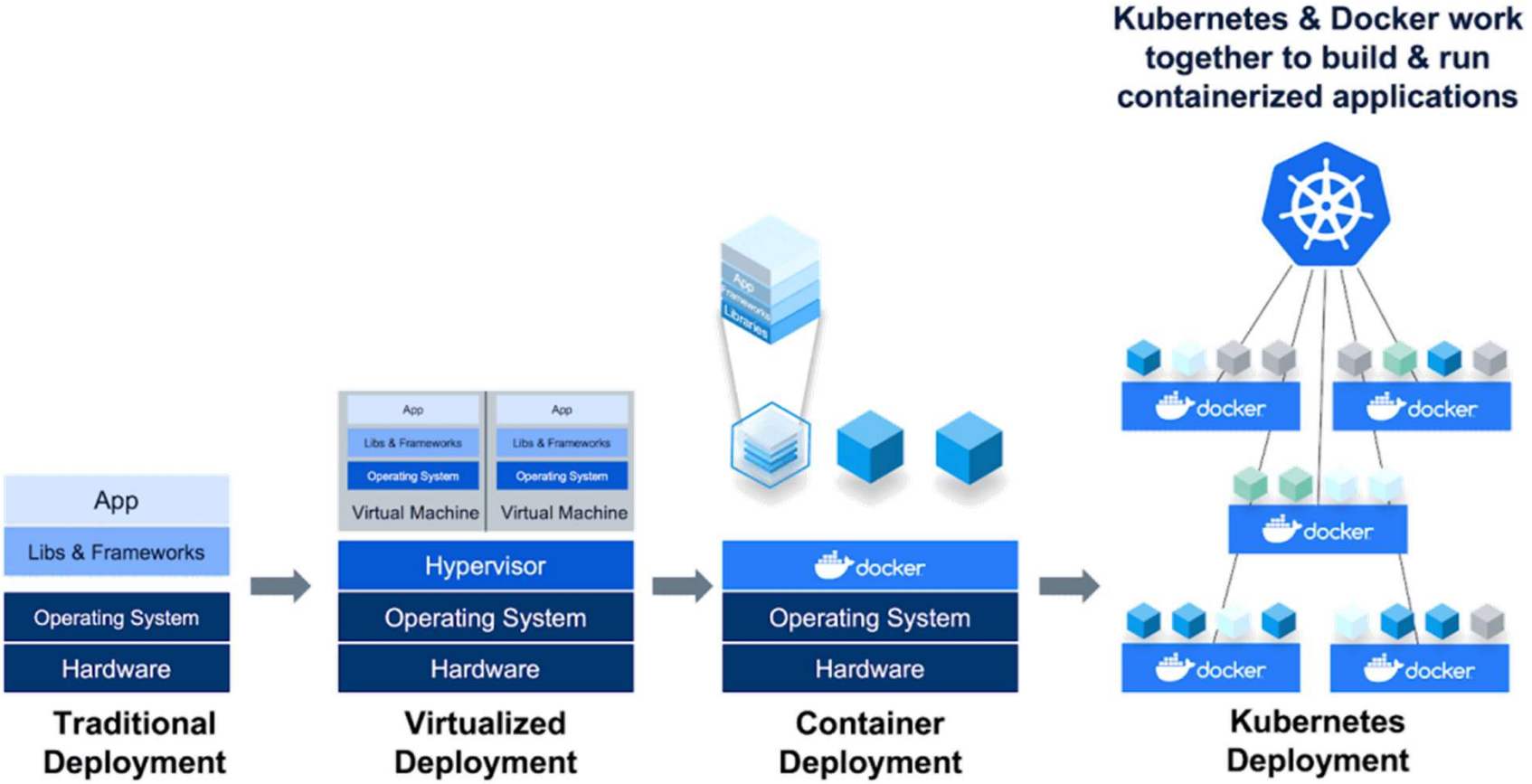
Kubernetes Overview

Kubernetes is ...

Kubernetes, also known as K8s,
is an open-source system for
automating deployment, scaling, and management
of containerized applications.

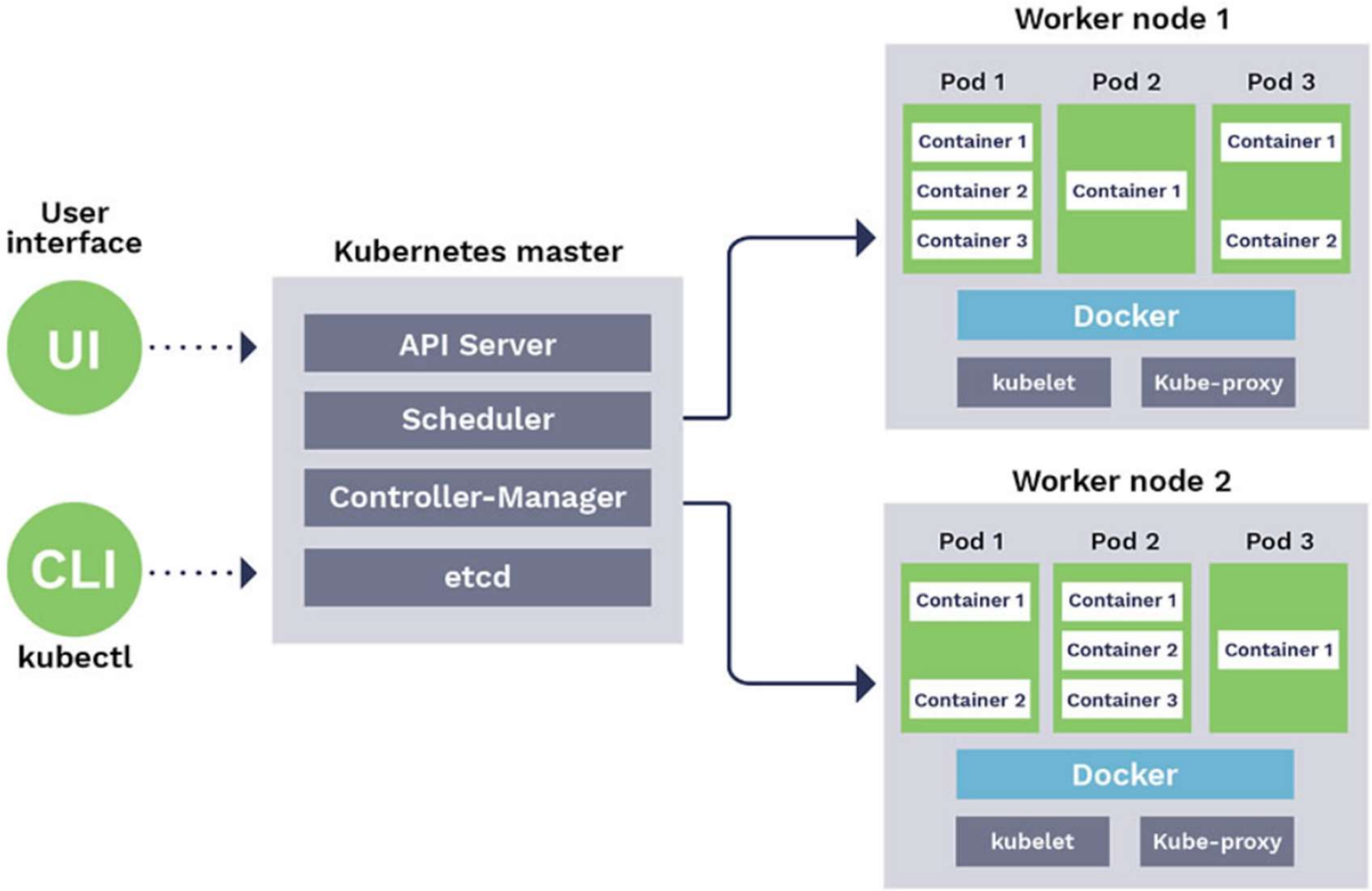
※ 참고 : https://kubernetes.io/

Why Kubernetes … ?



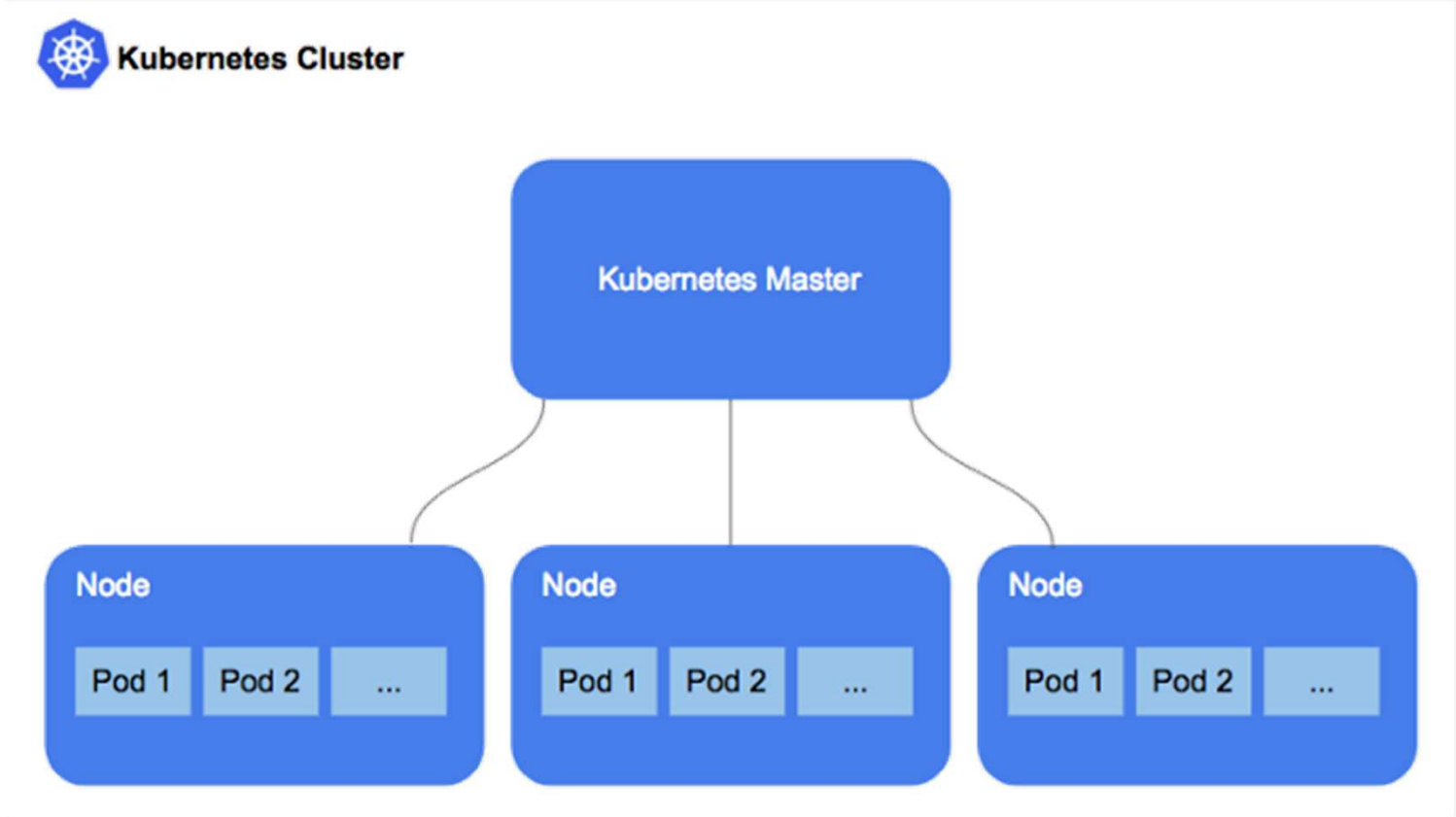
※ 참고 : https://www.docker.com/blog/top-questions-docker-kubernetes-competitors-or-together/

Kubernetes Architecture



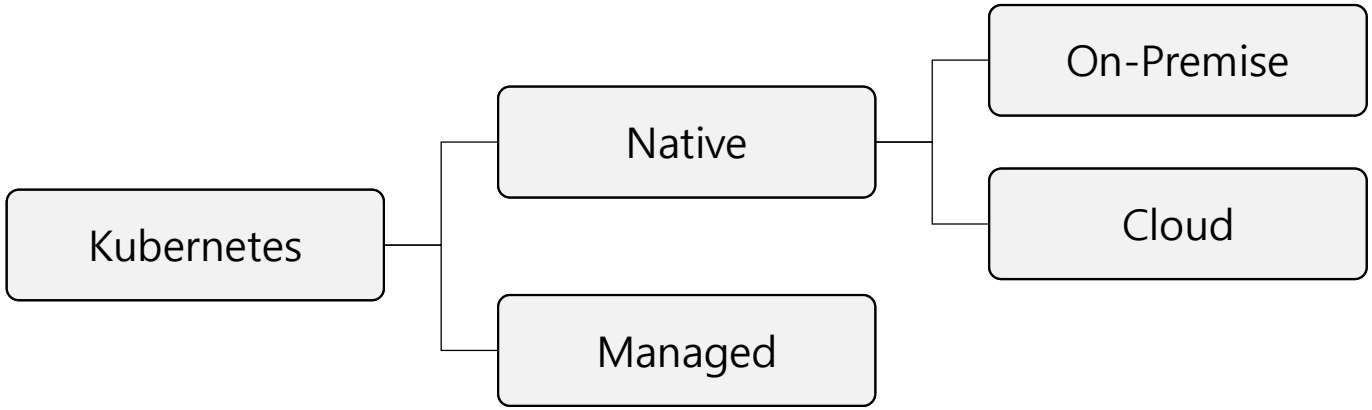
※ 참고 : https://keetmalin.wixsite.com/keetmalin/post/understanding-container-orchestration-with-kubernetes

Cluster



※ 참고 : https://medium.com/@tomerf/so-you-want-to-configure-the-perfect-db-cluster-inside-a-kubernetes-cluster-a4d2c26aca7a

How ...

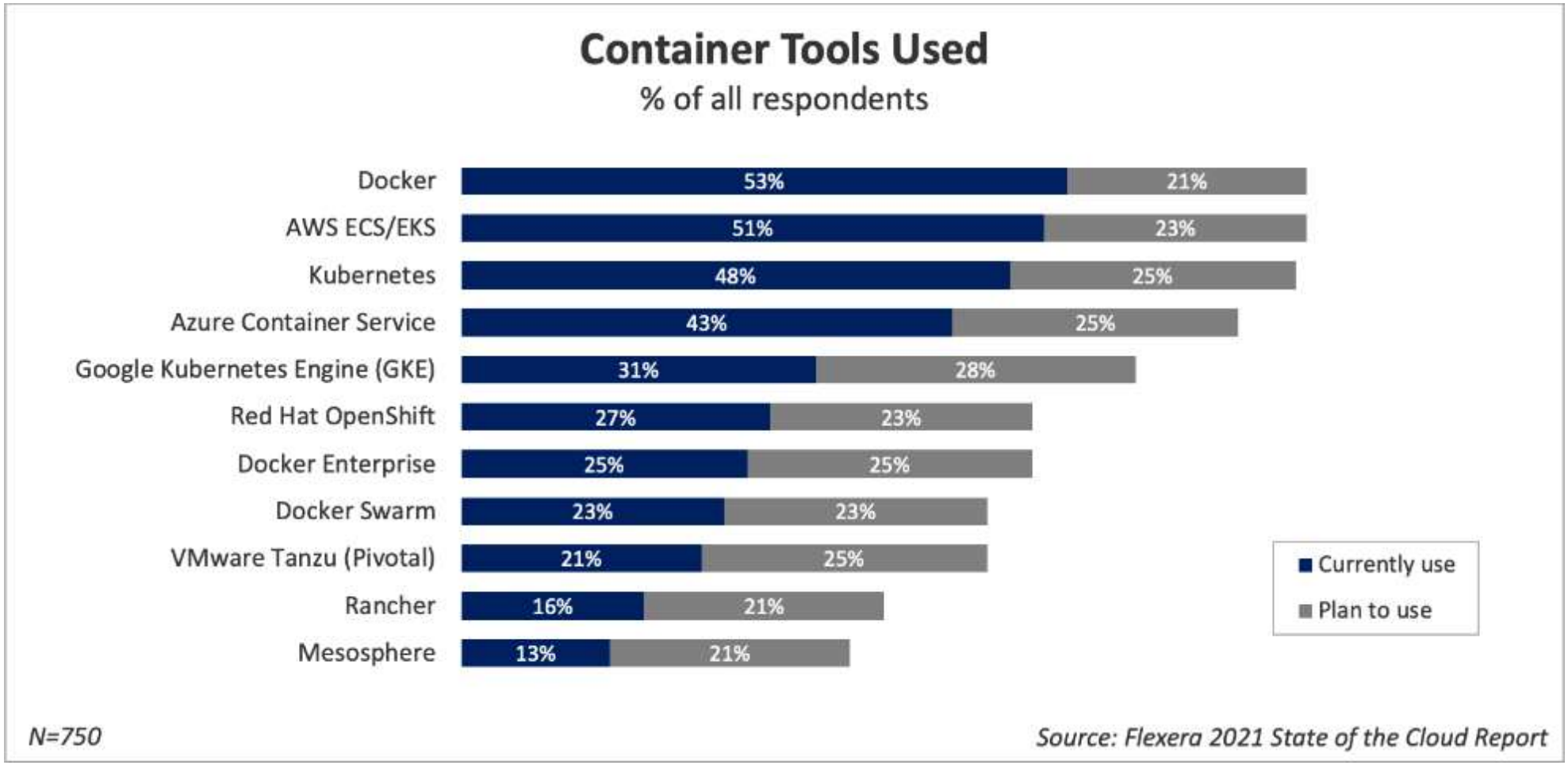


항목	자체 서버환경	Cloud 환경	Managed K8s
인프라 관리	사용자	벤더사	벤더사
K8s 설치, 관리	사용자	사용자	벤더사
K8s(클러스터) 백업	사용자	사용자	벤더사
K8s(클러스터) 스케일링	사용자	사용자	벤더사
워커노드 프로비저닝	사용자	사용자	벤더사
어플리케이션 스케일링	사용자	사용자	사용자
어플리케이션 배포	사용자	사용자	사용자

AWS	EKS (Elastic Kubernetes Service)
Azure	AKS (Azure Kubernetes Service)
GCP	GKE (Google Kubernetes Service)
NCP	NKS (Naver Kubernetes Service)

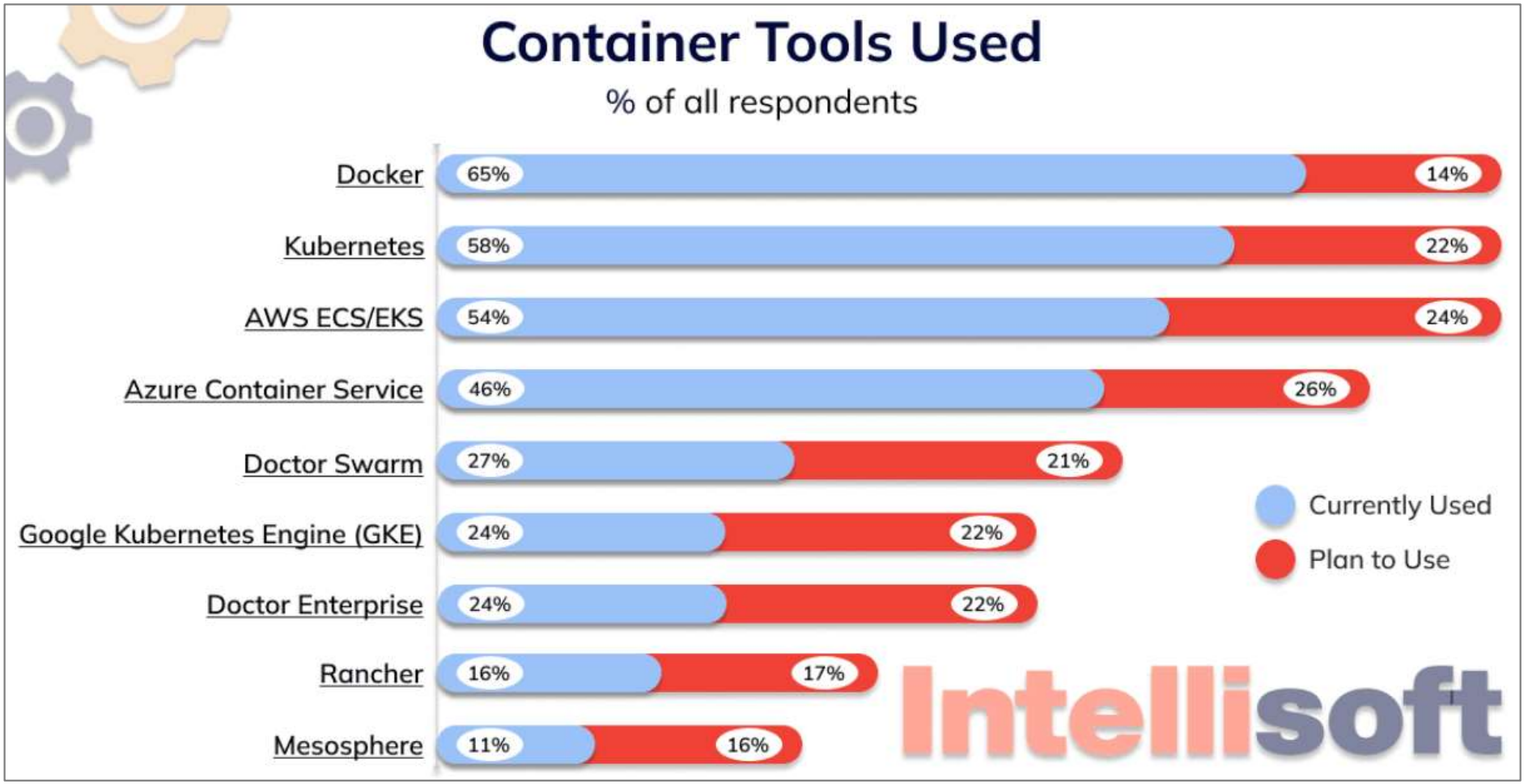
※ 참고 : https://www.cloocus.com/insight-kubernetes_2/

Kubernetes Trends to Watch in 2021



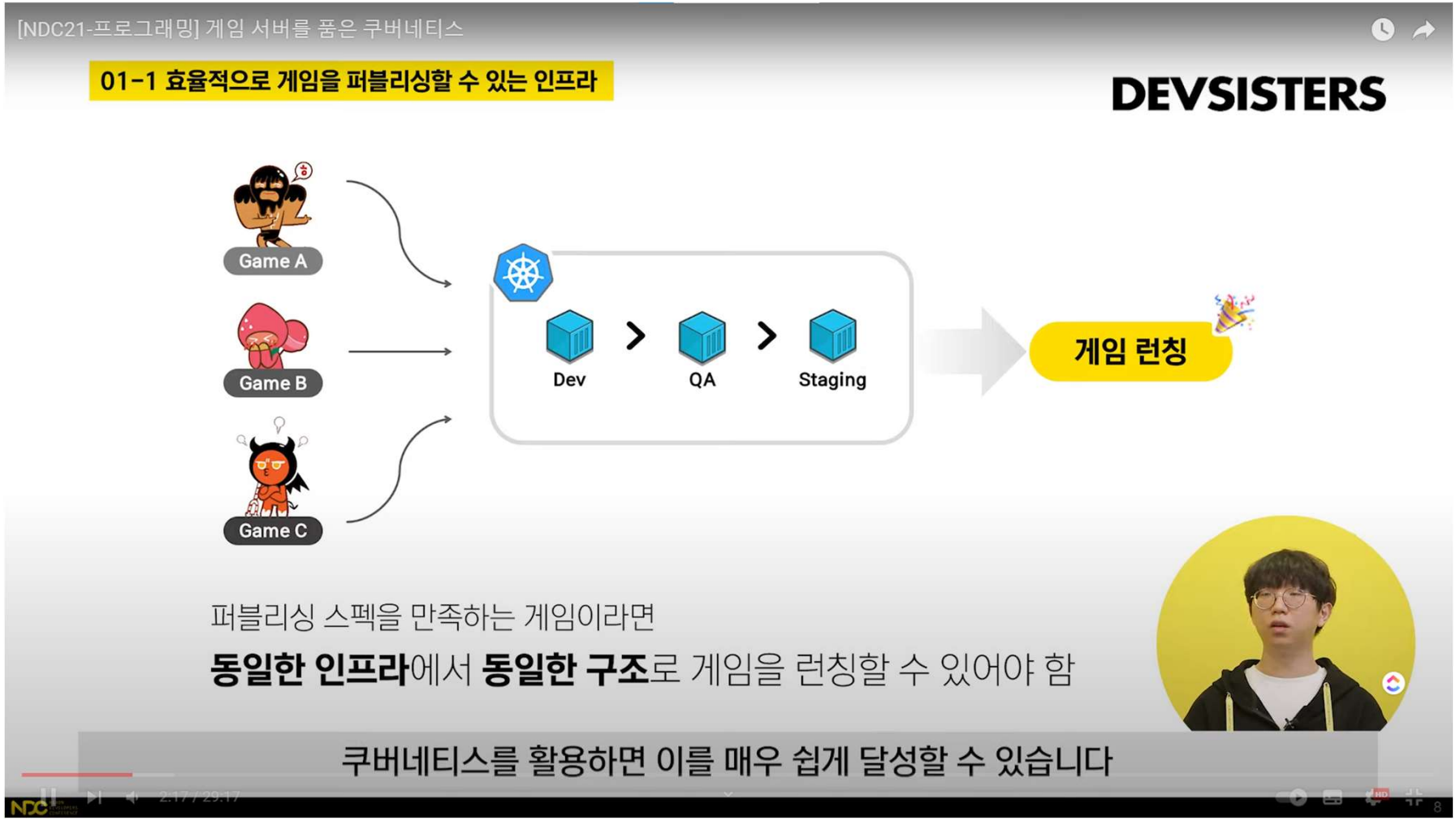
※ 참고 : https://blog.flant.com/kubernetes-and-containers-market-trends-2021/

Kubernetes Trends to Watch in 2023



※ 참고 : https://intellisoft.io/what-is-kubernetes-and-when-to-use-it-key-trends-in-2023/

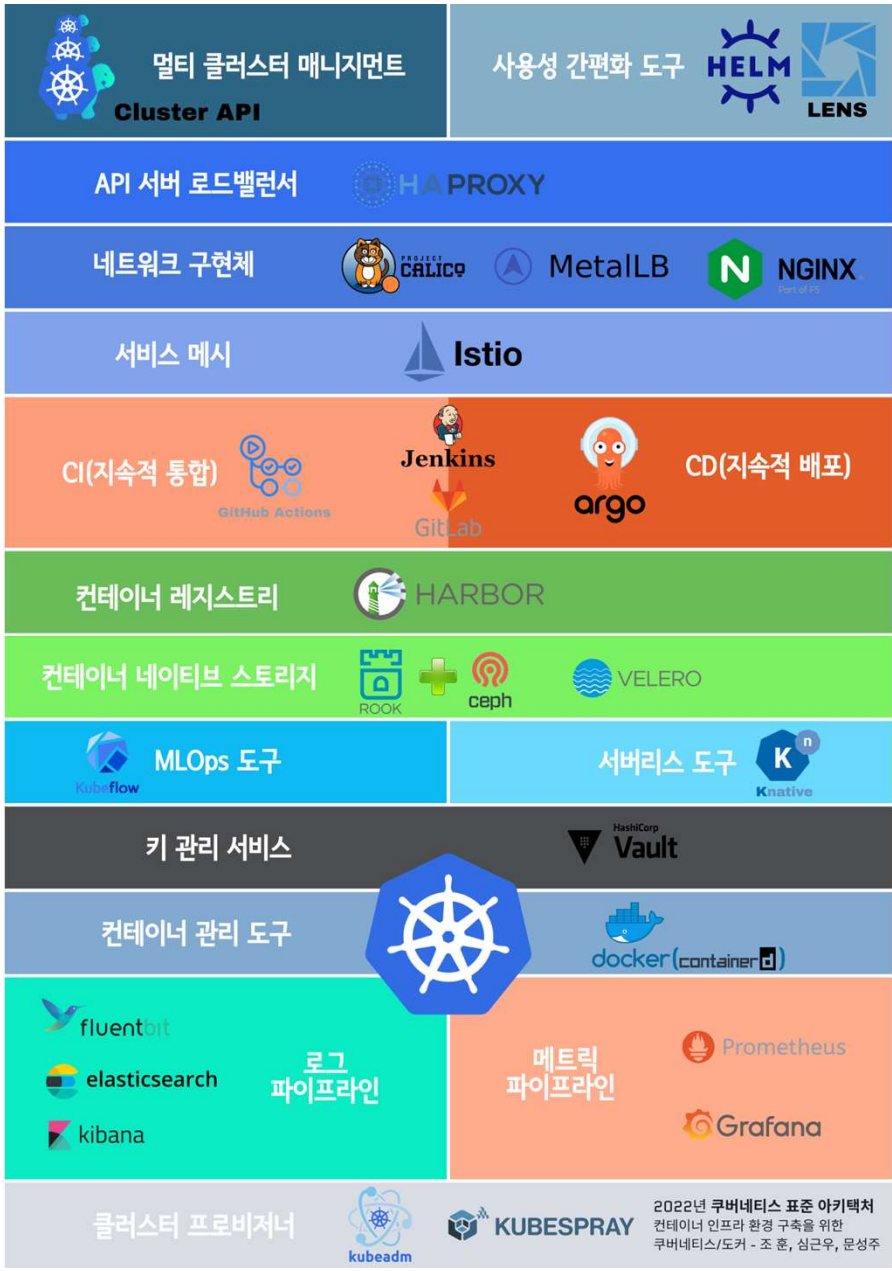
[자습 #1] 게임 서버를 품은 쿠버네티스



※ 참고 : https://www.youtube.com/watch?v=8R4DDEqjc0I

[자습 #2] Kubernetes 표준 아키텍처

- K8s를 안정적으로 구축하기 위해 (공부가) 필요한 도구들

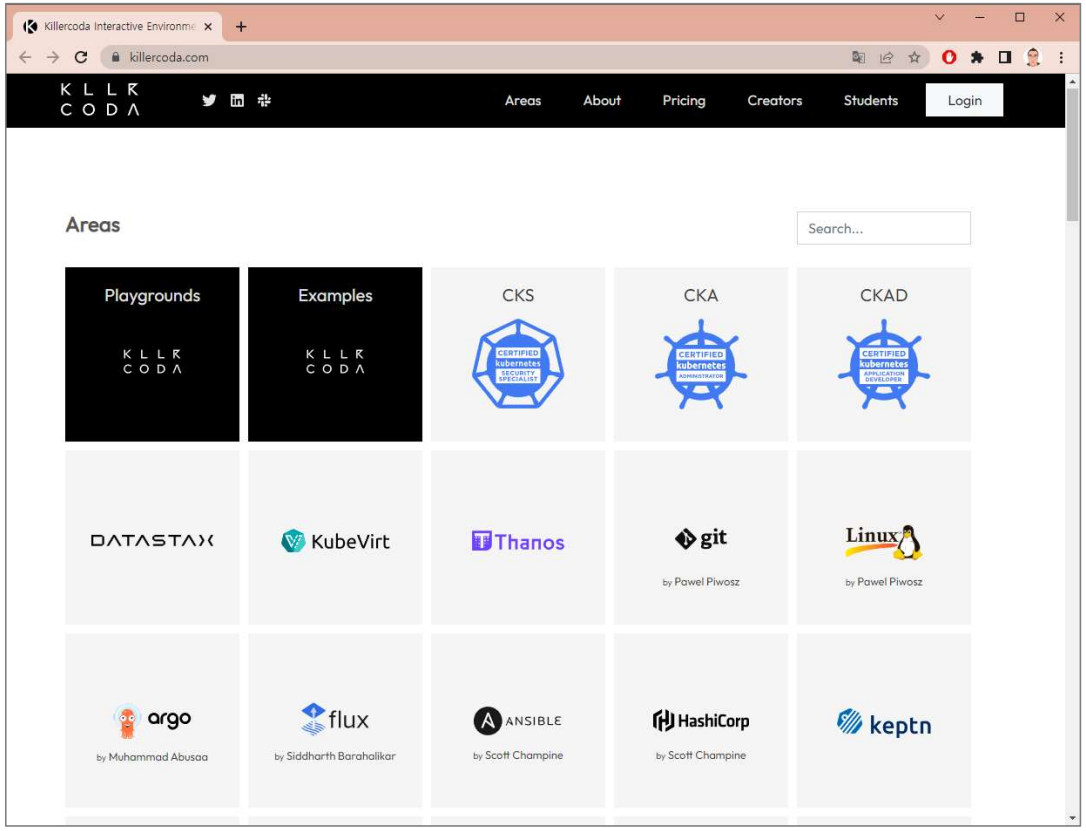


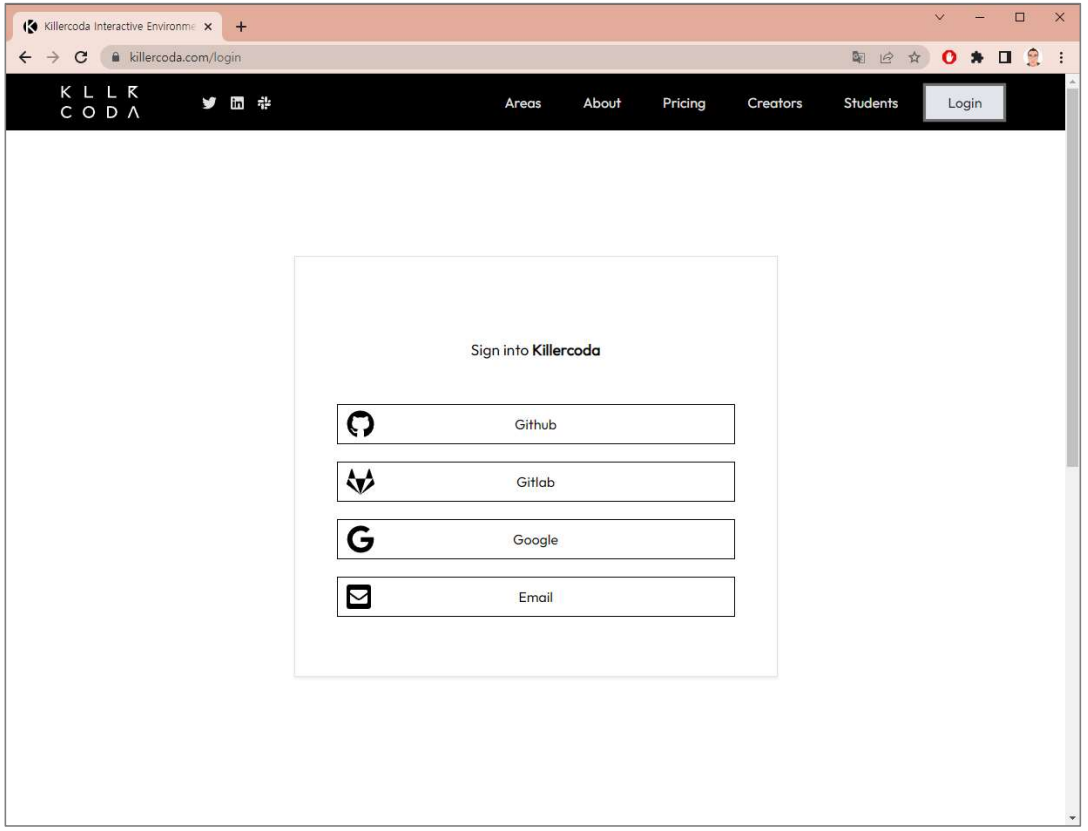
※ 참고 : https://github.com/sysnet4admin/_Book_k8sInfra/tree/main/docs/k8s-stnd-arch/2022

///

실습 환경 셋업

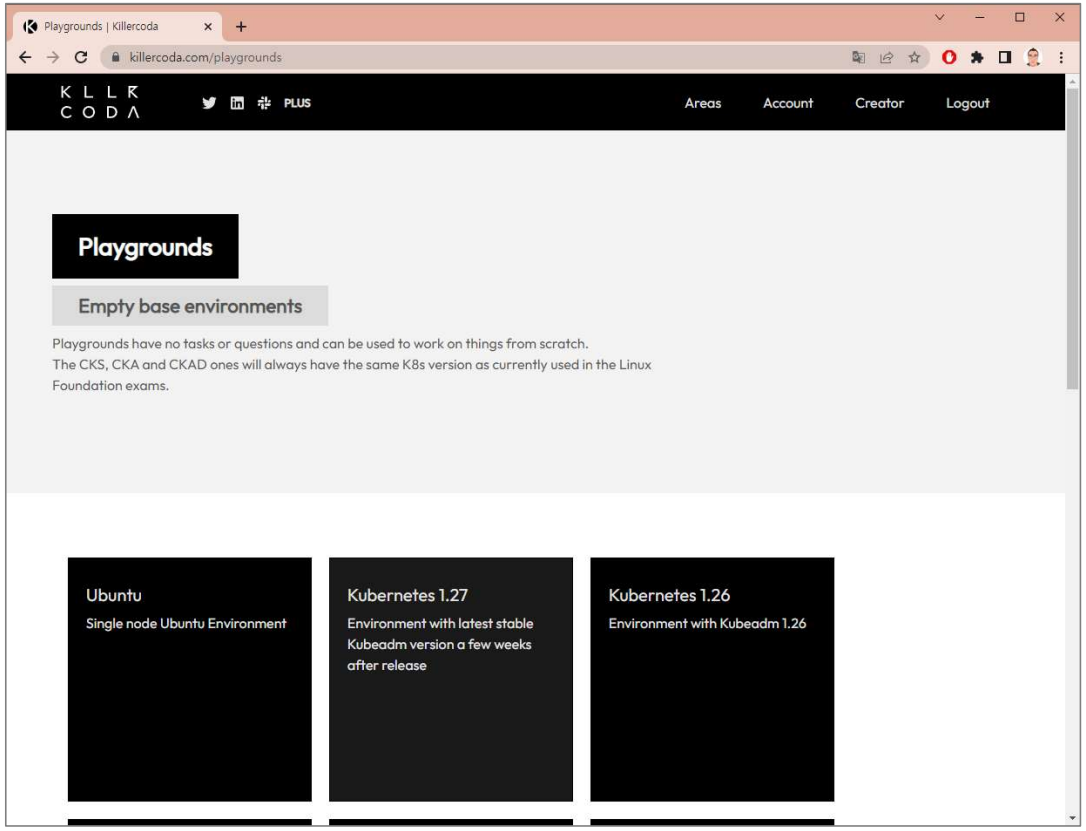
Killercoda

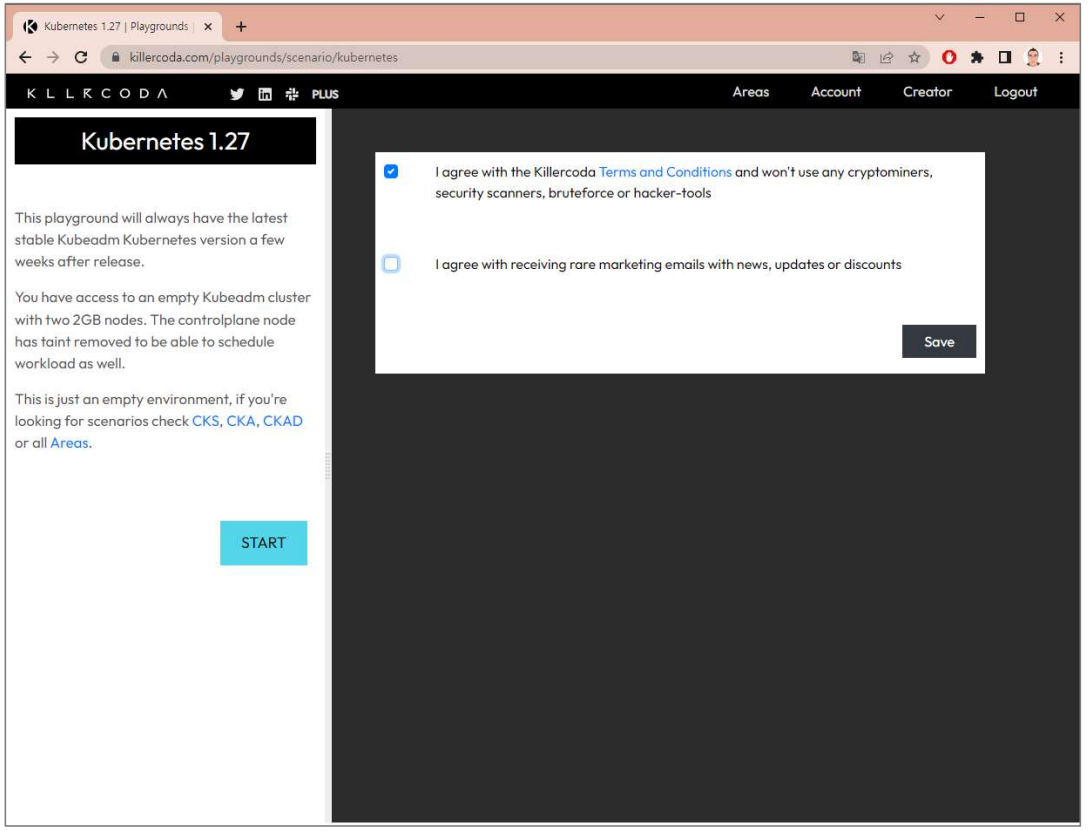




※ 참고 : https://killercoda.com/

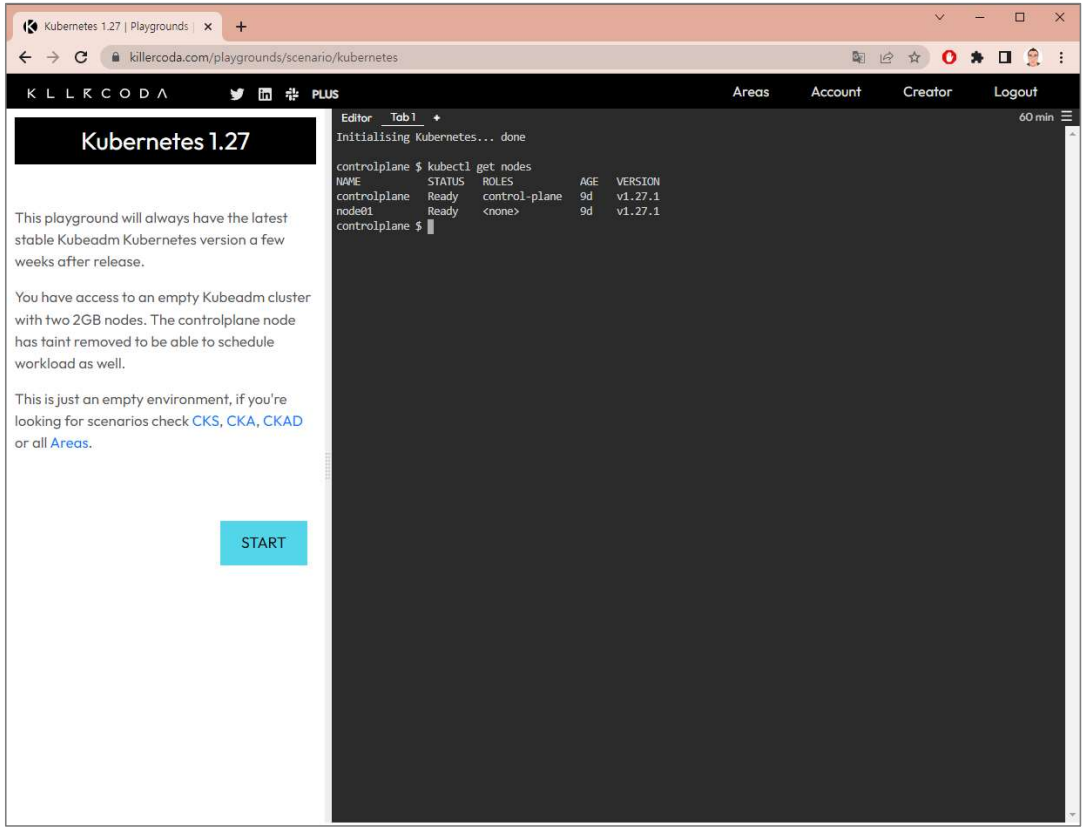
Killercoda

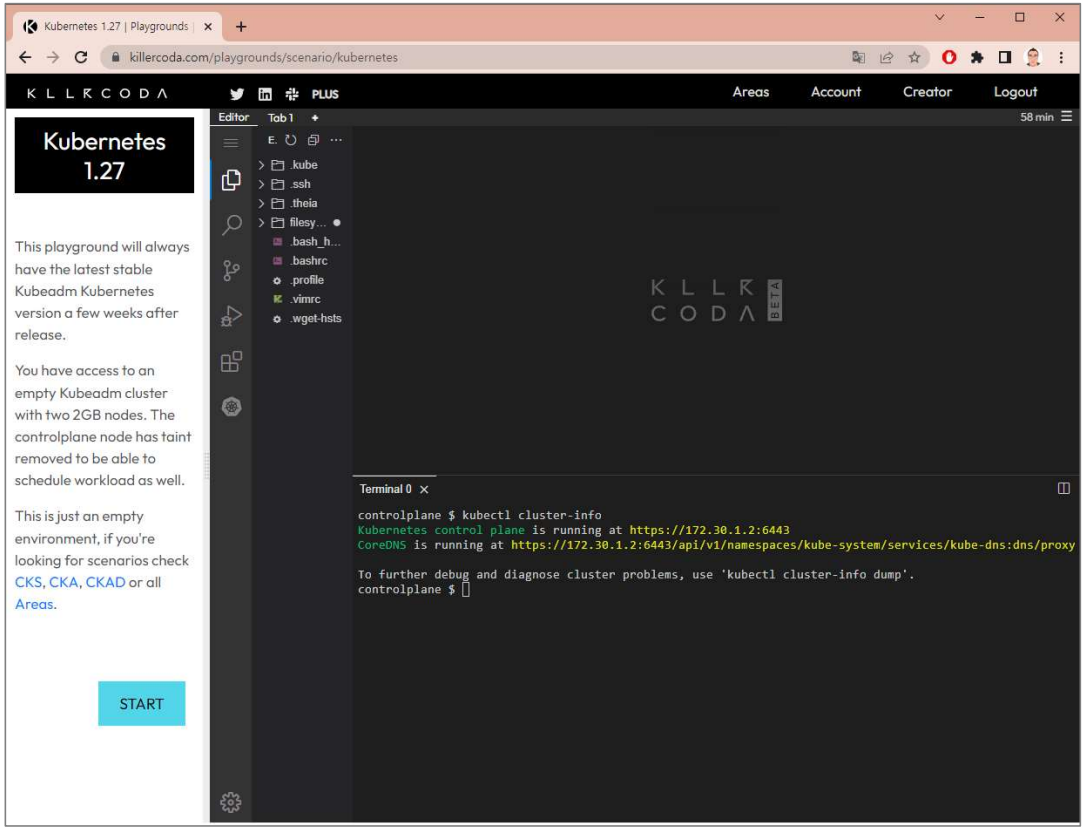




※ 참고 : https://killercoda.com/

Killercoda



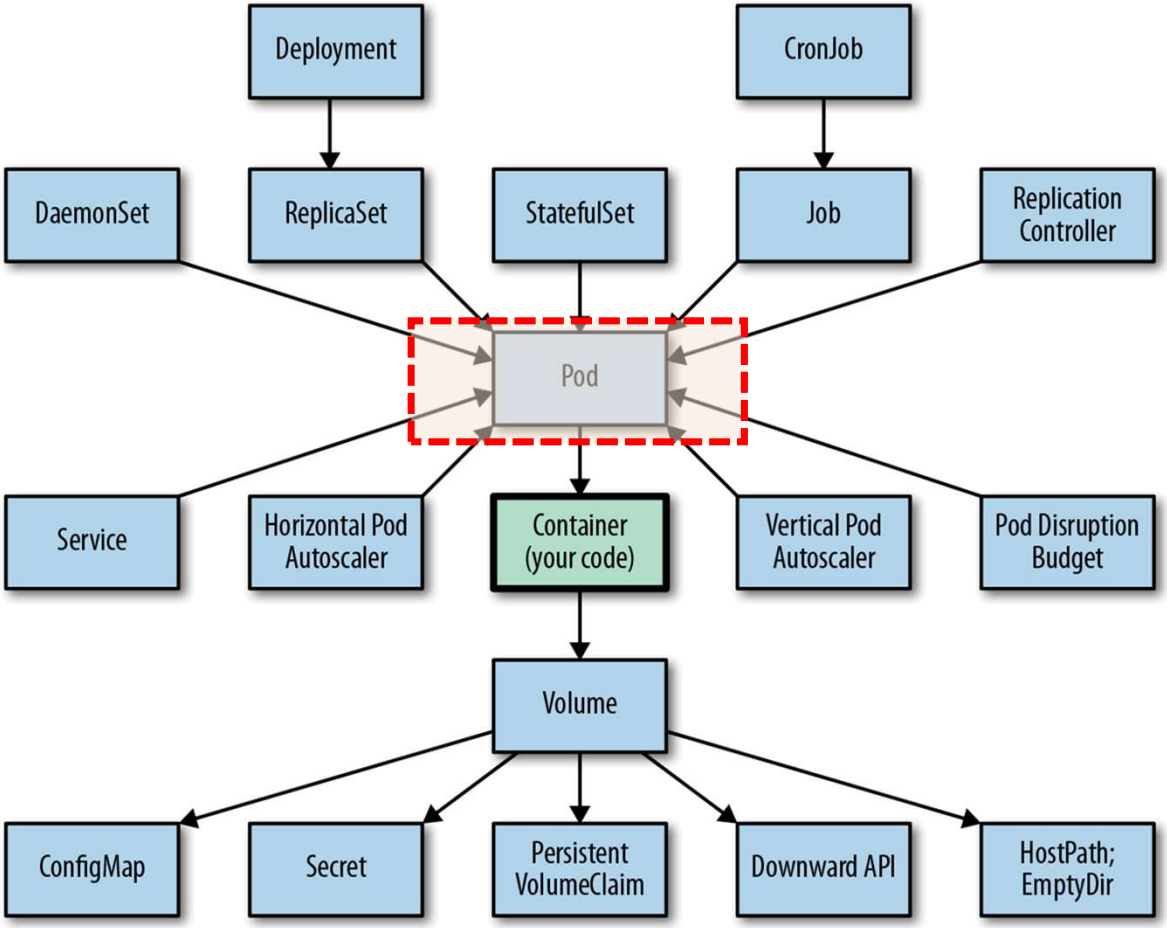


※ 참고 : https://killercoda.com/

///

Pod

Kubernetes concepts for developers



※ 참고 : https://www.oreilly.com/library/view/kubernetes-patterns/9781492050278/ch01.html

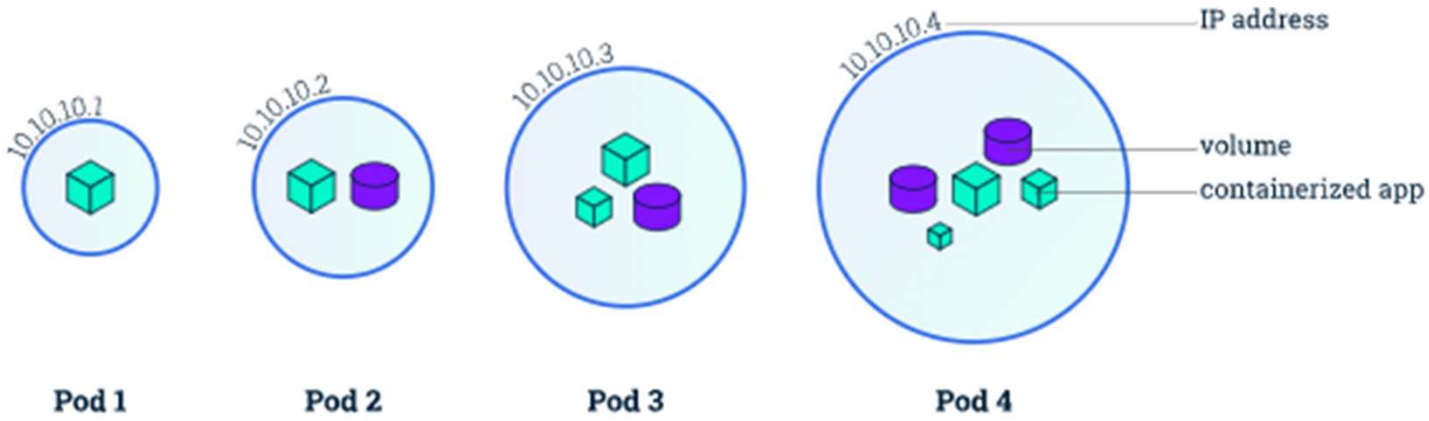
Kubernetes

Pod

Pod is ... #1

Pod는 Kubernetes에서 생성하고 관리할 수 있는 **배포 가능한 가장 작은 컴퓨팅 단위**이다.

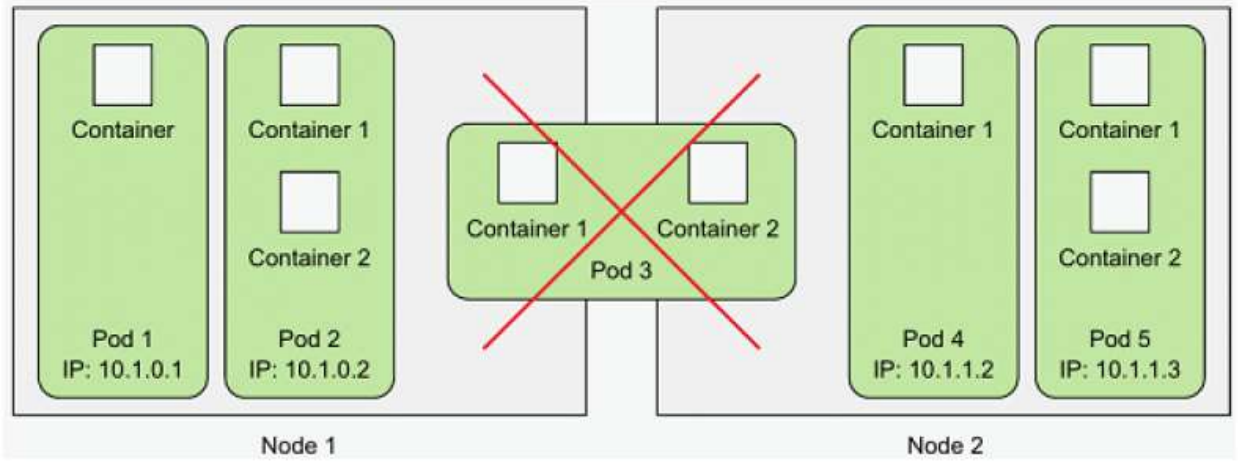
Pod는 **하나 이상의 컨테이너 그룹**이다.



※ 참고 : https://kubernetes.io/ko/docs/tutorials/kubernetes-basics/explore/explore-intro/

Pod is ... #2

- Pod는 함께 배치된 Container 그룹을 의미
- Container는 단일 프로세스를 실행하는 것을 목적으로 설계
- 따라서, 여러 Container를 묶고 하나의 단위로 관리할 수 있는 상위 구조가 필요 → Pod
- Kubernetes는 Pod 단위로 배포하고 운영



▲ 그림 3.1 파드 안에 있는 모든 컨테이너는 같은 노드에서 실행된다. 절대로 두 노드에 걸쳐 배포되지 않는다.

※ 참고 : https://livebook.manning.com/book/kubernetes-in-action/chapter-3/10

Pod YAML

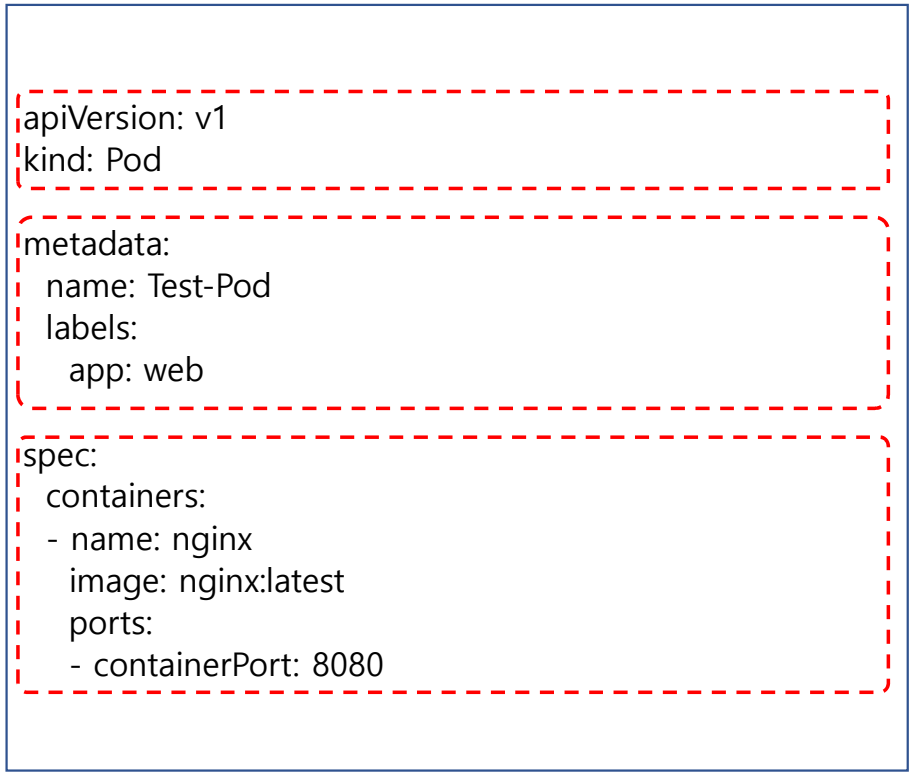
- 기본 요소들

. **apiVersion**: 리소스에 따라 알맞은 API Version 명시

. **kind**: 리소스 종류를 정의

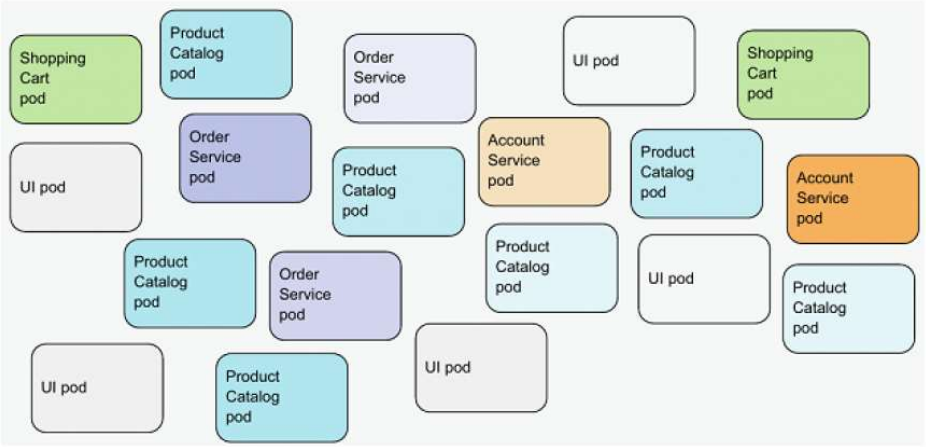
. **metadata**: 리소스의 메타데이터(이름/라벨) 작성

. **spec**: 생성할 리소스 상세 스펙 설정

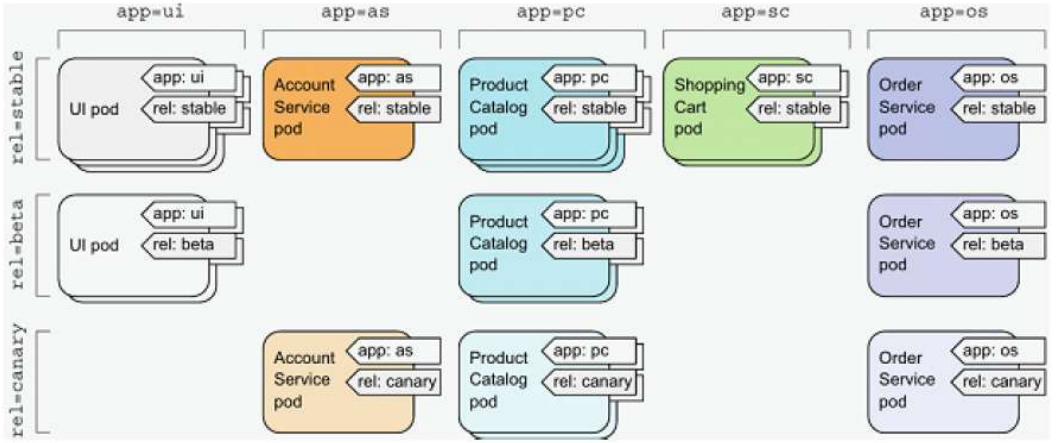


labels

- Label은 Kubernetes 리소스를 분류할 수 있는 기능
- 각 오브젝트는 하나 이상의 레이블을 가질 수 있으며 label은 Key-Value Pair로 이루어짐
- Kubernetes 명령어에서 동일한 label을 가진 오브젝트를 선택할 수 있음



▲ 그림 3.6 마이크로서비스 아키텍처 안에 있는 분류되지 않는 파드



▲ 그림 3.7. 파드 레이블을 이용해 마이크로서비스 아키텍처 안에 파드를 조직화했다.

※ 참고 : https://livebook.manning.com/book/kubernetes-in-action/chapter-3/176

///

K8s : Pod
Hands-On

Scenario

- 다음과 같이 실습을 해보겠다.

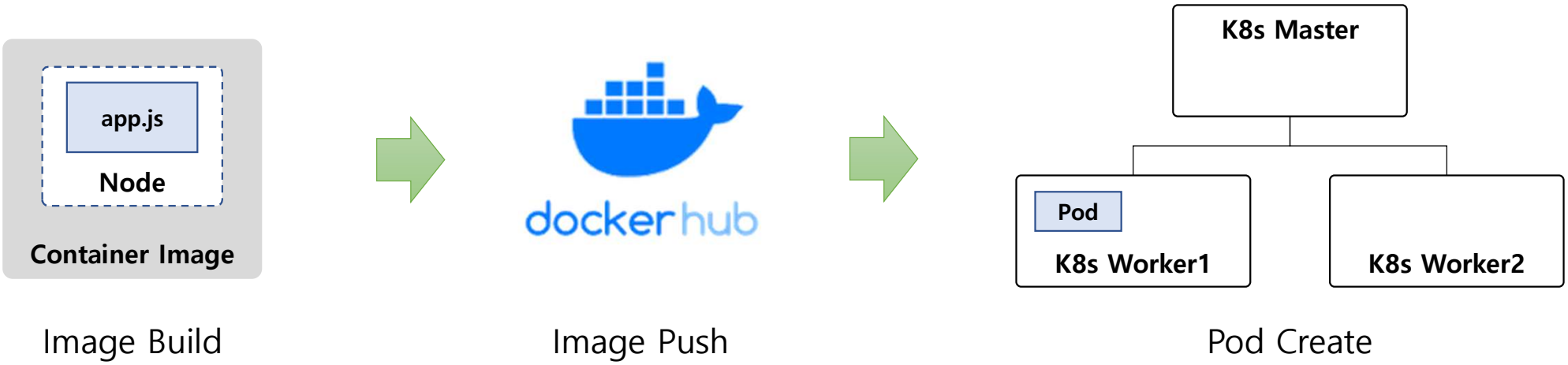


Image Build　　Image Push　　Pod Create

Make image

- node web image를 하나 만들어 보자.

app.js

```
const http = require('http');
const os = require('os');

console.log(“node-web server starting...");

var handler = function(request, response) {
  console.log("Received request from " +
request.connection.remoteAddress);
  response.writeHead(200);
  response.end("You've hit " + os.hostname() + "\n");
};

var www = http.createServer(handler);
www.listen(8080);
```

Dockerfile

```
FROM node:latest

ADD app.js /app.js

ENTRYPOINT ["node", "app.js"]
```

```
remote ❯ git clone https://github.com/whatwant-school/kubernetes.git
remote ❯ cd kubernetes/02_Pod_Namespace/hands-on/

remote ❯ docker buildx build -t node-web:1.0 .

remote ❯ docker tag node-web:1.0 <user-id>/node-web:1.0

remote ❯ docker push <user-id>/node-web:1.0
```

※ 참고 : https://github.com/luksa/kubernetes-in-action/tree/master/Chapter02/kubia

Create pod with YAML

- 앞에서 만든 container를 실행하기 위한 Pod를 생성해보자.

pod-node-web.yaml

```yaml
apiVersion: v1
kind: Pod

metadata:
  name: node-web
  labels:
    creation_method: manual
    env: stage

spec:
  containers:
  - image: whatwant/node-web:1.0
    name: node-web
    ports:
    - containerPort: 8080
      protocol: TCP
```

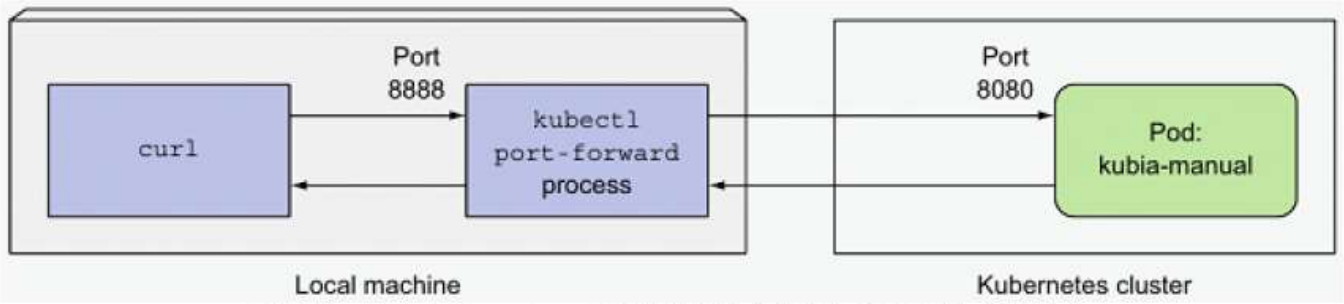
```
remote ❯ git clone https://github.com/whatwant-school/kubernetes.git
remote ❯ cd kubernetes/02_Pod_Namespace/hands-on

remote ❯ kubectl get pods

No resources found in default namespace.


remote ❯ kubectl create -f pod-node-web.yaml

pod/node-web created


remote ❯ kubectl get pods

NAME       READY   STATUS    RESTARTS   AGE
node-web   1/1     Running   0          71s


remote ❯ curl http://localhost:8080

curl: (7) Failed to connect to localhost port 8080 after 0 ms: Connection refused
```

- 웹 연결은 당연히 안된다.

※ 참고 : https://github.com/luksa/kubernetes-in-action/tree/master/Chapter02/kubia

Port forward

- Container에서 실행되고 있는 Web을 연결하기 위해 port-forwarding을 해보자.



▲ 그림 3.5 curl을 kubectl port-forward와 함께 사용할 때 일어나는 일을 간략하게 설명한다.

```
remote ❯ kubectl port-forward node-web 8080:8080 &

[1] 8466
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080


remote ❯ curl http://localhost:8080

Handling connection for 8080
You've hit node-web
```

※ 참고 : https://github.com/luksa/kubernetes-in-action/tree/master/Chapter02/kubia

Delete pod

- port-forward 삭제

```
remote ❯ ps -ef ¦ grep kubectl

chani     8466  3572  0 15:35 pts/1    00:00:00 kubectl port-forward node-web 8080:8080


remote ❯ kill -9 8466

[1]  + 8466 killed     kubectl port-forward node-web 8080:8080
```

- pod 삭제

```
remote ❯ kubectl get pods -o wide

NAME       READY   STATUS    RESTARTS   AGE   IP              NODE      NOMINATED NODE   READINESS GATES
node-web   1/1     Running   0          28m   10.233.103.67   worker2   <none>           <none>


remote ❯ kubectl delete pods node-web

pod "node-web" deleted


remote ❯ kubectl get pods -o wide

No resources found in default namespace.
```

Create pod with CLI

- YAML 파일 없이도 그냥 Pod를 생성할 수 있다.

```
remote ❯ kubectl run node-web-command --image whatwant/node-web:1.0 --port=8080

pod/node-web-command created


remote ❯ kubectl get pods -o wide

NAME               READY   STATUS    RESTARTS   AGE   IP              NODE      NOMINATED NODE   READINESS GATES
node-web-command   1/1     Running   0          4s    10.233.103.68   worker2   <none>           <none>
```

- 웹 연결 및 삭제는 앞에서 진행한 내용과 동일하다.

```
remote ❯ kubectl port-forward node-web-command 8080:8080 &
remote ❯ curl http://localhost:8080

remote ❯ ps -ef ¦ grep kubectl
remote ❯ kill -9 11766

remote ❯ kubectl delete pods node-web-command

pod "node-web-command" deleted
```

///

Kubernetes

Namespace

Namespace is ...

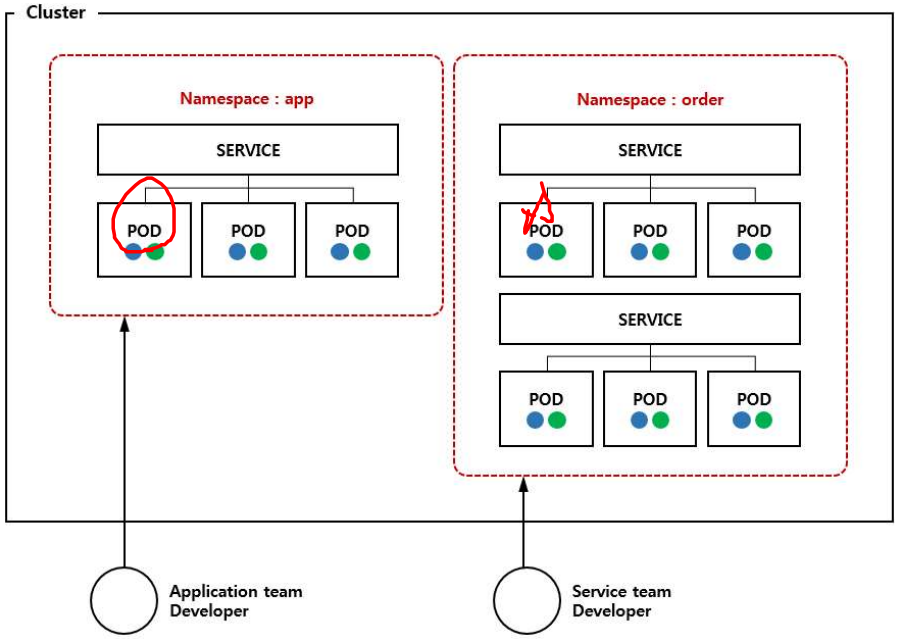
- 네임스페이스는 클러스터 자원을 (리소스 쿼터를 통해) 여러 사용자 사이에서 나누는 방법

  . 여러 개의 팀이나, 프로젝트에 걸쳐서 많은 사용자가 있는 환경에서 사용

  . 네임스페이스는 이름의 범위를 제공

  . 네임스페이스는 서로 중첩될 수 없으며, 각 쿠버네티스 리소스는 하나의 네임스페이스에만 존재

  . Pod, ReplicaSet, Deployment, Service 등을 묶어 놓을 수 있는 하나의 가상 공간 또는 그룹



※ 참고 : https://wiki.webnori.com/display/kubernetes/Namespace

Namespace YAML

- No Comment

```
apiVersion: v1
kind: Namespace

metadata:
  name: whatwant
```

///

K8s : Namespace Hands-On

Namespace

- 이미 많은 namespace가 있다. 확인해보자.

```
remote ❯ kubectl get namespaces

NAME              STATUS   AGE
default           Active   24h
kube-node-lease   Active   24h
kube-public       Active   24h
kube-system       Active   24h
```

- 명시적으로 지정하지 않으면 "default"

```
remote ❯ kubectl get pods --namespace default

No resources found in default namespace.
```

```
remote ❯ kubectl get pods --namespace kube-system

NAME                                       READY   STATUS    RESTARTS       AGE
calico-kube-controllers-6dd874f784-rjxtz   1/1     Running   2 (88m ago)    24h
calico-node-599ck                          1/1     Running   1 (89m ago)    24h
calico-node-qlhvf                          1/1     Running   1 (88m ago)    24h
calico-node-tpwvg                          1/1     Running   1 (89m ago)    24h
coredns-76b4fb4578-kc7vm                   1/1     Running   1 (89m ago)    24h
coredns-76b4fb4578-zbtvb                   1/1     Running   1 (89m ago)    24h
dns-autoscaler-7979fb6659-p5fpm            1/1     Running   1 (89m ago)    24h
kube-apiserver-master                      1/1     Running   2 (89m ago)    24h
kube-controller-manager-master             1/1     Running   2 (89m ago)    24h
kube-proxy-67mzz                           1/1     Running   1 (88m ago)    24h
kube-proxy-hznhg                           1/1     Running   1 (89m ago)    24h
kube-proxy-jhptq                           1/1     Running   1 (89m ago)    24h
kube-scheduler-master                      1/1     Running   2 (89m ago)    24h
nginx-proxy-worker1                        1/1     Running   1 (89m ago)    24h
nginx-proxy-worker2                        1/1     Running   1 (88m ago)    24h
nodelocaldns-7d25c                         1/1     Running   1 (88m ago)    24h
nodelocaldns-ct8zg                         1/1     Running   1 (89m ago)    24h
nodelocaldns-lb5pl                         1/1     Running   3 (89m ago)    24h
```

Create namespace with YAML

- YAML 파일을 이용해서 namespace를 생성해보자

namespace-whatwant.yaml

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: whatwant
```

```
remote ❯ git clone https://github.com/whatwant-school/kubernetes.git
remote ❯ cd kubernetes/02_Pod_Namespace/hands-on

remote ❯ kubectl create -f namespace-whatwant.yaml

namespace/whatwant created


remote ❯ kubectl get namespaces

NAME              STATUS   AGE
default           Active   24h
kube-node-lease   Active   24h
kube-public       Active   24h
kube-system       Active   24h
whatwant          Active   16s
```

Create pod in the namespace

- namespace를 지정해서 pod를 생성할 수 있다.

```
remote ❯ git clone https://github.com/whatwant-school/kubernetes.git
remote ❯ cd kubernetes/02_Pod_Namespace/hands-on

remote ❯ kubectl get pods

No resources found in default namespace.


remote ❯ kubectl create -f pod-node-web.yaml --namespace whatwant

pod/node-web created


remote ❯ kubectl get pods

No resources found in default namespace.


remote ❯ kubectl get pods --namespace whatwant

NAME       READY   STATUS    RESTARTS   AGE   IP             NODE      NOMINATED NODE   READINESS GATES
node-web   1/1     Running   0          25s   10.233.103.3   worker2   <none>           <none>
```

Delete namespace

- pod를 갖고 있는 namespace를 삭제하면 어떻게 될까?

```
remote ❯ kubectl get pods --namespace whatwant -o wide

NAME       READY   STATUS    RESTARTS   AGE   IP             NODE      NOMINATED NODE   READINESS GATES
node-web   1/1     Running   0          58s   10.233.103.4   worker2   <none>           <none>


remote ❯ kubectl delete namespace whatwant

namespace "whatwant" deleted


remote ❯ kubectl get namespaces

NAME              STATUS   AGE
NAME              STATUS   AGE
default           Active   24h
kube-node-lease   Active   24h
kube-public       Active   24h
kube-system       Active   24h
```

- 된다.

Create namespace with CLI

- resource 생성할 때 YAML 이용하는 것이 좋지만, namespace는 CLI로 생성해도 괜찮을 정도로 simple 하다.

```
remote ❯ kubectl create namespace whatwant

namespace/whatwant created


remote ❯ kubectl get namespaces

NAME              STATUS   AGE
NAME              STATUS   AGE
default           Active   24h
kube-node-lease   Active   24h
kube-public       Active   24h
kube-system       Active   24h


remote ❯ kubectl delete namespace whatwant

namespace "whatwant" deleted
```

///

Tip #1

K8s Cheat Sheet

Kubernetes Cheat Sheet

```
> kubectl cluster-info
```

```
> kubectl get nodes
```

```
> kubectl get namespaces
```

```
> kubectl create -f <yaml file>
```

```
> kubectl apply -f <yaml file>
```

```
> kubectl run <name> --image <image>
```

```
> kubectl get pods
```

```
> kubectl logs <pod name>
```

```
> kubectl describe pod <pod name>
```

```
> kubectl delete pod <pod name>
```

```
> kubectl get pods -w
```

```
> kubectl get pods -o wide
```

```
> kubectl get pods -n <namespace>
```

```
> kubectl get pods -l <label>
```