GitLab CI 파이프라인 구축하기

목차

Build를 위한 Dockerfile 작성
CI 파이프라인 설정하기
    ECR 생성하기
    Project Variables
    GitLab Runner 설치하기
문제 모음
    문제 1. Variables가 안들어가는 문제
    문제 2. AWS Credentials 문제
    문제 상황 3. Dokcer Daemon 권한 없음
    문제 상황 4.
    문제 상황 5. docker push - retrying… 문제

Build를 위한 Dockerfile 작성

▼ **Auth Dockerfile**

```
# Build Stage 1. Build Spring Boot App
FROM openjdk:11-jdk-slim as builder
COPY . .
# TODO: application.properties 복사
RUN chmod +x ./gradlew
RUN ./gradlew bootJAR

# Build Stage 2. Run Spring Boot App
FROM openjdk:11-jdk-slim
COPY --from=builder build/libs/*.jar app.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

▼ **Archive Dockerfile**

```
# Build Stage 1. Build Spring Boot App
FROM openjdk:11-jdk-slim as builder
COPY . .
# TODO: application.properties 복사
RUN chmod +x ./gradlew
RUN ./gradlew bootJAR

# Build Stage 2. Run Spring Boot App
FROM openjdk:11-jdk-slim
COPY --from=builder build/libs/*.jar app.jar
EXPOSE 8081
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

▼ **Signal Dockerfile**

```
# Build Stage 1. Build Spring Boot App
FROM openjdk:11-jdk-slim as builder
COPY . .
# TODO: application.properties 복사
RUN chmod +x ./gradlew
RUN ./gradlew bootJAR

# Build Stage 2. Run Spring Boot App
FROM openjdk:11-jdk-slim
COPY --from=builder build/libs/\*.jar app.jar
EXPOSE 8998
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

▼ **Frontend Dockerfile**

```
# Build Stage 1. Build React App
FROM node:alpine as builder
WORKDIR '/app'
COPY ./package.json .
RUN npm install
COPY . .
RUN npm run build

# Build Stage 2. Static Server
FROM node:alpine
COPY --from=builder /app/build build
RUN npm install -g serve
ENTRYPOINT [ "serve", "-s", "build" ]
```

CI 파이프라인 설정하기

Git 저장소에 push가 일어나면, 트리거 발생!

작성한 `.gitlab-ci.yml` 내용에 맞게 GitLab Runner가 작업을 진행함

ECR 생성하기

Amazon ECR (Elasitc Container Registry) 라는 컨테이너 저장소에 이미지를 올릴 예정이다.

이를 위해 이미지를 담을 저장소를 만들어보자.

우리는 총 4개의 이미지를 빌드할 것이므로, 4개의 레포지토리를 생성해야 한다.

```
aws ecr create-repository \
--repository-name reverse-app \
--image-scanning-configuration scanOnPush=true \
--region ap-northeast-2
```

```
aws ecr create-repository \
--repository-name auth-backend \
--image-scanning-configuration scanOnPush=true \
--region ap-northeast-2
```

```
aws ecr create-repository \
--repository-name archive-backend \
--image-scanning-configuration scanOnPush=true \
--region ap-northeast-2]
```

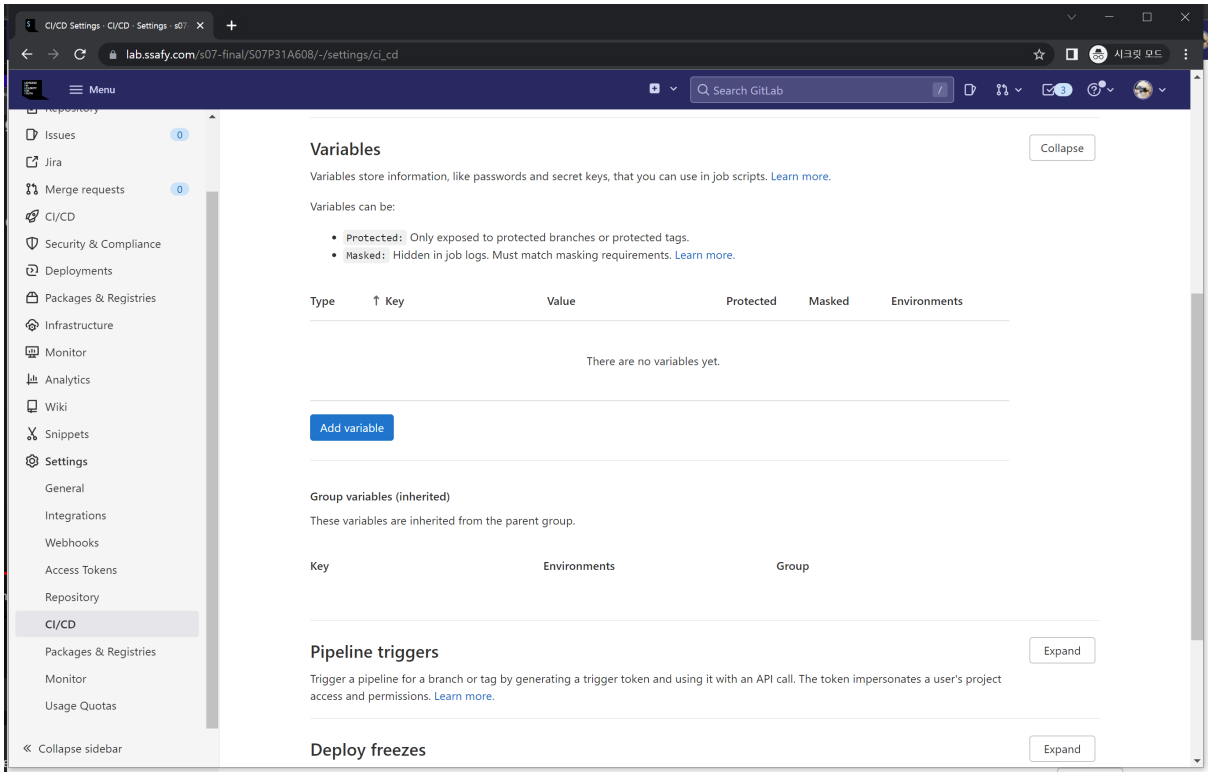
```
aws ecr create-repository \
--repository-name signal-backend \
--image-scanning-configuration scanOnPush=true \
--region ap-northeast-2
```

Project Variables

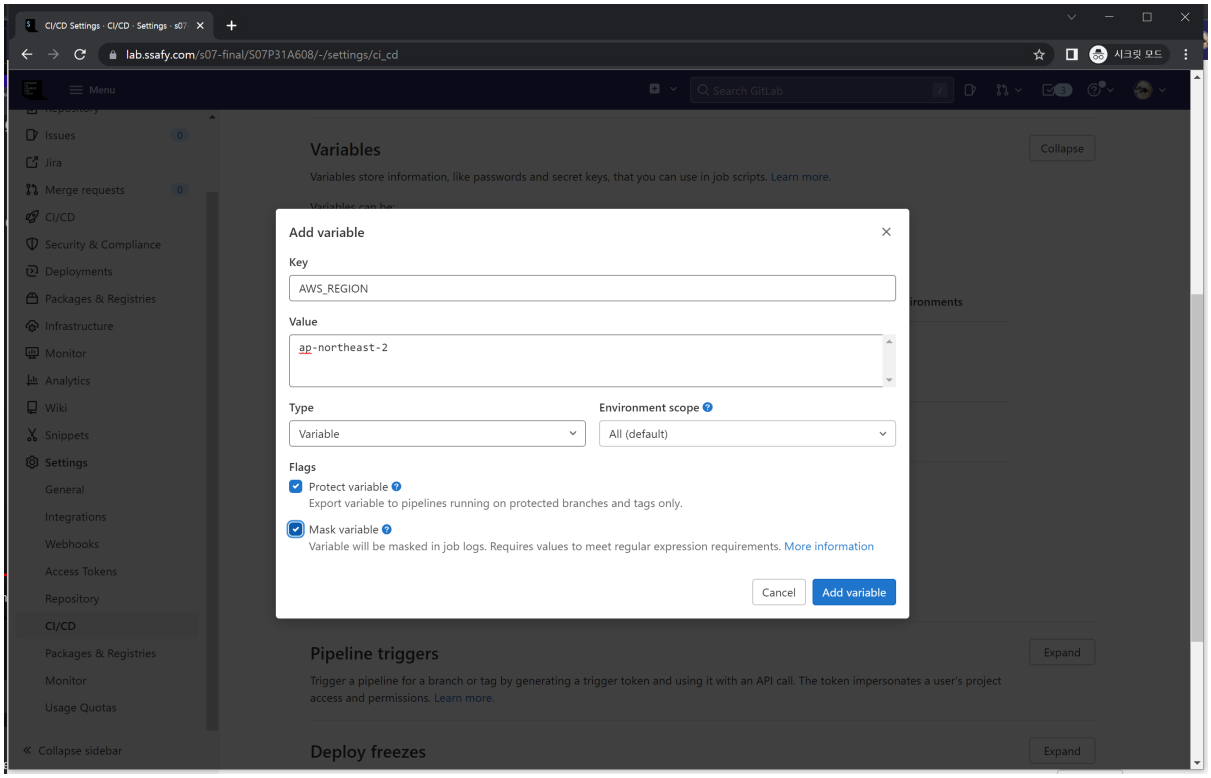
보안적인 이유로 Git Repository에 포함되서는 안되는 값들을 관리해주는 곳

ex. 토큰, 비밀번호 등

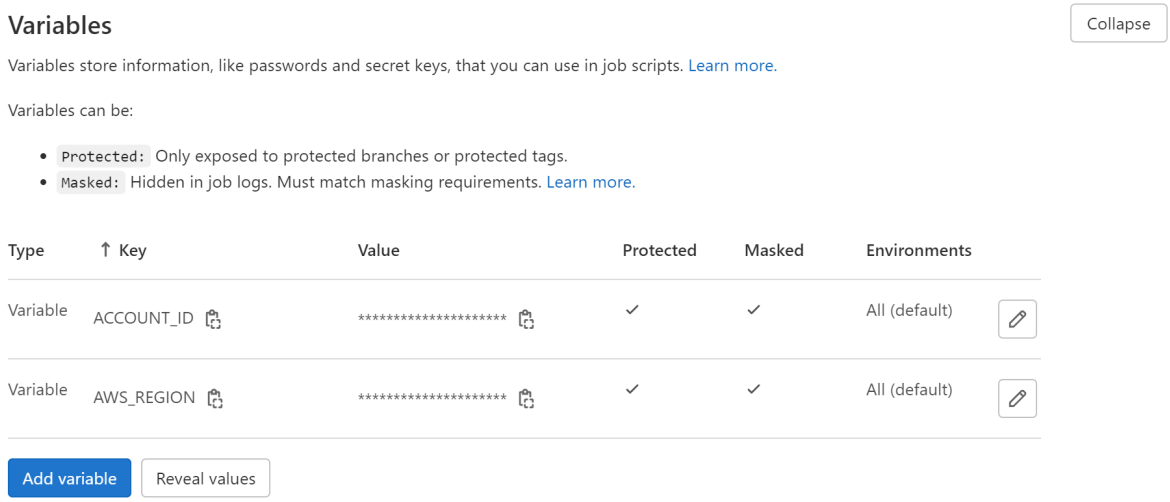
파이프라인 설정에서 해당 값들을 참조함



(1) Settings > CI/CD



(2) Key, Value 입력 후, Protect variable, Mask variable 클릭



(3) 잘 들어감을 확인

여기에 정의해놓은 Variabels들은 `.gitlab-ci.yml` 파일에서 `$변수명` 형태로 사용 가능하다.

`.gitlab-ci.yml`

```
variables:
    DOCKER_TLS_CERTDIR: "" # sharing same certificate between docker client and daemon
    DOCKER_HOST: tcp://docker:2375
    APP_NAME_1: reverse-app # frontend
    # APP_NAME_2: auth-backend
    APP_NAME_3: archive-backend
    # APP_NAME_4: signal-backend

stages:
    - build

build_image:
    stage: build
    image:
        name: amazon/aws-cli
        entrypoint: [""]
    services:
        - docker:20.10.16-dind # docker in docker (docker daemon)
    before_script:
        - amazon-linux-extras install docker
        # - aws --version
        # - docker --version
        - aws ecr get-login-password --region $AWS_REGION | docker login --username AWS --password-stdin $ACCOUNT_ID.dkr.ecr.$AWS_REGI
    script:
        - cp $ENV_FILE .env
        - docker build -t $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_1:$CI_PIPELINE_IID -f ./frontend/Dockerfile ./fronte
        - docker push $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_1:$CI_PIPELINE_IID
        - docker build -t $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_2:$CI_PIPELINE_IID -f ./backend/auth/AuthDockerfile
        - docker push $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_2:$CI_PIPELINE_IID
        - docker build -t $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_3:$CI_PIPELINE_IID -f ./backend/archive/ArchiveDocke
        - docker push $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_3:$CI_PIPELINE_IID
        - docker build -t $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_4:$CI_PIPELINE_IID -f ./signal/SignalDockerfile ./si
        - docker push $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_4:$CI_PIPELINE_IID
    tags:
        - reverse-runner
```

위 `build_image` Job은 다음과 같은 흐름으로 진행된다

1. 인증 토큰을 검색하고, 레지스트리에 대해 Docker 클라이언트 인증을 함

```
aws ecr get-login-password --region $AWS_REGION | docker login --username AWS --password-stdin $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazona
```

2. 도커 이미지를 빌드함

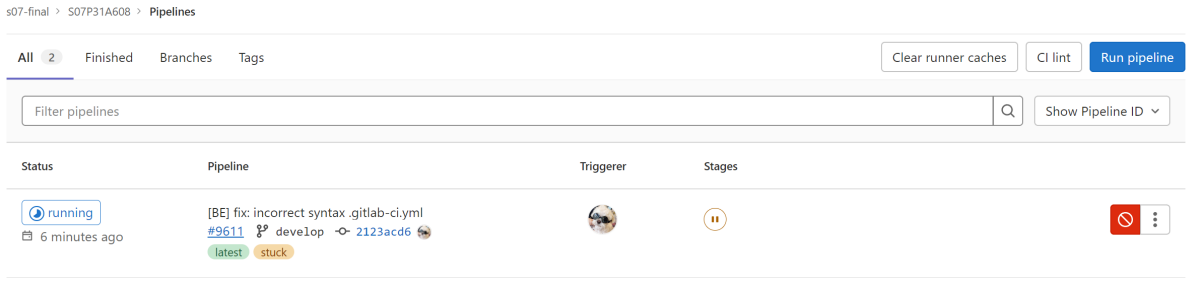
```
docker build -t reverse-frontend -f ./frontend/Dockerfile ./frontend
```

빌드 시, 이미지 이름 형식은 `[레포지토리 이름]:[버전]` 와 같이 정했다.

3. 해당 이미지를 AWS 레포지토리로 푸시하기

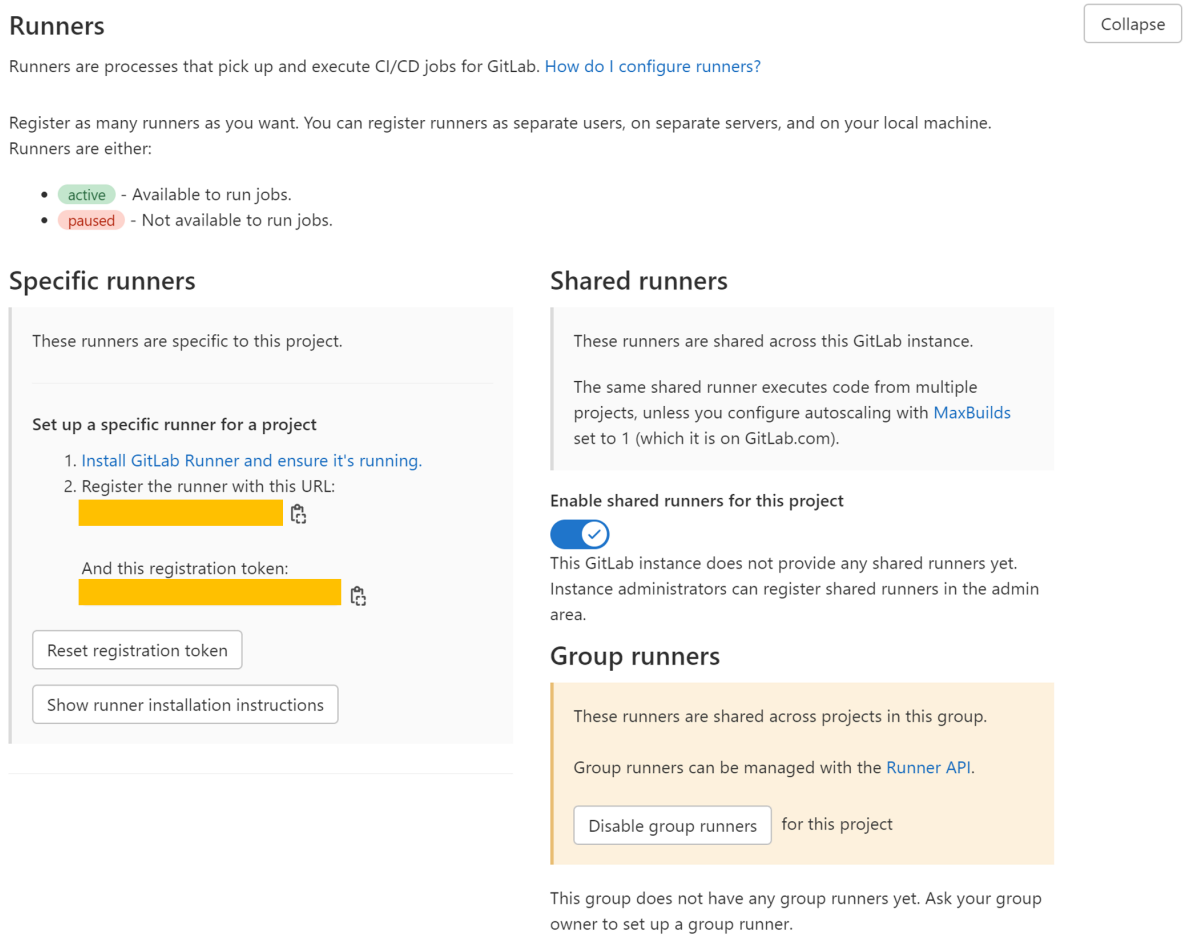
```
docker push $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/reverse-registry:latest
```

GitLab Runner 설치하기



`stuck` 상태에 있는 파이프라인

CI를 수행해줄 Runner가 없기 때문에 생기는 현상



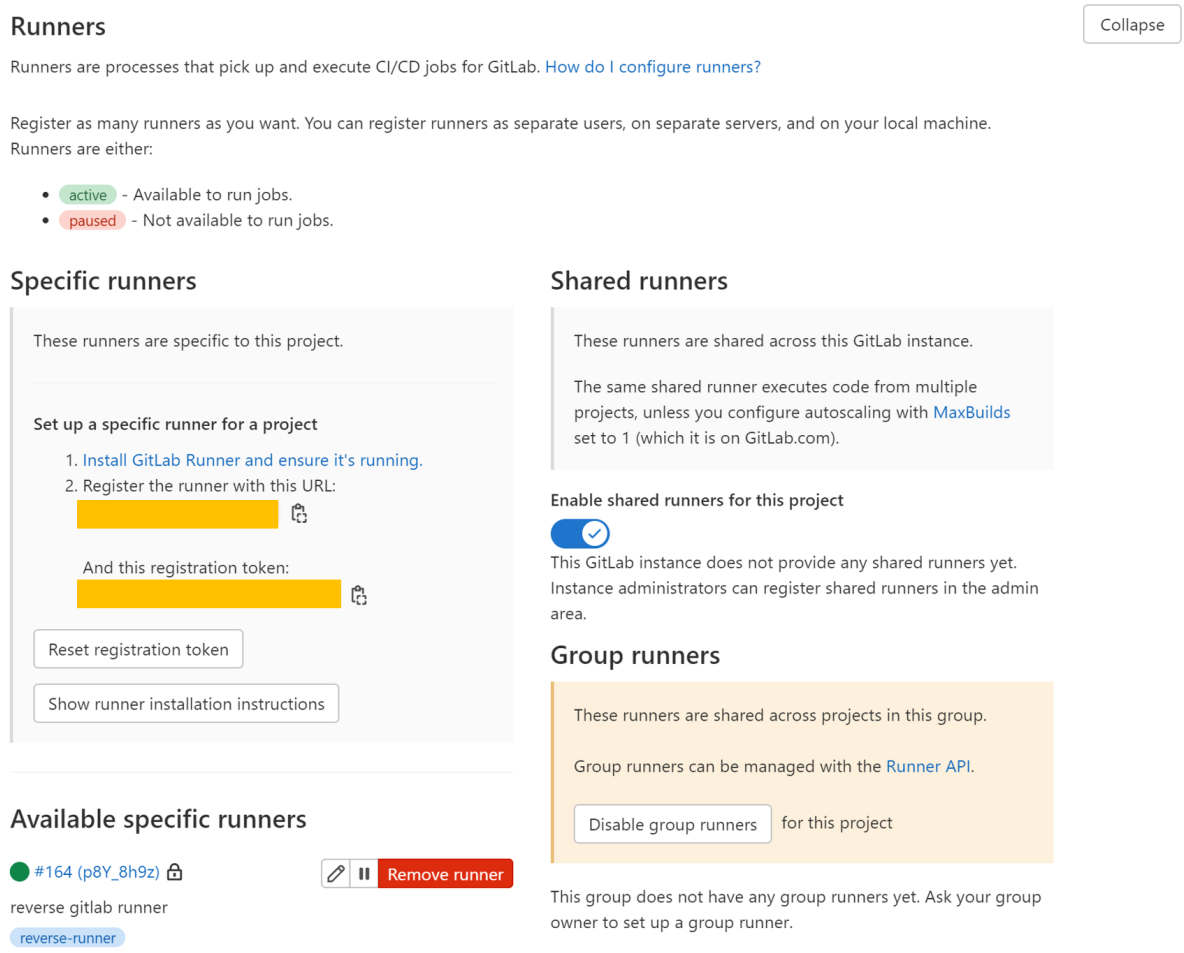
Gitlab Settings > CI/CD > Runners

해당 Gitlab 서버의 관리자라면, Shared Runners를 활성화시켜주면 되지만,

아쉽게도 서버의 관리자가 아니기 때문에 Specific Runners를 등록해서 사용할 예정이다.

우리의 Bastion Host인 EC2에 접속하여 우리만의 Gitlab Runner를 설치해보자.

설치 과정은 문제 3에 다시 첨부한다.



우리의 gitlab runner가 연결된 것을 확인

문제 모음

문제 1. Variables가 안들어가는 문제

계속 Variables 값이 안들어갔었음

develop branch에서 진행하고 있었는데, Protected Branches가 아니었음

https://insight.infograb.net/docs/user/settings/protected_branches/

문제 2. AWS Credentials 문제

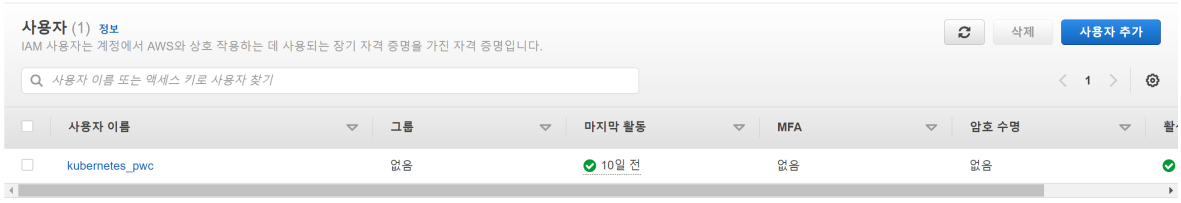
```
135 $ aws ecr get-login-password --region $AWS_REGION | docker login --username AWS --password-stdin $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com
136 Unable to locate credentials. You can configure credentials by running "aws configure".
137 Error: Cannot perform an interactive login from a non TTY device
```

ECR Private 레포지토리에 이미지를 push하고 pull 해오기 위해서는,

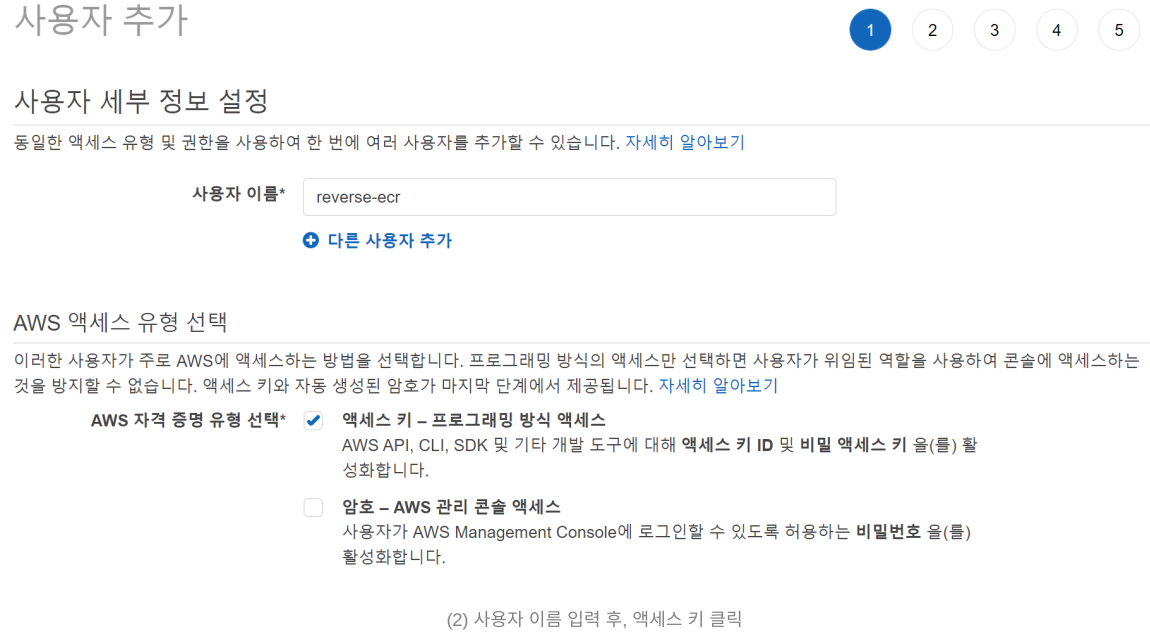
가장 먼저 해당 레포지토리에 Login, 즉 인증(authentication)을 받아야 한다.

그리고 이러한 권한을 가지고 있음을 증명하기 위한 특정 값이 필요하다.

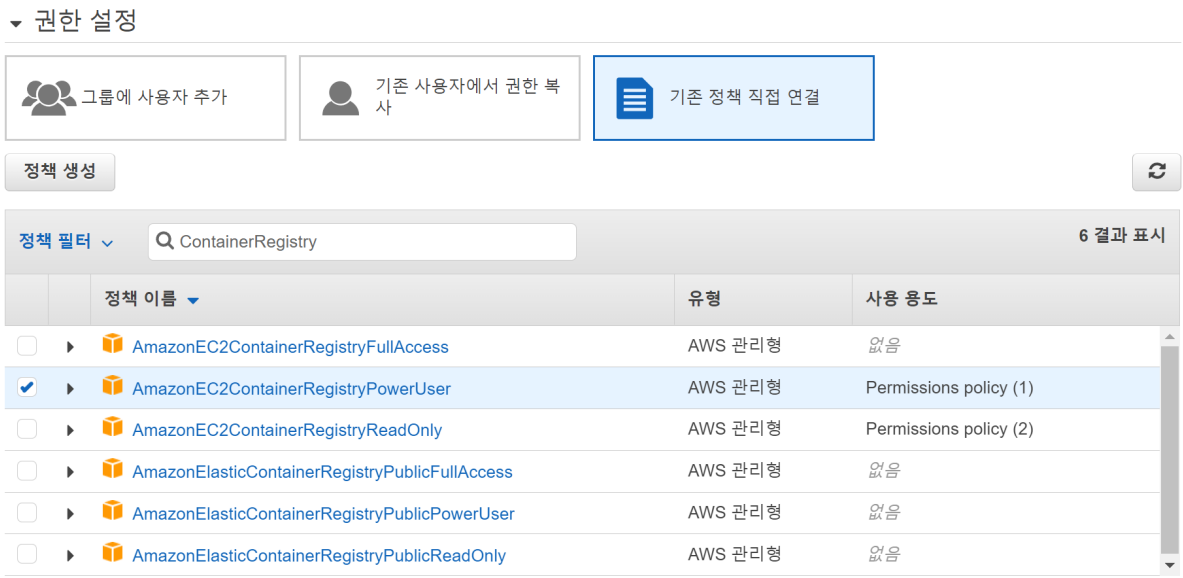
ECR 권한 전용 사용자 추가



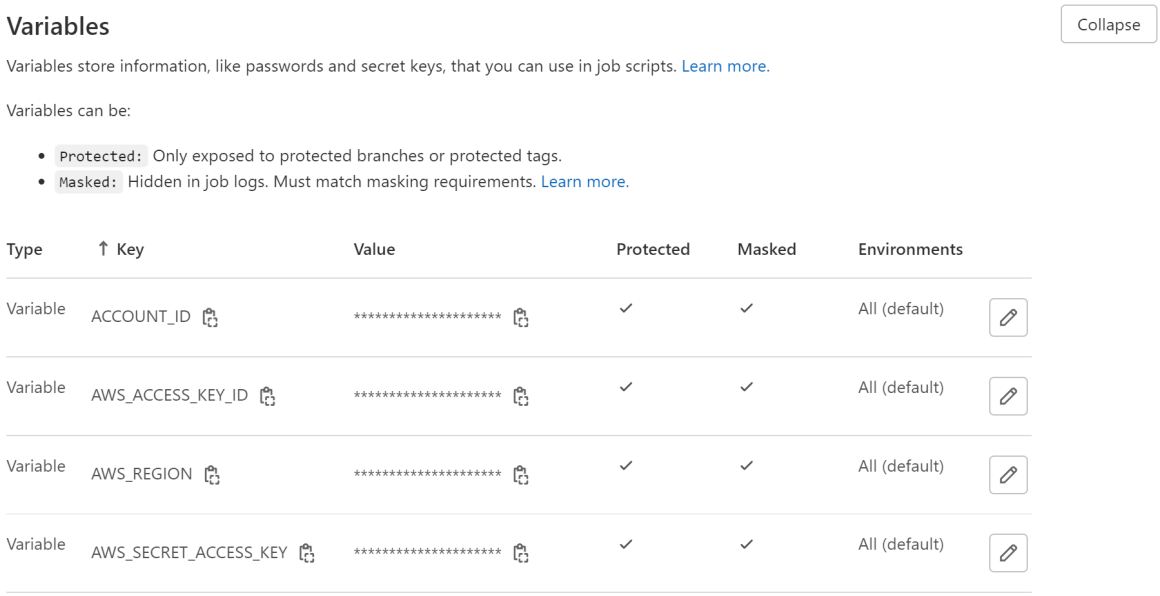
(1) IAM > 사용자 추가 버튼 클릭



(2) 사용자 이름 입력 후, 액세스 키 클릭



(3) AmazonEC2ContainerRegistryPowerUser 정책 클릭



제공해준 `access_key_id` 와 `secret_access_key` 를 GitLab의 Secret Variables에 넣어준다.

문제 상황 3. Dokcer Daemon 권한 없음

```
135 $ aws ecr get-login-password --region $AWS_REGION | docker login --username AWS --password-stdin $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com
136 error during connect: Post "http://docker:2375/v1.24/auth": dial tcp: lookup docker on 10.0.0.2:53: no such host
```

Shared Runner를 사용하면 해당 문제는 발생하지 않는다고 한다…

하지만 나에게는 Shared Runner를 Enable할 권한이 없다…

[Gitlab] dind 란?, 트러블 슈팅, error during connect: Post http://dind:2375/v1.40/auth: dial tcp: lookup dind on 172.31.0.2:53: no such host

dind 란? 젠킨스에서 보는 dind 깃랩에서의 dind Gitlab runner 에서 마주한 에러 해결방법 Docker in Docker (DinD) 란? 도커 안에 도커라는 의미이다. 도커 바이너리를 설정하고 컨테이너 내부의 격리된 Docker 데몬을 실행하는 작업 CI 측면에서 접근한다면 job(task)을 수행하는 Executor(Agent)가 Docker Client와 Docker Daemon 역할까지 하게 되어 도커 명령을 수행하는데 문제가 없어진다.

https://nearhome.tistory.com/141

다시 나만의 Gitlab Runner를 설치하러 가자…

```
sudo gitlab-runner unregister --all-runners
```

```
sudo gitlab-runner register
# url: OUR_URL
# registration-token: OUR_REGISTRATION_TOKEN
# description: reverse gitlab runner
# tag: reverse-runner
# executor: docker
# docker-image: "docker"
```

“/etc/gitlab-runner” 의 config.toml

```
[runners.docker]
  tls_verify = false
  image = "ruby:2.7"
  privileged = true
```

privileged = true 로 변경하자

문제 상황 4.

```
215  $ docker build -t $REPO_NAME/frontend:1.0 -f ./frontend/Dockerfile ./frontend
216  Cannot connect to the Docker daemon at tcp://docker:2375. Is the docker daemon running?
```

도커 데몬 service container로 잘 띄워줬는데, 왜 연결을 못하니..!!! 날 의심하다니…!!!

Use Docker to build Docker images | GitLab

You can use GitLab CI/CD with Docker to create Docker images. For example, you can create a Docker image of your application, test it, and publish it to a container registry. To run Docker commands in your CI/CD jobs, you must configure GitLab Runner to support docker commands.

https://docs.gitlab.com/ee/ci/docker/using_docker_build.html#docker-in-docker-with-tls-enabled-in-the-docker-executor

문서 제대로 안읽은 내가 바보였던 걸로 하자!

문제 상황 5. docker push - retrying… 문제

```
$ docker push $ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/$APP_NAME_2:$TAG_NAME
The push refers to repository [[MASKED].dkr.ecr.[MASKED].amazonaws.com/auth_backend]
5d90ebea037a: Preparing
eb6ee5b9581f: Preparing
e3abdc2e9252: Preparing
eafe6e032dbd: Preparing
92a4e8a3140f: Preparing
5d90ebea037a: Retrying in 5 seconds
e3abdc2e9252: Retrying in 5 seconds
92a4e8a3140f: Retrying in 5 seconds
eb6ee5b9581f: Retrying in 5 seconds
eafe6e032dbd: Retrying in 5 seconds
5d90ebea037a: Retrying in 4 seconds
```

ECR에 없는 Registry로 push를 하려해서 생기는 문제이다.