FE 배포환경

Naver 쇼핑 최효석

주제

- 입사 후 배치된 팀의 배포 환경을 어떻게 분석할까?
- 팀에서 신규 배포 환경 구축 시, 어떤 걸 고려해야 할까?

세부 주제

1. Docker, k8s 를 통한 배포
2. Docker , k8s 를 통한 배포 이전 일반적인 배포 방식
3. k8s 의 장점이 어떻게 가능할까?
4. k8s 배포
5. 쇼핑윈도 k8s 환경

1. Docker, k8s 를 통한 배포

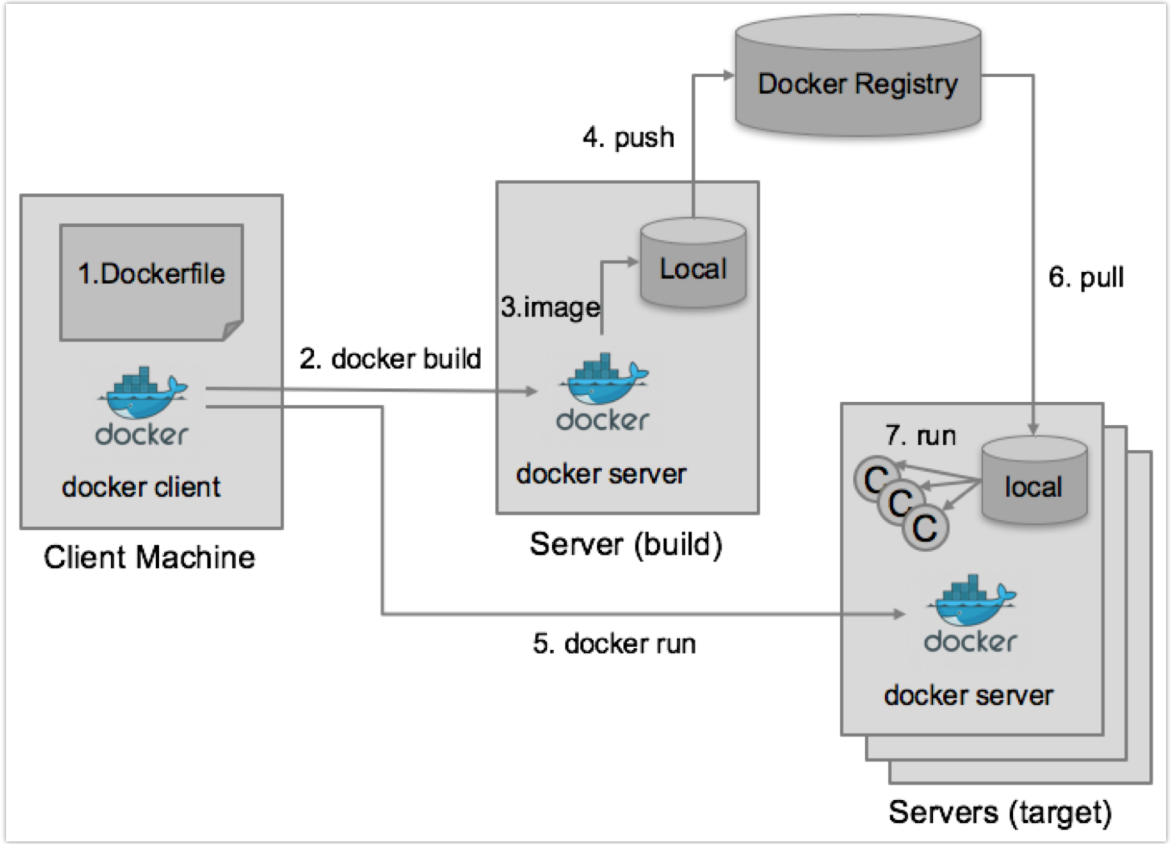
- Docker 를 통한 배포
- kubernetes (k8s) 를 통한 배포

Docker 를 통한 배포

- container image 를 생성하고, 생성한 container image 를 Docker Engine 위에서 실행
- container image 란?
  - OS 를 제외한, application 을 실행하는데 필요한 모든 것을 포함한 실행 가능한 software package
- container 란?
  - image 의 runtime instance
  - image 가 Docker Engine 위에서 실행될 때 image 가 container 로 변환됨

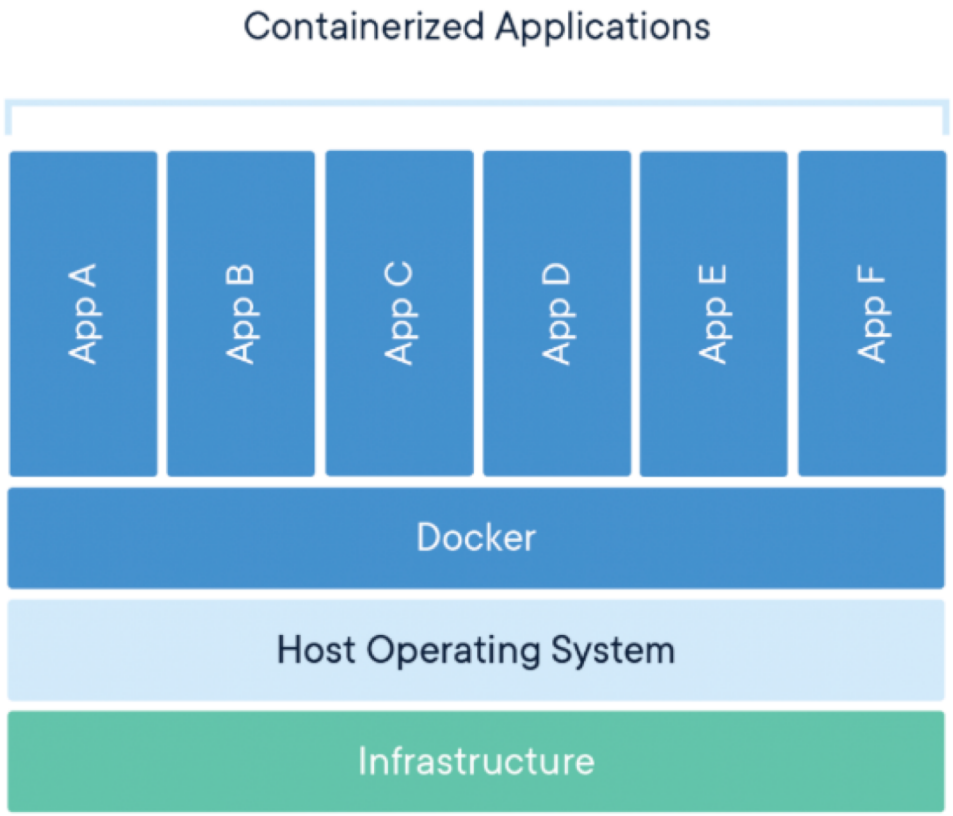
Docker 를 통한 배포

- flow



Docker 를 통한 배포

- Containerized Applications 예시



Docker 를 통한 배포

- 어떤 장점?
  - application 이 deploy 되는 서버의 OS 를 고려하지 않아도 됨
  - 구성 시점이 다른 복수개의 서버에 deploy 시, library 버전이 달라서 발생하는 문제가 없음
    - 동일한 서비스를 serve 하지만 library / OS 버전 등이 다른 서버를 눈송이 서버(snowflake server) 라고 함
    - 눈송이 서버들이 있더라도 container image 내 application 이 사용하는 library 는 container image 내 library 이므로 deploy 되는 서버의 상황에 영향을 받지 않음
  - immutable 한 image 를 통한 배포 => 롤백이 쉬움
- 참고
  - https://www.docker.com/resources/what-container
  - https://www.44bits.io/ko/post/why-should-i-use-docker-container

kubernetes (k8s) 를 통한 배포

- container orchestration
  - 여러 host 에서 container 를 운영/관리하게 해주는 기술
- 구글의 container orchestration

  → 현재 container orchestration 의 최강자로 AWS, GCP, Azure, Naver cloud 에서 서비스 제공

kubernetes (k8s) 를 통한 배포

- 어떤 장점?
  - Deploy Containerized Applications To a Cluster
    - 개별 machine 에 deploy 하지 않음
    - 이것이 가능하려면, application 이 개별 host 와 분리되어 package 되어 있어야 함
    - 참고) cluster 란?
      - cluster of computers that are connected to work as a single unit
  - Infrastructure and Configuration as Code
  - On-Demand Infrastructure
    - scaling
    - auto scaling
      - HPA (Horizontal Pod Autoscaler)
  - Zero-Downtime Deployments (rolling update)
  - Self-Healing
- 참고
  - https://containerjournal.com/editorial-calendar/kubernetes-in-the-enterprise/devops-and-kubernetes-a-perfect-match/

2. Docker, k8s 를 통한 배포 이전 일반적인 배포 방식

- 개별 server 에 application 설치
- k8s 의 장점이 가능한가? 어려움

Docker, k8s 를 통한 배포 이전 일반적인 배포 방식

- 개별 server 에 application 설치
  - 예시 flow
    - L4 에서 해당 서버 제외
    - application 중단
    - 소스 복사
    - application 실행
    - L4 에서 해당 서버 추가

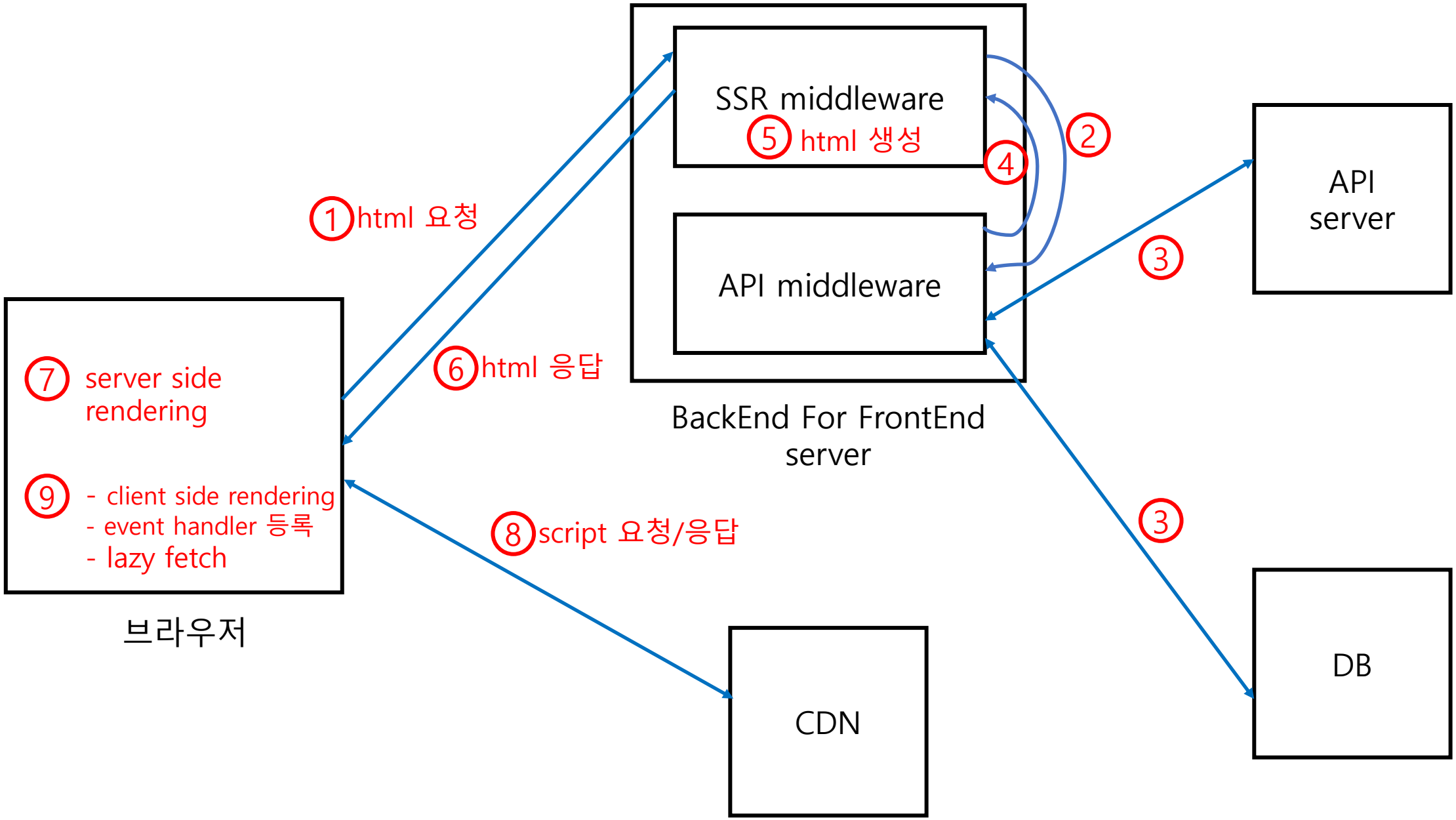
Docker, k8s 를 통한 배포 이전 일반적인 배포 방식

- k8s 의 장점이 가능한가? 어려움
  - Deploy Containerized Applications To a Cluster
    - 불가
    - 개별 server 에 application 설치
  - Infrastructure and Configuration as Code
    - 가능은 하지만, 불편
      - ansible 등을 통해 Infrastructure as Code 관리 가능
      - configuration 및 script 를 git 에서 관리 가능
  - On-Demand Infrastructure
    - 불가
    - 요청량이 늘어 서버를 늘리려면?
      - 서버 구매 -> OS 설치 -> java or node 등 관련 library 설치 -> ...
    - 요청량이 줄어 서버를 줄이려면?
      - 중고 판매?
      - 다른 서비스에서 사용하려면?
        - 기존 library 삭제 -> 관련 library 설치 -> ...

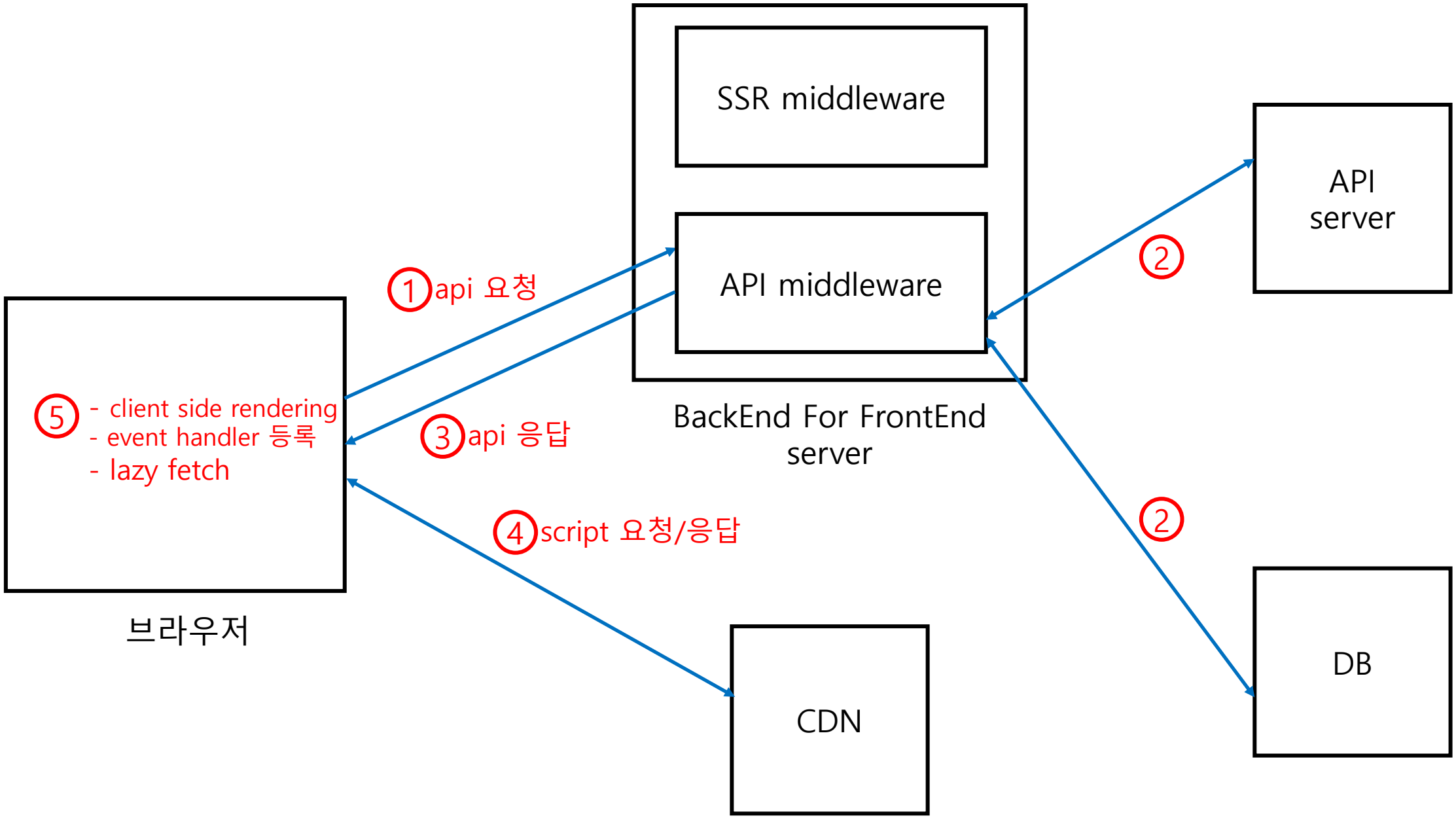
Docker, k8s 를 통한 배포 이전 일반적인 배포 방식

- k8s 의 장점이 가능한가? 어려움
  - Zero-Downtime Deployments (rolling update)
    - downtime 은 없으나, 아래와 같은 상황이 발생할 수 있음
    - 상황 예시
      - blue-green 배포가 까다로움 => 변경 내역을 순차적으로 반영
        - blue-green 배포?
          - 기존 코드가 배포된 서버들(blue) 이 serve 되는 상태에서 신규 코드가 배포된 서버들(green)을 띄움
          - 특정 시점에 serve 하는 server 들을 blue 에서 green 으로 한꺼번에 변경
      - 변경 내역을 순차적으로 반영 시 문제 상황 예시
        - 특정 api versioning 을 했고, 기존 코드로 동작하는 client는 '/v1' api, 변경 코드로 동작하는 client는 '/v2' api 를 호출
          - api versioning ?
            - 복수개 version (ex. v1, v2) 의 api 를 동시에 serve
      - client가 변경 코드로 동작하는 서버로부터 html, script 를 받은 후, '/v2' api 를 기존 코드로 동작하는 서버로 요청 시, 404 에러 발생 => 페이지 에러 가능
      - blue-green 배포됐다면, 위와 같은 상황은 없음
  - Self-Healing
    - 불가

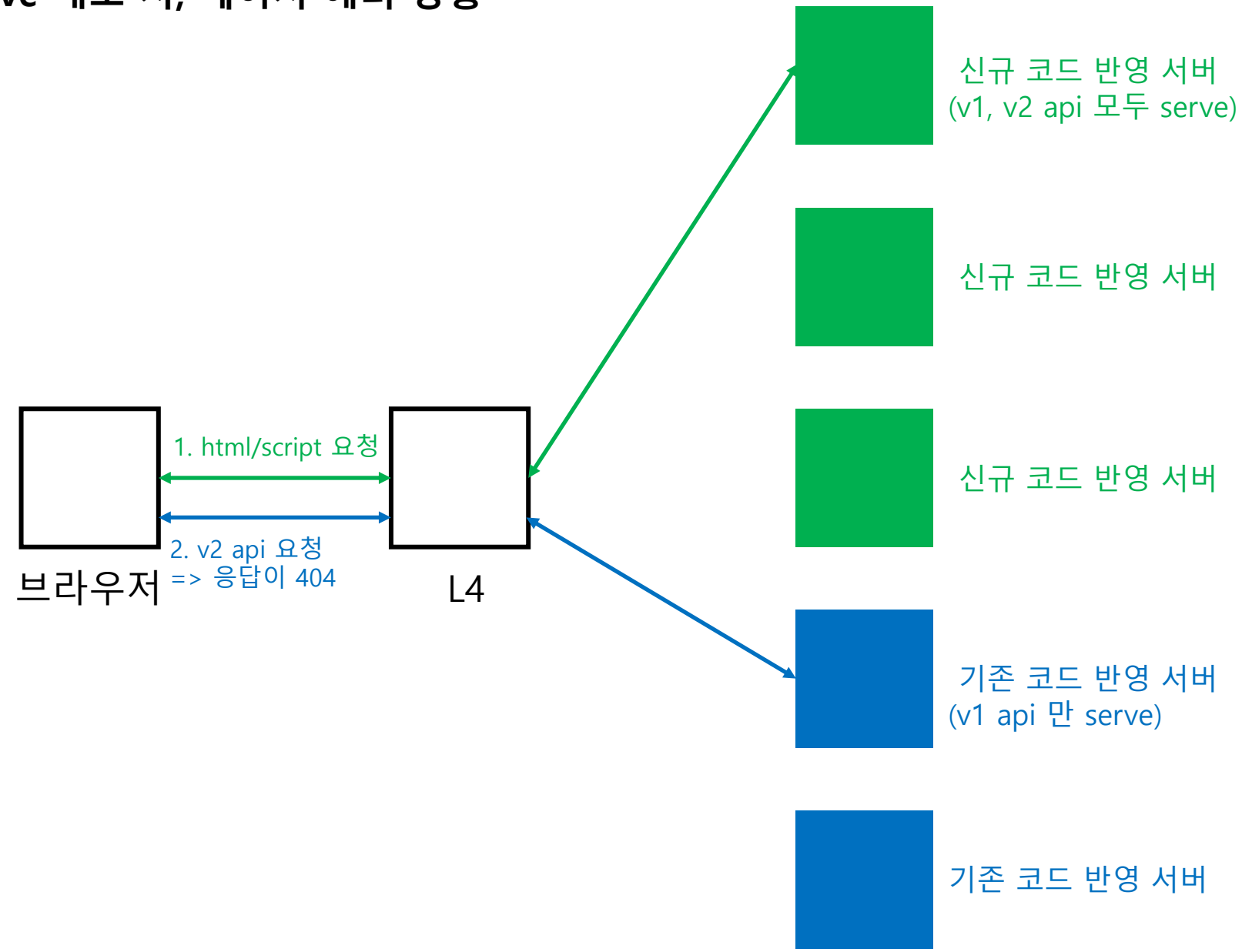
최초 페이지 요청 / server side navigation



client side navigation



progressive 배포 시, 페이지 에러 상황



3. k8s 의 장점이 어떻게 가능할까?

- container orchestration
- container orchestration 상세

k8s 의 장점이 어떻게 가능할까?

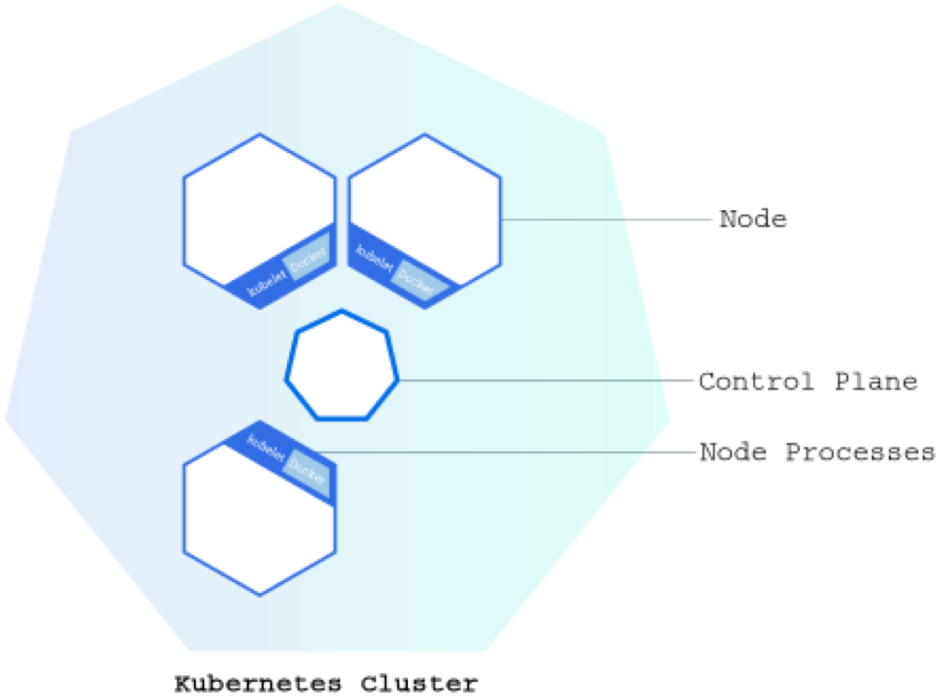
- container orchestration
  - sheduling
    - 여러 host 에 container 분배. host 장애 시 재분배
  - networking
    - container 간 통신, L4/L7
  - logging
    - mutable 한 container 들의 로그 조회
  - monitoring
    - mutable 한 container 들 모니터링
  - storage
    - remote storage
    - blocked / shared

k8s 의 장점이 어떻게 가능할까?

- container orchestration 상세

https://kubernetes.io/docs/tutorials/kubernetes-basics/

1. Create a Cluster
    - cluster ?
        - a single unit 으로 동작하는 복수개의 machine
    - k8s cluster 구성
        - control plane (master)
        - node
    - 복수개의 physical 또는 virtual machine 에 k8s cluster 를 구성



k8s 의 장점이 어떻게 가능할까?

- container orchestration 상세

2. Deploy an App

  - Pod
    - k8s 의 atomic unit
    - 여러개의 container 묶음
      - ex) node + nginx + logstash
    - pod 에 묶여 있는 container 들은 network 와 volume 공유
  - ReplicaSet
    - 몇개의 pod instances 를 만들까?
    - 어떤 pod 들에 대해 ReplicaSet 을 설정할건지 설정 필요
  - Deployment
    - Deployment object 생성 -> Deployment Controller 가 ReplicaSet 을 생성 -> pod 생성

k8s 의 장점이 어떻게 가능할까?

- container orchestration 상세

2. Deploy an App

    - Deployment Controller
        - 아래 사항들을 주기적으로 체크
        - pod 이 죽었다면, 해당 pod 을 종료시키고 새로운 pod 를 시작시킴
            - container 의 livenessProbe 로 정의
                - liveness 는 어떻게 알 수 있는지?
                    - command 실행 / HTTP request / TCP request
                - initialDelaySeconds, periodSeconds, failureThreshold
        - pod 이 아직 준비가 안됐다면, 서비스에서 자동으로 제외시킴. 준비가 됐다면 서비스에 포함시킴
            - container 의 readinessProbe 로 정의
                - liveness 는 어떻게 알 수 있는지?
                    - command 실행 / HTTP request / TCP request
                - initialDelaySeconds, periodSeconds, failureThreshold

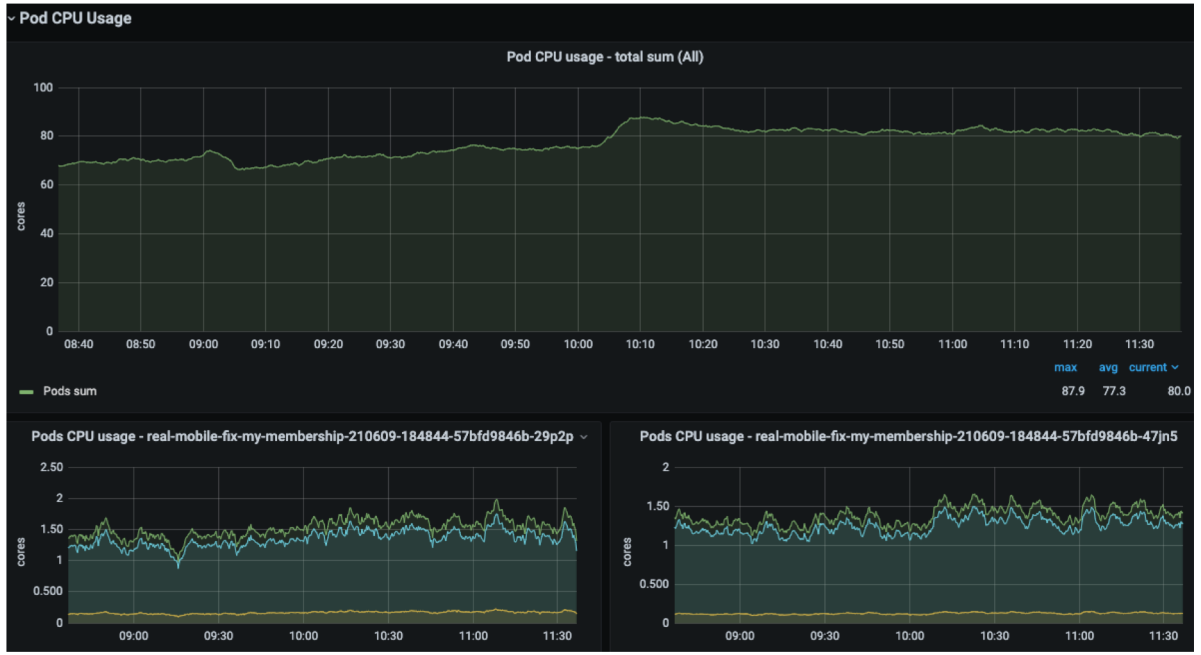
k8s 의 장점이 어떻게 가능할까?

- container orchestration 상세

3. Expore an App

  - Pod, Deployment, Service 상태에 대해 탐색 가능
  - 모니터링 가능

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
beta-mobile-release-gift-0610-210609-201001	1/1	1	1	15h
beta-pc-feature-plan-event-210609--210609-193550	1/1	1	1	16h
real-mobile-fix-my-membership-210609-184844	50/50	50	50	16h
real-pc-master-210608-172825	25/25	25	25	42h
shopping-static-resources-default	5/5	5	5	342d
stage-mobile-fix-my-membership-210609-183923	1/1	1	1	16h
stage-pc-master-210608-155744	1/1	1	1	43h

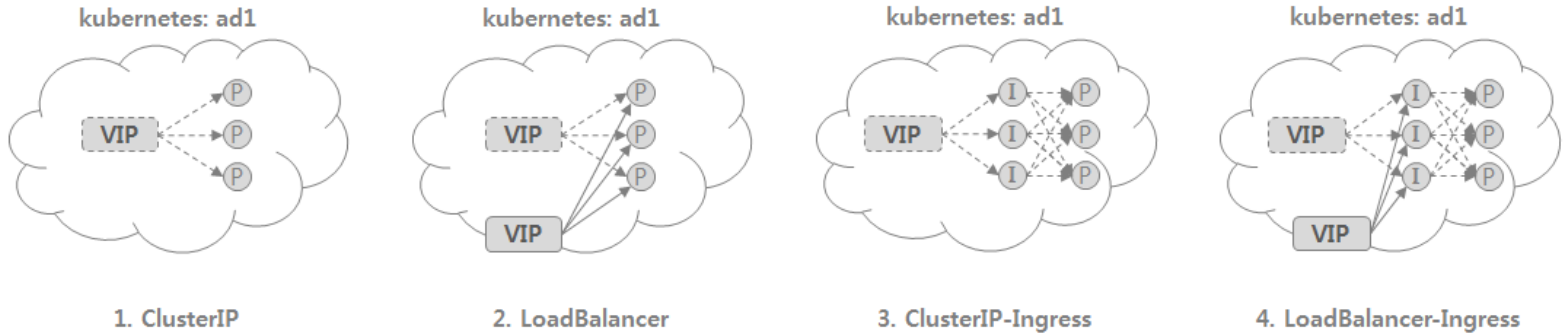


k8s 의 장점이 어떻게 가능할까?

- container orchestration 상세

4. Expose Your App Publicly

- Service
  - Pod 에 request 를 하려면 어떻게 해야 할까?
    - Pod 에 바로 request 하는 것은 어렵다.
      - Pod 은 고유의 ip 가 부여되지만, 언제든 삭제되고, 새로운 ip 가 부여된 pod 이 생길 수 있음 (Pods are mortal)
        => client 에서 Pod 에 ip 로 request 하는 것이 어렵고, L4 에서 Pod 에 load balance 하기도 어려움
    - Service 를 통해 가능
      - Service 란?
        - a logical set of Pods 과 Pods 에 접근할 수 있는 정책을 정의
        - cluster 외부 또는 cluster 내부에서 접근할 수 있는 ip 가 부여되고, request 들을 pod 들에 load-balancing 함
      - Service 의 type 별 상세



k8s 의 장점이 어떻게 가능할까?

- container orchestration 상세

5. Scale Your App
    - 요청 시 또는 자동으로 Pod instance 개수를 늘릴 수 있음
    - Deployment, ReplicaSet, HPA

6. Update Your App
    - downtime 없이 배포 가능
    - blue-green 배포를 쉽게 적용 가능

4. k8s 배포

- helm chart
- 배포 모드

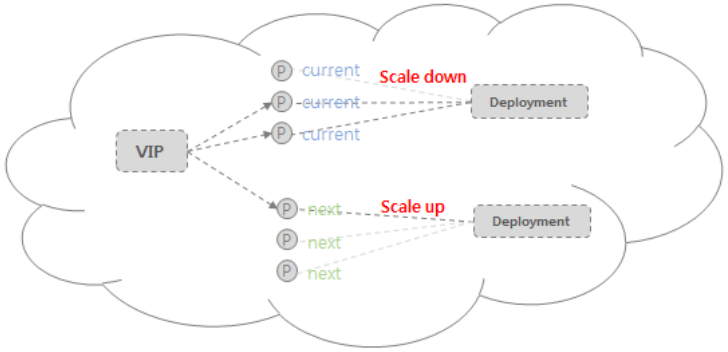
k8s 배포

- helm chart
  - helm 은 k8s 의 package manager
    - k8s 상에 배포되는 소프트웨어를 쉽게 배포/update/삭제할 수 있도록 해줌
  - helm 은 하나의 배포 프로젝트를 chart 라는 단위로 관리
    - Chart.yaml
      - chart 의 name, version 등 각종 metadata
    - values.yaml
      - chart 내에서 사용하는 각종 변수들의 기본값
    - templates/
      - chart 가 k8s cluster 에 설치할 리소스 template
      - _helpers.tpl
        - 여러 template 에서 공통으로 사용되는 template
  - 자세한 사항은 아래 url 참고
    - https://docs.helm.sh/docs/chart_template_guide/getting_started/
    - https://v2.helm.sh/docs/developing_charts/
    - https://v2.helm.sh/docs/chart_template_guide/

k8s 배포

- 배포 모드
  - Rolling Update
    - Deployment 에 설정된 image 를 사용자가 변경
      - Deployment controller가 자동으로 replicationset을 하나 더 추가하여 pod들을 차례로 교체

  - Progressive 배포
    - 새로운 버전의 서비스가 잘 동작하는지 점진적으로 확인하면서 반영 (운영자가 조절)
      - 기존 버전의 deployment와는 별개로 새로운 버전의 image를 갖는 deployment 추가
      - 새로 추가된 deployment의 pod label을 서비스(VIP)의 selector label에 지정된 값으로 설정
        => 새로운 deployment를 통해 추가된 pod들도 모두 서비스에 포함되어 운영
      - 새 버전 pod들이 문제없이 동작하는 것이 확인이 되면, 점진적으로 pod개수를 늘리고(scale up), 이와 동시에 기존 버전의 pod들은 줄여나감 (scale down)



Progressive Deploy

k8s 배포

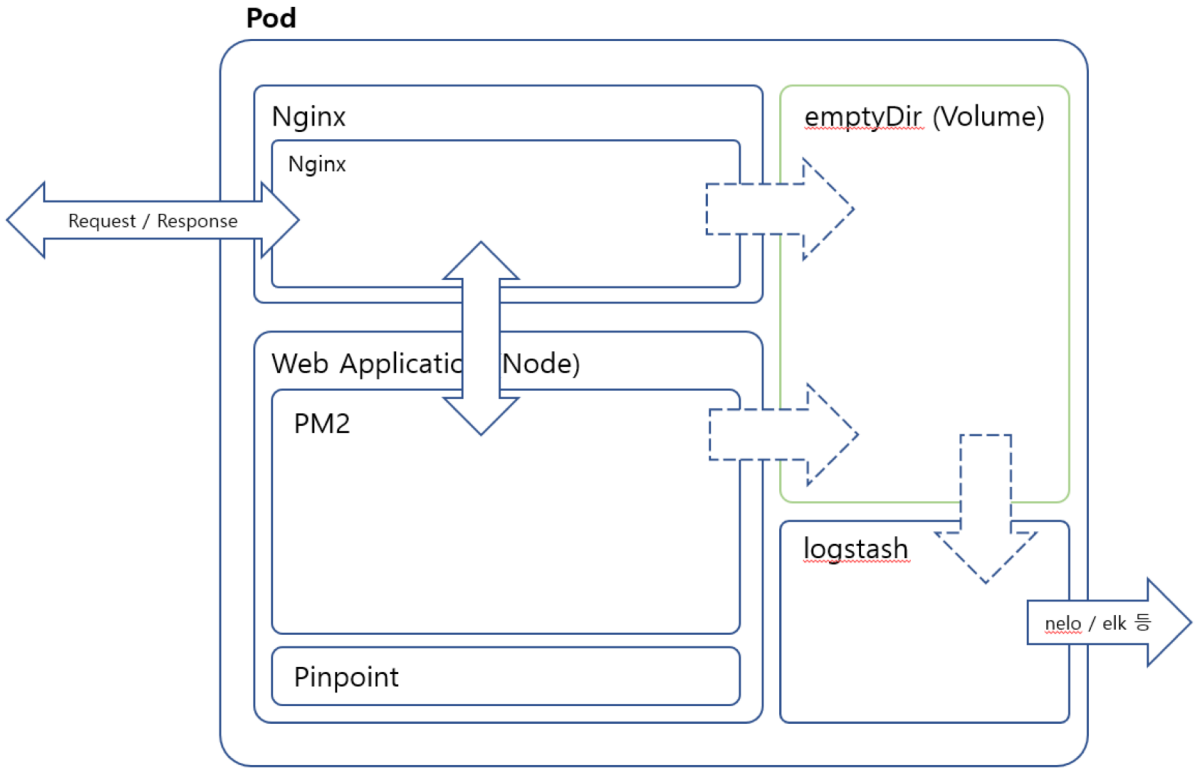
- 배포 모드
  - Blue/Green 배포
    - Service 설정을 변경해서 새 버전의 Deployment 으로 한꺼번에 교체
    - 기존 버전의 deployment와는 별개로 새로운 버전의 image를 갖는 deployment 추가
      - 기본 버전의 deployment 에 'blue' 라는 label 이 붙어 있다면, 신규 버전의 deployment 에는 'green' 이라는 label 을 붙임
    - Service 에서 load-balancing 하는 label 을 'blue' 에서 'green' 으로 교체 => 새 버전의 Deployment 으로 한꺼번에 교체
  - 가중치(Weight) 를 이용한 부하 분산
    - Service
    - Ingress Traefik

5. 쇼핑윈도 k8s 환경

- 쇼핑윈도 Pod 구성
- replicas, HPA

쇼핑윈도 k8s 환경

- 쇼핑윈도 Pod 구성
  - https://shopping.naver.com/market/home



쇼핑윈도 k8s 환경

- replicas, HPA
  - replicas (쇼핑윈도 모바일 기준)
    - min - 50
    - max - 90
  - Hpa
    - pod 의 평균 cpu 사용량이 6 core 를 넘는 경우, 최대 90 까지 Pod instance 늘림