2023 후기 졸업과제 중간보고서
지도교수 김태운

클라우드 배포를 위한 쿠버네티스 플랫폼: 개발자 통합 배포 관리 솔루션

Kubernetes platform for cloud deployments:
Developer-integrated deployment management solution

2024년 03월 26일

부산대학교 정보컴퓨터공학부/전기컴퓨터공학부
척척학사 팀(4번/분과C)
201824413 구성현
201812118 김찬호
201724551 임주은

차 례

간추린 글

다양한 요구 조건과 제약 사항 분석을 통해, 개발자들이 CI/CD 파이프라인을 통해 소프트웨어를 빠르고 안정적으로 배포할 수 있는 플랫폼을 제공하는 프로젝트를 진행합니다. 이를 위해 사용자의 인증부터 프로젝트 생성, 지속적인 통합 및 배포, 롤백 기능까지 다양한 기능을 구현합니다.

클라우드 환경의 자원을 이용해 유연한 확장을 고려하고 단일 공용 IP 주소로 인한 제약 사항을 해결합니다. 이러한 요구사항과 제약 사항을 고려하여 쿠버네티스 내부 구조부터 프론트엔드와 백엔드의 구현까지 다양한 영역에서 작업이 진행되고 있습니다. 현재 구성원들은 각자의 역할을 분담하여 인프라 구축, CI/CD 파이프라인 설정, 그리고 프론트엔드와 백엔드의 구현을 진행하고 있습니다.

프로젝트는 사용자의 효율성을 높이고 소프트웨어 배포 과정을 표준화함으로써 개발자들에게 큰 가치를 제공할 것으로 기대됩니다.

1. 요구 조건 및 제약 사항 분석

1.1. 과제 목표 및 요구 조건

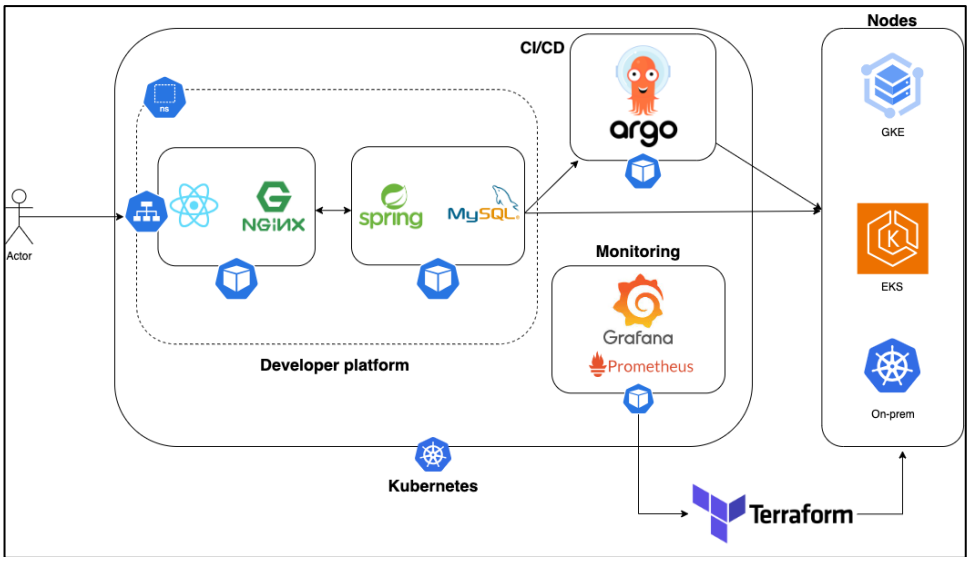


그림 1<전체 구성>

1.1.1. 과제 목표

개발자들이 배포 과정에서 겪는 부담을 최소화하면서 빠르고 안정적으로 소프트웨어를 배포할 수 있도록 CI/CD 파이프라인을 구축하고 운영하는 기능을 추상화 하여 플랫폼 형태로 개발자에게 제공한다. 개발자가 UI형태로 확인하고 활용할 수 있도록 제공함으로써 개발 효율을 높인다.

배포 전반 과정의 작업 방식을 표준화된 형식으로 제공함으로써 일관된 형태로 인프라를 관리한다. 또한 배포된 프로젝트에 대한 오너십을 관리하여 추후에 배포된 코드에 대한 특이사항이 생겼을 때, 빠르게 대응할 수 있도록 프로젝트 관련 정보를 유지보수한다.

1.1.2. 요구 조건

1. 사용자 인증
    - 로그인 시 GitHub 인증 및 OAuth 2.0 승인 프레임워크를 사용하여 인증된 사용자로써 권한을 얻는다.
    - 데이터베이스에 사용자의 데이터를 저장하고 관리한다.
2. 프로젝트 생성
    - 특정 GitHub 레포지토리를 선택하고 프로젝트로 생성할 수 있다.
3. 지속적인 통합
    - 대상 프로젝트의 소스코드가 변경되면 테스트 및 빌드를 수행한다.
    - 결과물을 바탕으로 컨테이너 이미지가 생성된다.
4. 지속적인 배포
    - 이미지가 성공적으로 생성되면 사용자는 UI를 통해 해당 컨테이너 이미지를 쿠버네티스에 배포한다.
    - 배포 성공 시 접속가능한 IP 주소 혹은 URL을 제공받는다.
5. 롤백
    - 버전 ID를 제공하면 해당하는 버전으로 배포 매니페스트를 롤백한다.
6. 서비스 제공
    - 웹 어플리케이션 형태로 서비스를 제공하여 편의성과 접근성을 높인다.
    - 별도의 환경설정, 시크릿 관리, 비용관리 등 부차적인 과정을 추상화하여 사용자에게 편의를 제공한다.

1.2. 제약 사항 분석 및 수정 사항

프로젝트 별 도메인	제약 사항	현재 공용 IP 주소가 하나만 존재하므로 개별 하위 도메인에 고유한 IP를 할당하는 것이 제한됩니다. 이러한 제약은 프로젝트 별로 최적의 기능과 보안을 위해 고유한 IP 주소가 필요한 다양한 온라인 서비스와 애플리케이션에 개별 도메인을 제공하는데에 어려움이 된다.
	해결 방안	단일 공용 IP 주소의 한계를 극복하기 위한 해결책은 Ingress NGINX를 리버스 프록시로 활용하는 것입니다. 이를 통해 단일 공용 IP로 전달되는 요청을 각 하위 도메인으로 라우팅되도록 하여, 제한된 수의 IP 주소로 여러 온라인 서비스 및 애플리케이션을 운영할 수 있습니다. 이 방법은 리소스 활용을 최적화하고 보안을 강화하며, 변화하는 요구에 유연하게 대응할 수 있는 장점을 제공합니다.
클라우드 버스팅	제약 사항	클라우드 환경에서 자원의 유연한 확장이 필요한 상황에서, 기존 자체 온프레미스 환경의 용량 한계로 인해 추가적인 요구사항을 충족시키는 것이 어려워졌습니다. 이로 인해 서비스의 안정성과 가용성이 저하될 우려가 있었습니다.
	해결 방안	클라우드 버스팅은 기존의 컴퓨팅 자원 외에 필요할 때 추가 클라우드 자원을 확보하여 서비스의 부하를 분산시키는 것을 의미합니다. 이는 예상치 못한 트래픽 급증이나 리소스 부족에 대응하고 서비스의 안정성과 가용성을 유지하는데 효과적입니다.

표 1<제약사항 및 해결방안>

2. 설계 상세화 및 변경 내역

2.1. 쿠버네티스

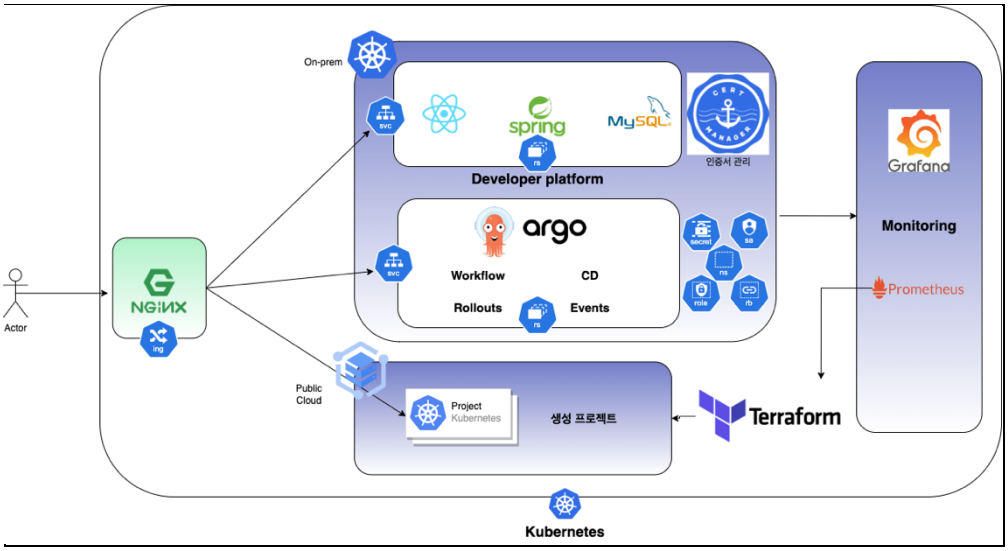


그림 2<쿠버네티스 내부 구조>

쿠버네티스는 컨테이너 오케스트레이션 시스템으로, 대규모 컨테이너 기반 애플리케이션을 관리하기 위한 플랫폼이다. 이를 효과적으로 활용하기 위해 다양한 구성 요소와 관련 도구들이 사용된다. 각 구성 요소와 도구는 클러스터 관리, 리소스 분배, 모니터링 등 다양한 기능을 제공하며, 그 중에서도 Ingress, Service, Replica Set, Secret, Role 등은 쿠버네티스에서 핵심적인 역할을 수행한다. 그 외에 서드파티 도구들은 쿠버네티스 환경을 보다 효과적으로 관리하고 모니터링하기 위해 사용된다.

Ingress	L7레이어에서 요청을 서브도메인에 따라 분기한다.
Service	다수의 파드의 단일 엔드포인트로 요청을 파드에 전달한다.
Replica Set	파드의 묶음으로 필요에 따라 유동적으로 파드의 수를 조절할 수 있다.
Secret	쿠버네티스 리소스가 인증을 위해 사용할 크리덴셜을 담고 있으며, 리소스들에 바인딩 되어 환경변수 형태로 이용된다.

Role	사용자 혹은 서비스 어카운트의 행동과 권한을 정의한다.
Role Binding	정의된 롤을 사용자 혹은 서비스어카운트에 연결하여 권한을 할당하는 메커니즘이다.
Namespace	클러스터 내의 리소스를 논리적으로 분리하고 격리하는 데 사용되는 가상 클러스터를 나타내는 논리적인 구분이다.
Service Account	애플리케이션이나 프로세스가 다른 리소스와 상호작용할 수 있는 인증 정보를 제공하는 엔터티이다.
Cert-manager	SSL/TLS 인증서를 관리하고 자동으로 갱신하는 도구이다.
Prometheus	시스템 및 서비스의 상태와 성능을 모니터링하는 오픈 소스 시스템이다.
Grafana	다양한 데이터 소스에서 수집한 데이터를 시각적으로 분석하고 모니터링하는 오픈 소스 플랫폼이다.
Terraform	인프라스트럭처를 코드로 관리하고 프로비저닝하기 위한 오픈 소스 도구이다. 온프레미스 리소스가 부족해져 더이상 파드가 생성되지 않을 때, 클라우드 리소스를 프로비저닝한다.

표 3<쿠버네티스 리소스>

2.2. CI/CD 파이프라인

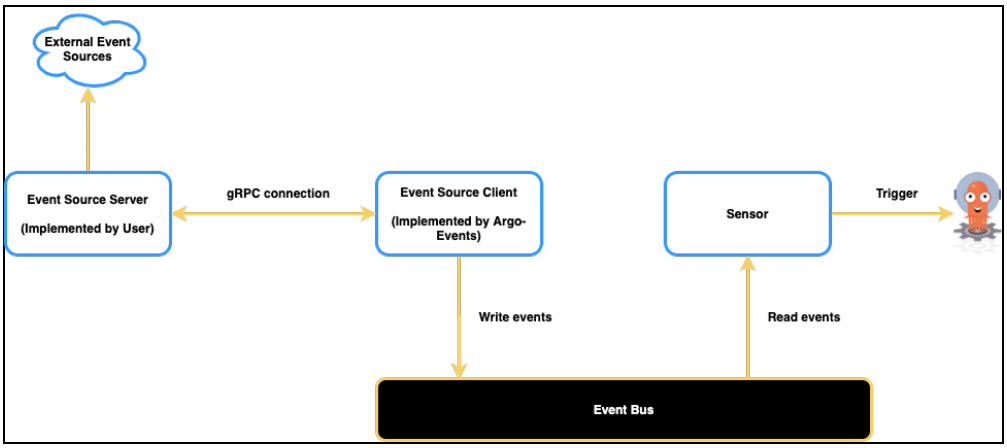


그림 3<CI/CD 파이프라인>

External Event Sources	GitHub, GitLab등 외부 소스에서 커밋 푸시 등과 같은 이벤트가 발생한다.
Event Source	외부 이벤트 소스에 대한 엔드포인트이다. Push될 때 발생하는 Webhook에 의한 요청에 의해 Event를 발행한다.
Sensor	발행된 Event를 읽어들여서 통합 및 배포를 트리거한다.
Workflows	GitHub 레포지토리를 클론한 후, 컨테이너 이미지를 빌드한다. 빌드된 이미지를 컨테이너 레지스트리에 푸시하고, Argo CD에 Sync 요청을 보낸다.
Argo CD	요청에 따라 배포되어 있는 컨테이너 이미지를 배포 매니페스트에 따라 Sync한다.

표 3<CI/CD>

2.3. Frontend

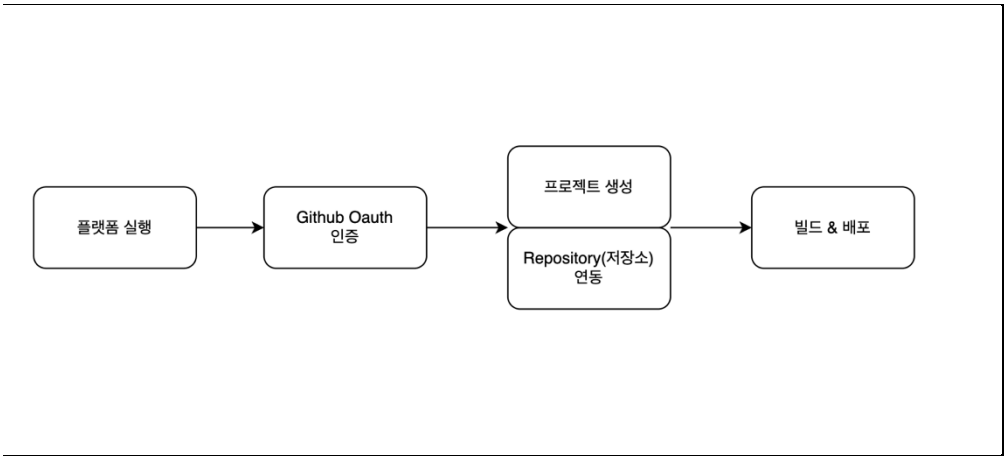


그림 4<사용자 흐름>

React를 활용한 웹페이지를 구현하여 사용자에게 최적의 경험을 제공한다. 사용자 경험을 중심에 두고 디자인된 화면은 직관적이고 사용하기 쉬우며, 시각적으로 매력적인 디자인으로 사용자의 관심을 끌어낸다. React의 뛰어난 성능과 유연성을 활용하여 동적인 웹 애플리케이션을 구축하여 사용자들에게 편리하고 효율적인 서비스를 제공한다.

로그인 화면	사용자 친화적인 UI와 보안 강화를 위한 인증 기능을 포함한 로그인 화면을 제공한다.
프로젝트 생성	사용자가 새로운 프로젝트를 생성하고 필요한 정보를 입력할 수 있도록 지원한다.
프로젝트 상세	각 프로젝트의 세부 정보를 표시하고 편집할 수 있는 기능을 제공하여 사용자가 프로젝트를 효율적으로 관리할 수 있다.
배포 로그	프로젝트의 배포 이력을 확인하고 관리할 수 있는 기능을 제공한다.

표 4<UI 구성>

2.4. Backend

서브도메인 할당

DNS 설정을 통해 서브도메인을 생성하고 관리합니다. ‘app-〈생성된 문자열〉.pitapat.ne.kr’같은 서브도메인을 생성한다.

1. 서브 도메인 문자열 생성

Base64문자열에서 O or 0, l or I 와 같이 시각적으로 동일하게 보이거나 헷갈리는 문자열을 제외한 Base58문자열을 통해 길이가 8인 문자열을 랜덤하게 생성하는 알고리즘을 구현한 뒤, 문자열을 생성하고 서브도메인을 지정한다.

2. DNS 레코드 설정

생성된 서브도메인을 사용하여 DNS 서비스(예: Amazon Route 53, Cloudflare, Google Cloud DNS 등)에 A 레코드나 CNAME 레코드를 설정해야 한다. 이는 대부분의 DNS 서비스 제공업체에서 제공하는 API를 통해 프로그래밍 방식으로 수행한다.

리버스 프록시

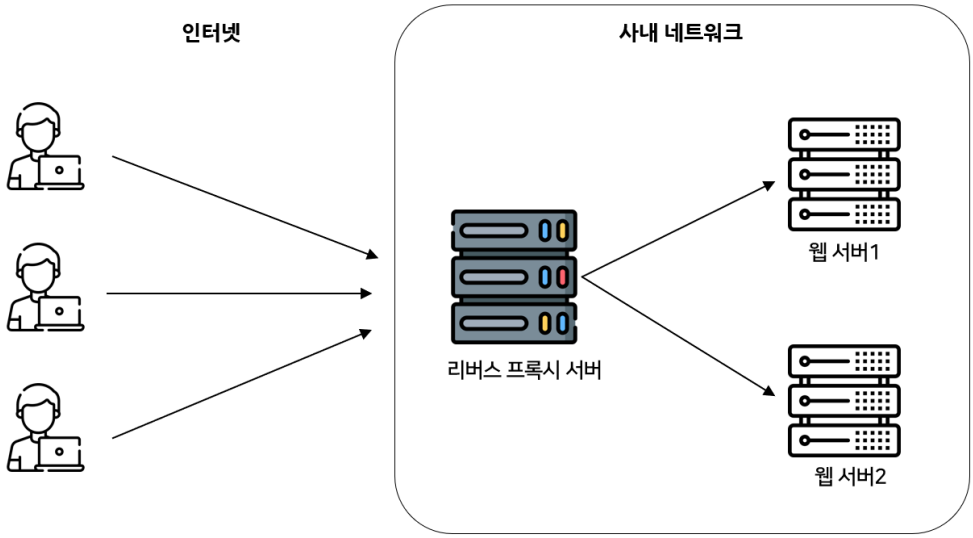


그림 5<리버스 프록시>

리버스 프록시란 클라이언트의 요청을 서버로 전달하는 중간자 역할을 수행한다. 클라이언트로부터 요청을 받아 하나 이상의 서버로 요청을 전달하고, 그 서버로부터 받은 응답을 다시 클라이언트에게 전달한다. 이 과정에서 리버스 프록시는 로드 밸런싱, 액세스 제어 등의 여러 추가적인 기능을 제공한다.
서브도메인을 특정 포트에 연결하는 작업도 리버스 프록시를 통해 관리된다.

3. 갱신된 과제 추진 계획

3.1. 기존 계획

진행된 것: 녹색, 진행 예정: 주황

구분	작업 일정													
	3월				4월				5월					6월
요구사항 분석														
인프라 구축														
Github 인증 구현														
프론트엔드 구현														
CI/CD 파이프라인 설정														
중간 보고서														
REST 정의 및 DB 설계														
API연동														
모니터링 설정 및 연동														
테스트 및 디버깅														
배포 테스트														
최종 배포														
최종 보고서														
발표 준비														

표 5 <개발 일정>

4. 과제 진행 내용

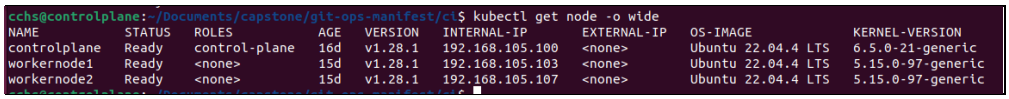
4.1. 구성원별 진척도

이름	분류	세부 역할 분담
구성현	인프라	● Kubernetes 클러스터 설정 및 관리 ● CI/CD 파이프라인 설정
김찬호	프론트엔드	● React 기술을 활용하여 웹 사이트의 프론트엔드 구현 ● 웹 사이트 디자인과 사용자 플로우 정의
임주은	백엔드	● OAuth 사용자 인증 구현

표 6 <진척도>

4.2. 보고 시점까지의 과제 수행 내용 및 중간 결과

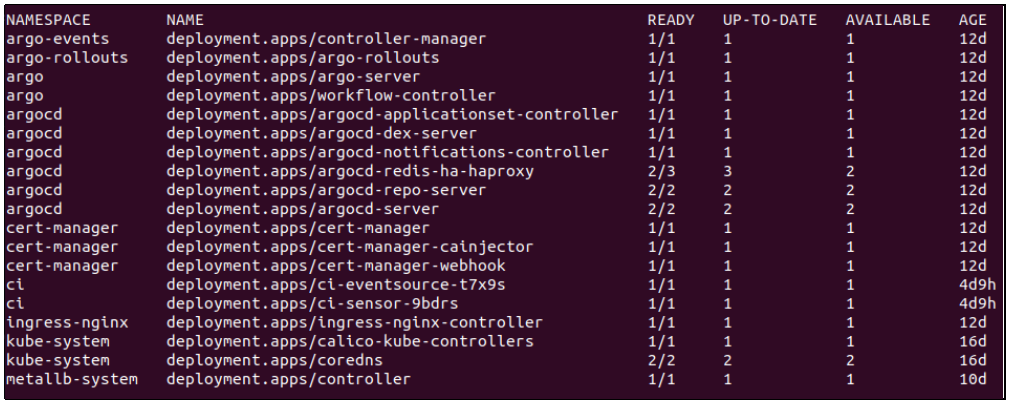
4.2.1. 인프라



```
cchs@controlplane:~/Documents/capstone/git-ops-manifest/ci$ kubectl get node -o wide
NAME           STATUS   ROLES           AGE   VERSION   INTERNAL-IP       EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION
controlplane   Ready    control-plane   16d   v1.28.1   192.168.105.100   <none>        Ubuntu 22.04.4 LTS   6.5.0-21-generic
workernode1    Ready    <none>          15d   v1.28.1   192.168.105.103   <none>        Ubuntu 22.04.4 LTS   5.15.0-97-generic
workernode2    Ready    <none>          15d   v1.28.1   192.168.105.107   <none>        Ubuntu 22.04.4 LTS   5.15.0-97-generic
```

그림 6<쿠버네티스 노드>

- Kubernetes 클러스터 구성



```
NAMESPACE        NAME                                                READY   UP-TO-DATE   AVAILABLE   AGE
argo-events      deployment.apps/controller-manager                  1/1     1            1           12d
argo-rollouts    deployment.apps/argo-rollouts                       1/1     1            1           12d
argo             deployment.apps/argo-server                         1/1     1            1           12d
argo             deployment.apps/workflow-controller                 1/1     1            1           12d
argocd           deployment.apps/argocd-applicationset-controller    1/1     1            1           12d
argocd           deployment.apps/argocd-dex-server                   1/1     1            1           12d
argocd           deployment.apps/argocd-notifications-controller     1/1     1            1           12d
argocd           deployment.apps/argocd-redis-ha-haproxy             2/3     3            2           12d
argocd           deployment.apps/argocd-repo-server                  2/2     2            2           12d
argocd           deployment.apps/argocd-server                       2/2     2            2           12d
cert-manager     deployment.apps/cert-manager                        1/1     1            1           12d
cert-manager     deployment.apps/cert-manager-cainjector             1/1     1            1           12d
cert-manager     deployment.apps/cert-manager-webhook                1/1     1            1           12d
ci               deployment.apps/ci-eventsource-t7x9s                1/1     1            1           4d9h
ci               deployment.apps/ci-sensor-9bdrs                     1/1     1            1           4d9h
ingress-nginx    deployment.apps/ingress-nginx-controller            1/1     1            1           12d
kube-system      deployment.apps/calico-kube-controllers             1/1     1            1           16d
kube-system      deployment.apps/coredns                             2/2     2            2           16d
metallb-system   deployment.apps/controller                          1/1     1            1           10d
```

그림 7<소프트웨어 구성>

- Argo CD, Events, Rollouts, Workflow 세팅
- Cert-manager 구성
- Ingress-nginx-controller 구성

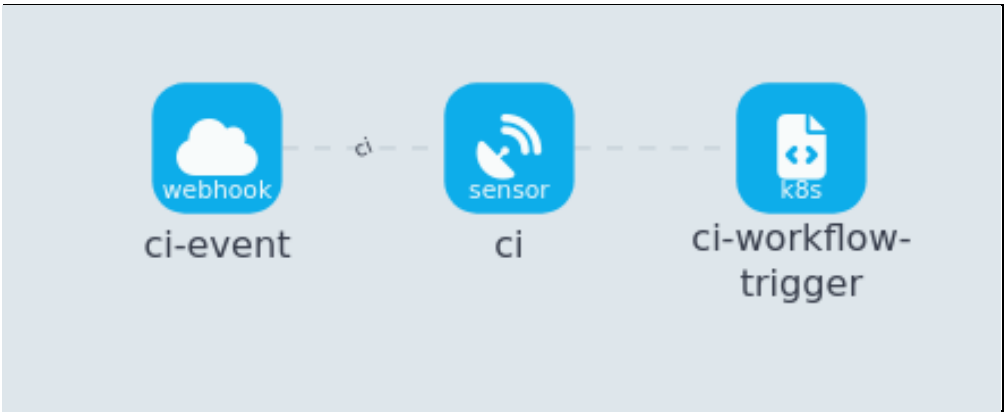


그림 8<CI/CD>

- Go 어플리케이션 템플릿 CI/CD 파이프라인 구성

그림 9<어플리케이션 템플릿>

- Go 어플리케이션 템플릿 구현

- 배포 템플릿 작성 – 진행중

→ 각 프레임워크에 해당하는 기본 템플릿 코드 작성

- PC 설치 현황

그림 10<설치 PC>

MSI Cubi5 mini PC 3대

CPU: Intel core i3-1215U 1.25GHz

MEM: 16GB

OS: Ubuntu 22.04.4

- 인프라 도구 버전

Kubernetes: v1.28.1

Argo CD: v2.10.2

Argo Rollouts: v1.6.6

Argo Workflows: v3.4.4

Argo Events: v.1.7.5

Ingress-nginx: v1.5.1

Cert-manager: v1.11.0

4.2.2. Frontend

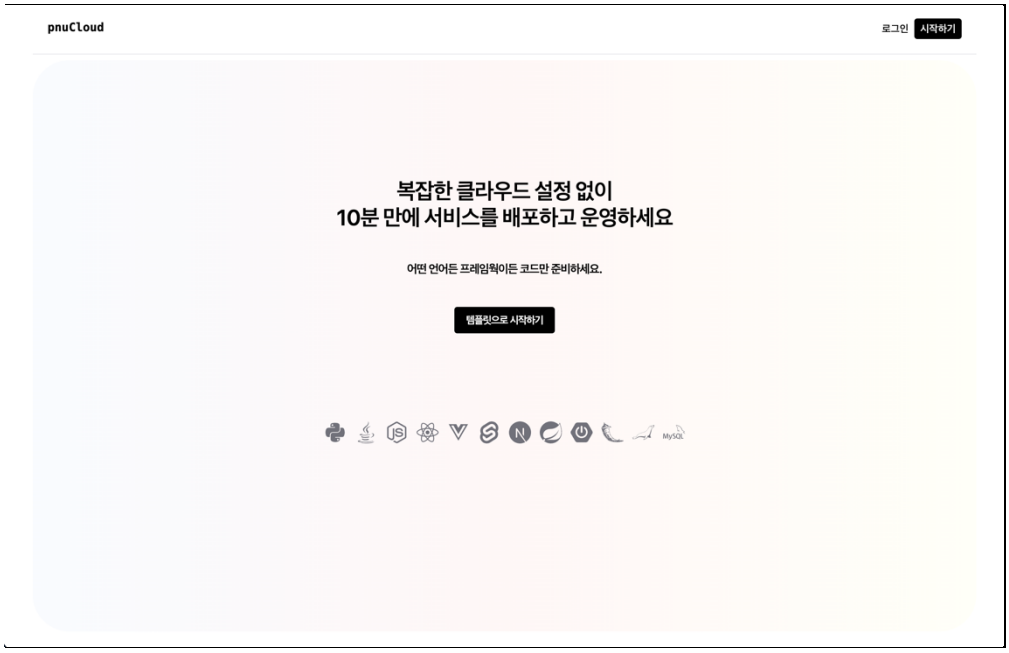


그림 4<메인 페이지>

- 웹 UI 설계(React 사용)
- 메인 페이지

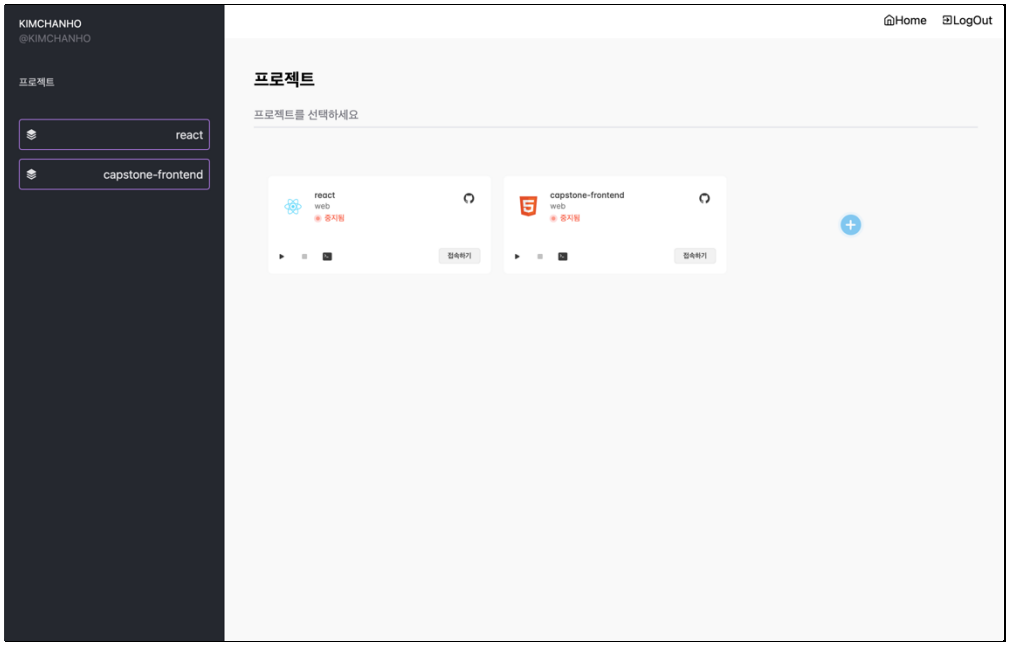


그림 5<프로젝트 페이지>

- 프로젝트 페이지

- 프로젝트 생성 컴포넌트 – 구현 중

→ 프로젝트를 생성할 때 서비스의 프레임워크, 언어, 버전 등 배포를 위한 정보를 입력 받는다.

- 생성된 프로젝트 렌더링 – 구현 중

→ 본인의 생성된 프로젝트 목록들을 HTTP통신을 통해 정보를 받아와서 렌더링 한다.

- 프로젝트 상세 페이지 – 진행 예정

→ 프로젝트의 생성된 URL, 배포 로그 내역, 설정(서비스 삭제 여부) 등 상세 내역을 볼 수 있는 컴포넌트가 필요하다.

- 프론트엔드 도구 버전

  NodeJs: 18.16.0

  npm: 9.5.1

  React: 18버전

4.2.3. Backend

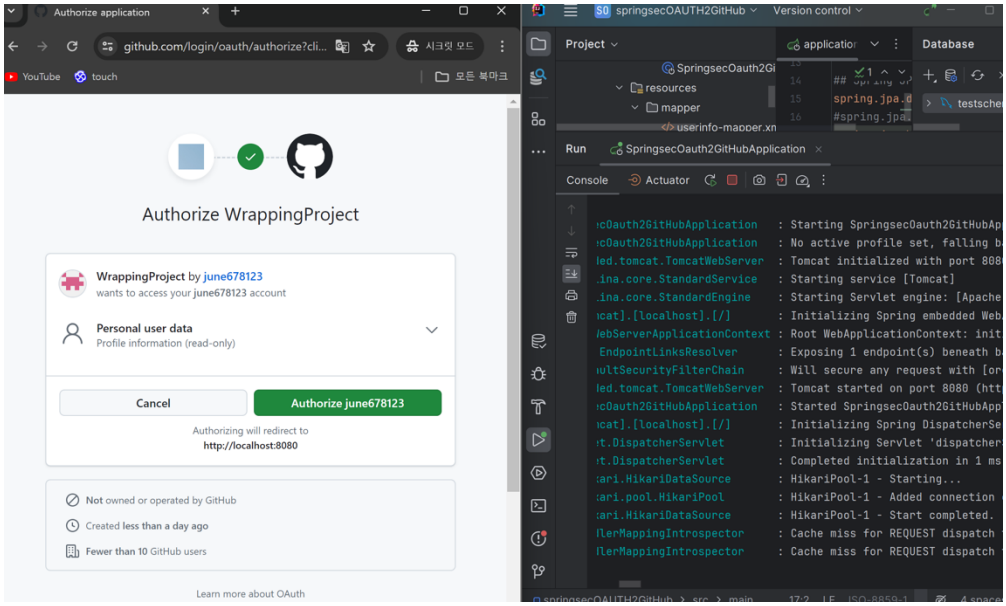


그림 6<OAuth>

- OAuth 방식으로 깃허브 계정을 통한 로그인 서비스 구현(진행중)

- 백엔드 도구 버전

  Spring 6.1.1

  Spring Boot 3.2.0

  Mybatis 3.0.3

그림 및 표 차례

그림 차례

표 차례