CI/CD

순서대로 따라하면 CI/CD 끝! (약 15분 소요)

EC2

접속하기

EC2에 ssh를 사용하여 접속한다.

- pem 파일을 저장
- 저장한 경로에서 다음 명령어 실행 (처음 접속 시 known host 추가)

```
ssh -i J10S004T.pem ubuntu@j10s004.p.ssafy.io
```

Docker

> https://docs.docker.com/engine/install/ubuntu/#install-using-the-repository

Docker Engine 설치

1. 도커 `apt` 레포지토리 설정

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" |
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

2. 도커 패키지 설치

```
sudo apt-get install docker-ce docker-ce-cli containerd.io do
```

3. 도커가 설치되었는지 확인

```
docker -v
# Docker version 25.0.4, build 1a576c5
docker compose -v
# Docker Compose version v2.24.7
```

4. 도커 권한 그룹 설정: sudo 없이 docker 명령어 사용 가능

```
sudo groupadd docker
sudo usermod -aG docker $USER
```

EC2 기본 설정

1. `chatda-setup` 레포지토리 클론

```
git clone https://lab.ssafy.com/0onionion0/chatda-setup
cd chatda-setup
```

2. 환경 변수 설정 (직접 설정 또는 파일 추가)

▼ `.env`

```
# Jenkins container
OPENAI_API_KEY={OPENAI_API_KEY}
MYSQL_USER={MYSQL_USER}
MYSQL_PASSWORD={MYSQL_PASSWORD}
MYSQL_HOST=chatda-mysql
MYSQL_PORT=3306
```

```
MYSQL_DATABASE=chatda
# MySQL container
MYSQL_ROOT_PASSWORD={MYSQL_ROOT_PASSWORD}
MYSQL_DATABASE=chatda
MYSQL_USER={MYSQL_USER}
MYSQL_PASSWORD={MYSQL_PASSWOR}
```

3. Secret 추가

▼ `./server/keys/google-cloud-key.json`

```
{
  "type": "service_account",
  "project_id": "active-incline-388904",
  "private_key_id": "private_key_id",
  "private_key": "private_key_id",
  "client_email": "ieum-878@active-incline-388904.iam.gserviceaccount.com",
  "client_id": "105566186475653400788",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/ieum-878%40active-incline-388904.iam.gserviceaccount.com",
  "universe_domain": "googleapis.com"
}
```

4. Docker Compose 실행

```
docker compose up -d
```

Nginx

- `80`, `443` 포트를 listen한다.
- `/api` : 기본 API 서버

- `/jenkins` : 젠킨스 서버

HTTPS 설정

Docker Compose 실행 완료 후 아래 명령어 실행

```
docker compose exec nginx certbot --nginx --agree-tos
```

이메일, 도메인을 입력하면 완료된다.

Jenkins

> https://j10s004.p.ssafy.io/jenkins

Jenkins 설정

처음 접속 시

- **Unlock Jenkins**: 아래 명령어 실행 후 출력 붙여넣기

```
docker compose exec jenkins cat /var/jenkins_home/secrets/initialAdminPassword
```

- **Create First Admin User**: 관리자 계정 생성, 알아서 하자.
- **Jenkins URL**: `https://j10s004.p.ssafy.io/jenkins` (자동 설정)

플러그인 설치

- **System Configuration - Plugins - Available plugins**에서 아래 플러그인 체크
  - GitLab Plugin
  - Pipeline: Stage View
- **+ Install** 클릭해서 설치

Credentials 추가

- **Security - Credentials - System - Global credentials** 접속 (필요 시 Store, Domain 변경 가능)
- **+ Add Credentials** 클릭 후 GitLab 로그인 정보 입력

    - **Kind:** `Username with password`
    - **Scope:** `Global (Jenkins, nodes, items, all child items, etc)`
    - **Username**: GitLab 아이디
    - **Password**: GitLab 비밀번호
    - **ID:** `gitlab-personal-login`
- **Create** 클릭

GitLab 연결 설정

- **System Configuration - System**에서 아래 항목 설정
- **GitLab**
    - Connection name: 깃랩 연결 이름 (자유)
    - GitLab host URL: `https://lab.ssafy.com`
    - Credentials: 위 과정에서 추가한 GitLab Username with password 선택

Pipeline 추가 및 관리

Pipeline 생성

- 메인 페이지에서 좌측의 **새로운 Item** 클릭
- 작업 이름 입력
- 아래 목록에서 **Pipeline** 선택
- **OK** 클릭 (생성되고 설정 페이지로 이동됨)

Pipeline 설정

Item 페이지에서 좌측의 **구성**으로 접속

- **General - Build Triggers**에서 아래 항목 체크
    - **Build when a change is pushed to GitLab. GitLab webhook URL: …**
    - 하위 항목 중 **Push Events**만 체크, 나머지는 체크 해제
    - **고급 - Secret Token:** Generate 후 저장해두기
    - GitLab webhook URL 저장해두기
- **Pipeline** 스크립트 작성 (TODO: Jenkinsfile로 최대한 분리)

▼ (우리가 사용하는 Pipeline 목록)

```
pipeline {
    agent any

    stages {
        stage('Clone') {
            steps {
                echo 'Clone'
                git branch: 'develop',
                    url: 'https://lab.ssafy.com/s10-s-pr
oject/S10P21S004.git',
                    credentialsId: 'gitlab-personal-toke
n'
            }
        }

        stage('Copy key') {
            // when {
                // changeset 'server/**'
            // }
            steps {
                sh 'echo server changed'
                sh 'cp /var/server/compose.yaml ./serve
r/compose.yaml'
                sh 'cp /var/server/keys/google-cloud-ke
y.json ./server/chatdaAPI/secret/google-cloud-key.json'
            }
        }

        stage('Run Docker') {
            // when {
                // changeset 'server/**'
            // }
            steps {
                dir('server') {
                    sh 'echo server changed'
                    sh 'docker compose up -d'
```

```
                        }
                    }
                }
            }
        }
```

GitLab 연결 설정 (GitLab)

- GitLab 레포지토리에서 **Settings - Webhooks - Project Hooks**에서 **Add new webhook** 클릭
- 아래 항목 체크
  - **URL**: 위에서 저장한 GitLab webhook URL 입력
  - **Secret token**: 위에서 저장한 Secret Token 입력
  - **Trigger**에서 **Push events**만 체크, **Wildcard pattern**으로 `develop` 입력 (배포할 브랜치)
- **Add webhook** 클릭

💡 위 Pipeline 설정은 develop 브랜치 업데이트 기준으로 배포되므로 프로덕션 배포 시 수정하기

Maybe

```
#Jenkins Pipeline

1. git clone
2. docker image build & docker container run
3. change port & reload nginx
4. remove docker container, docker image
5. send Mattermost message
```

ELK

`docker-compose.yaml`

```
version: '3.7'

services:

  # The 'setup' service runs a one-off script which initializes users inside
  # Elasticsearch — such as 'logstash_internal' and 'kibana_system' — with the
  # values of the passwords defined in the '.env' file. It also creates the
  # roles required by some of these users.
  #
  # This task only needs to be performed once, during the *initial* startup of
  # the stack. Any subsequent run will reset the passwords of existing users to
  # the values defined inside the '.env' file, and the built-in roles to their
  # default permissions.
  #
  # By default, it is excluded from the services started by 'docker compose up'
  # due to the non-default profile it belongs to. To run it, either provide the
  # '--profile=setup' CLI flag to Compose commands, or "up" the service by name
  # such as 'docker compose up setup'.
  setup:
    profiles:
      - setup
    build:
      context: setup/
      args:
        ELASTIC_VERSION: ${ELASTIC_VERSION}
    init: true
    volumes:
      - ./setup/entrypoint.sh:/entrypoint.sh:ro,Z
      - ./setup/lib.sh:/lib.sh:ro,Z
      - ./setup/roles:/roles:ro,Z
    environment:
      ELASTIC_PASSWORD: ${ELASTIC_PASSWORD:-}
      LOGSTASH_INTERNAL_PASSWORD: ${LOGSTASH_INTERNAL_PASSW
ORD:-}
      KIBANA_SYSTEM_PASSWORD: ${KIBANA_SYSTEM_PASSWORD:-}
      METRICBEAT_INTERNAL_PASSWORD: ${METRICBEAT_INTERNAL_P
ASSWORD:-}
      FILEBEAT_INTERNAL_PASSWORD: ${FILEBEAT_INTERNAL_PASSW
ORD:-}
      HEARTBEAT_INTERNAL_PASSWORD: ${HEARTBEAT_INTERNAL_PAS
SWORD:-}
      MONITORING_INTERNAL_PASSWORD: ${MONITORING_INTERNAL_P
ASSWORD:-}
      BEATS_SYSTEM_PASSWORD: ${BEATS_SYSTEM_PASSWORD:-}
    networks:
      - elk
    depends_on:
      - elasticsearch

  elasticsearch:
    build:
      context: elasticsearch/
      args:
        ELASTIC_VERSION: ${ELASTIC_VERSION}
    volumes:
      - ./elasticsearch/config/elasticsearch.yml:/usr/shar
e/elasticsearch/config/elasticsearch.yml:ro,Z
      - elasticsearch:/usr/share/elasticsearch/data:Z
    ports:
      - 9200:9200
      - 9300:9300
    environment:
      node.name: elasticsearch
      ES_JAVA_OPTS: -Xms512m -Xmx512m
      # Bootstrap password.
      # Used to initialize the keystore during the initial
startup of
```

```
      # Elasticsearch. Ignored on subsequent runs.
      ELASTIC_PASSWORD: ${ELASTIC_PASSWORD:-}
      # Use single node discovery in order to disable produ
ction mode and avoid bootstrap checks.
      # see: https://www.elastic.co/guide/en/elasticsearch/
reference/current/bootstrap-checks.html
      discovery.type: single-node
    networks:
      - elk
    restart: unless-stopped

  logstash:
    build:
      context: logstash/
      args:
        ELASTIC_VERSION: ${ELASTIC_VERSION}
    volumes:
      - ./logstash/config/logstash.yml:/usr/share/logstash/
config/logstash.yml:ro,Z
      - ./logstash/pipeline:/usr/share/logstash/pipeline:r
o,Z
    ports:
      - 5044:5044
      - 50000:50000/tcp
      - 50000:50000/udp
      - 9600:9600
    environment:
      LS_JAVA_OPTS: -Xms256m -Xmx256m
      LOGSTASH_INTERNAL_PASSWORD: ${LOGSTASH_INTERNAL_PASSW
ORD:-}
    networks:
      - elk
    depends_on:
      - elasticsearch
    restart: unless-stopped

  kibana:
    build:
      context: kibana/
      args:
        ELASTIC_VERSION: ${ELASTIC_VERSION}
    volumes:
      - ./kibana/config/kibana.yml:/usr/share/kibana/confi
g/kibana.yml:ro,Z
    ports:
      - 5601:5601
    environment:
      KIBANA_SYSTEM_PASSWORD: ${KIBANA_SYSTEM_PASSWORD:-}
    networks:
      - elk
    depends_on:
      - elasticsearch
    restart: unless-stopped

  filebeat:
    build:
      context: extensions/filebeat/
      args:
        ELASTIC_VERSION: ${ELASTIC_VERSION}
    # Run as 'root' instead of 'filebeat' (uid 1000) to all
ow reading
    # 'docker.sock' and the host's filesystem.
    user: root
    command:
      # Log to stderr.
      - -e
      # Disable config file permissions checks. Allows moun
ting
      # 'config/filebeat.yml' even if it's not owned by roo
t.
      # see: https://www.elastic.co/guide/en/beats/libbeat/
current/config-file-permissions.html
      - --strict.perms=false
    volumes:
      - ./extensions/filebeat/config/filebeat.yml:/usr/shar
e/filebeat/filebeat.yml:ro,Z
```

```
      - type: bind
        source: /var/lib/docker/containers
        target: /var/lib/docker/containers
        read_only: true
      - type: bind
        source: /var/run/docker.sock
        target: /var/run/docker.sock
        read_only: true
    environment:
      FILEBEAT_INTERNAL_PASSWORD: ${FILEBEAT_INTERNAL_PASSW
ORD:-}
      BEATS_SYSTEM_PASSWORD: ${BEATS_SYSTEM_PASSWORD:-}
    networks:
      - elk
    depends_on:
      - elasticsearch
      - logstash
      - kibana

networks:
  elk:
    driver: bridge

volumes:
  elasticsearch:
```

`elasticsearch/Dockerfile`

```
ARG ELASTIC_VERSION

# https://www.docker.elastic.co/
FROM docker.elastic.co/elasticsearch/elasticsearch:${ELASTIC_

# Add your elasticsearch plugins setup here
# Example: RUN elasticsearch-plugin install analysis-icu
```

`elasticsearch/config/elasticsearch.yaml`

```
---
## Default Elasticsearch configuration from Elasticsearch bas
## https://github.com/elastic/elasticsearch/blob/main/distrib
#
cluster.name: docker-cluster
network.host: 0.0.0.0

## X-Pack settings
## see https://www.elastic.co/guide/en/elasticsearch/referenc
#
xpack.license.self_generated.type: trial
xpack.security.enabled: true
```

`logstash/Dockerfile`

```
ARG ELASTIC_VERSION

# https://www.docker.elastic.co/
FROM docker.elastic.co/logstash/logstash:${ELASTIC_VERSION}

# Add your logstash plugins setup here
# Example: RUN logstash-plugin install logstash-filter-json
```

`logstash/config/logstash.yaml`

```
---
## Default Logstash configuration from Logstash base image.
## https://github.com/elastic/logstash/blob/main/docker/data/
#
http.host: 0.0.0.0

node.name: logstash
```

`logstash/pipeline/logstash.conf`

```
input {
    beats {
        port => 5044
    }

    tcp {
        port => 50000
    }
}

## Add your filters / logstash plugins configuration here

output {
    elasticsearch {
        hosts =>  "elasticsearch:9200"
        user => "logstash_internal"
        password => "${LOGSTASH_INTERNAL_PASSWORD}"
    }
}
```

`kibana/Dockerfile`

```
ARG ELASTIC_VERSION

# https://www.docker.elastic.co/
FROM docker.elastic.co/kibana/kibana:${ELASTIC_VERSION}

# Add your kibana plugins setup here
# Example: RUN kibana-plugin install <name|url>
```

`kibana/config/kibana.yaml`

```
---
## Default Kibana configuration from Kibana base image.
## https://github.com/elastic/kibana/blob/main/src/dev/build/
#
```

```
server.name: kibana
server.host: 0.0.0.0
elasticsearch.hosts: [ http://elasticsearch:9200 ]

monitoring.ui.container.elasticsearch.enabled: true
monitoring.ui.container.logstash.enabled: true

## X-Pack security credentials
#
elasticsearch.username: kibana_system
elasticsearch.password: ${KIBANA_SYSTEM_PASSWORD}

## Encryption keys (optional but highly recommended)
##
## Generate with either
##  $ docker container run --rm docker.elastic.co/kibana/kiba
##  $ openssl rand -hex 32
##
## https://www.elastic.co/guide/en/kibana/current/using-kiban
## https://www.elastic.co/guide/en/kibana/current/kibana-encr
#
#xpack.security.encryptionKey:
#xpack.encryptedSavedObjects.encryptionKey:
#xpack.reporting.encryptionKey:

## Fleet
## https://www.elastic.co/guide/en/kibana/current/fleet-setti
#
xpack.fleet.agents.fleet_server.hosts: [ http://fleet-server:

xpack.fleet.outputs:
  - id: fleet-default-output
    name: default
    type: elasticsearch
    hosts: [ http://elasticsearch:9200 ]
    is_default: true
    is_default_monitoring: true
```

```
xpack.fleet.packages:
  - name: fleet_server
    version: latest
  - name: system
    version: latest
  - name: elastic_agent
    version: latest
  - name: docker
    version: latest
  - name: apm
    version: latest

xpack.fleet.agentPolicies:
  - name: Fleet Server Policy
    id: fleet-server-policy
    description: Static agent policy for Fleet Server
    monitoring_enabled:
      - logs
      - metrics
    package_policies:
      - name: fleet_server-1
        package:
          name: fleet_server
      - name: system-1
        package:
          name: system
      - name: elastic_agent-1
        package:
          name: elastic_agent
      - name: docker-1
        package:
          name: docker
  - name: Agent Policy APM Server
    id: agent-policy-apm-server
    description: Static agent policy for the APM Server integ
    monitoring_enabled:
      - logs
      - metrics
    package_policies:
      - name: system-1
        package:
          name: system
      - name: elastic_agent-1
        package:
          name: elastic_agent
      - name: apm-1
        package:
          name: apm
        # See the APM package manifest for a list of possible
        # https://github.com/elastic/apm-server/blob/v8.5.0/a
        inputs:
          - type: apm
            vars:
              - name: host
                value: 0.0.0.0:8200
              - name: url
                value: http://apm-server:8200
```

`extensions/filebeat/Dockerfile`

```
ARG ELASTIC_VERSION

FROM docker.elastic.co/beats/filebeat:${ELASTIC_VERSION}
```

`extensions/filebeat/config/filebeat.yaml`

```
## Filebeat configuration
## https://github.com/elastic/beats/blob/main/deploy/docker/f
#

name: filebeat

filebeat.config:
  modules:
    path: ${path.config}/modules.d/*.yml
    reload.enabled: false

filebeat.autodiscover:
  providers:
    # The Docker autodiscover provider automatically retrieve
    # containers as they start and stop.
    - type: docker
      hints.enabled: true
      hints.default_config:
        type: container
        paths:
          - /var/lib/docker/containers/${data.container.id}/*
      templates:
        - condition:
            contains:
              docker.container.image: elasticsearch
          config:
            - module: elasticsearch
              server:
                input:
                  type: container
                  paths:
                    - /var/lib/docker/containers/${data.conta

processors:
  - decode_json_fields:
      fields: ["message"]
      process_array: false
      max_depth: 2
      target: ""
      overwrite_keys: true
      add_error_key: false

monitoring:
  enabled: false
  #enabled: true
  #logstash:
```

```
  #  username: beats_system
  #  password: ${BEATS_SYSTEM_PASSWORD}

#output.elasticsearch:
#  hosts: [ http://elasticsearch:9200 ]
#  username: filebeat_internal
#  password: ${FILEBEAT_INTERNAL_PASSWORD}

output.logstash:
  enabled: true
  hosts: [ 'logstash:5044' ]
  username: logstash_internal
  password: ${LOGSTASH_INTERNAL_PASSWORD}
## HTTP endpoint for health checking
## https://www.elastic.co/guide/en/beats/filebeat/current/htt
#

http:
  enabled: true
  host: 0.0.0.0
```