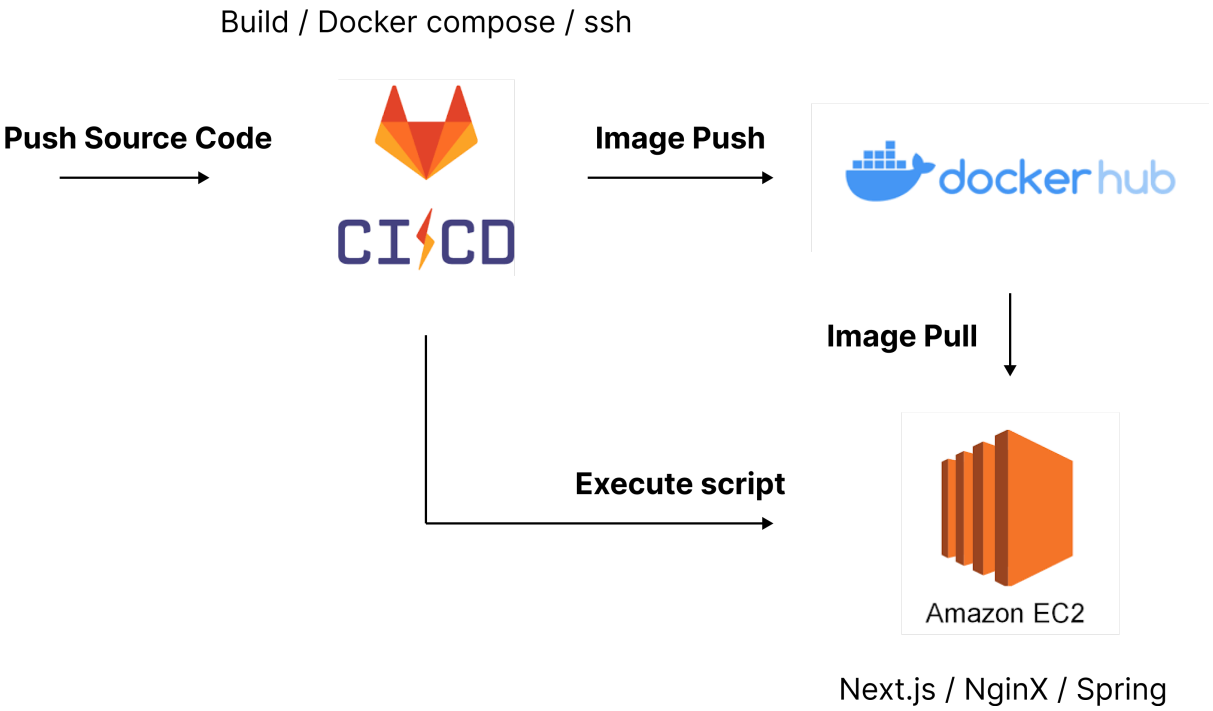
빌드/배포

GitLab CI/CD



- Backend
  - SpringBoot: 2.7.3
    - project Metadata
      - Group: com.ssafy
      - Artifact: uniqon
      - Name: uniqon
      - Package Name: com.ssafy.uniqon
  - jdk: zulu-openjdk:8
  - mysql: 8.0.29
  - Intellij: 2022.1.3
  - Dockerfile

```
FROM azul/zulu-openjdk:8
VOLUME /tmp
ARG JAR_FILE=./build/libs/uniqon-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

- Frontend
  - Node.js: 16.16
  - VS Code: 1.70.0
  - Dockerfile

```
FROM node:16.16-alpine3.15

WORKDIR /app

ADD . .

RUN yarn install
RUN yarn build

EXPOSE 3000

CMD ["yarn", "start"]
```

- Docker-compose

```
version: "3"

volumes:
  mysql_db_vol: {}
  redis_cache_vol: {}

services:
  nginx:
    container_name: nginx
    image: nginx
    volumes:
      - ./nginx/:/etc/nginx/conf.d/
      - ./data/certbot/conf:/etc/letsencrypt
      - ./data/certbot/www:/var/www/certbot
    command: "/bin/sh -c 'while :; do sleep 6h & wait $${!}; nginx -s reload; done & nginx -g \"daemon off;\"'"
    ports:
      - "80:80"
      - "443:443"
    expose:
      - 80
      - 443
    depends_on:
      - spring
      - react

  certbot:
    container_name: certbot
    image: certbot/certbot
    volumes:
      - ./data/certbot/conf:/etc/letsencrypt
      - ./data/certbot/www:/var/www/certbot
    entrypoint: "/bin/sh -c 'trap exit TERM; while :; do certbot renew; sleep 12h & wait $${!}; done;'"

  react:
    image: $FRONT_IMAGE_NAME
    container_name: react
    command: |
      yarn start

  spring:
    image: $BACK_IMAGE_NAME
    container_name: spring
    restart: always
    ports:
      - 8080:8080
    depends_on:
      - mysql
      - redis
    environment:
      SPRING_DATASOURCE_URL: jdbc:mysql://mysql:3306/${MYSQL_DATABASE}?useSSL=false&serverTimezone=UTC&useLegacyDatetimeCode=false
      SPRING_DATASOURCE_USERNAME: root
      SPRING_DATASOURCE_PASSWORD: ${MYSQL_ROOT_PASSWORD}

  redis:
    image: redis
    container_name: redis
    volumes:
      - redis_cache_vol:/data
    expose:
      - 6379

  mysql:
    image: mysql:8.0.29
    container_name: mysql
    environment:
      - MYSQL_DATABASE=${MYSQL_DATABASE}
      - MYSQL_ROOT_HOST=%
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
```

```
    command: ['--character-set-server=utf8mb4', '--collation-server=utf8mb4_unicode_ci']
    volumes:
      - mysql_db_vol:/var/lib/mysql
    expose:
     - 3306
```

- gitlab-ci.yml

```
# This file is a template, and might need editing before it works on your project.
# To contribute improvements to CI/CD templates, please follow the Development guide at:
# https://docs.gitlab.com/ee/development/cicd/templates.html
# This specific template is located at:
# https://gitlab.com/gitlab-org/gitlab/-/blob/master/lib/gitlab/ci/templates/Nodejs.gitlab-ci.yml

# Official framework image. Look for the different tagged releases at:
# https://hub.docker.com/r/library/node/tags/
stages:
  - spring-build
  - spring-dockerize
  - react-test
  - react-dockerize
  - deploy

cache:
  paths:
    - node_modules/

spring-build:
  image: openjdk:8
  stage: spring-build
  script: |
    echo -e $BACK_APIKEY > ./backend/uniqon/src/main/resources/application-API-KEY.properties
    value=`cat ./backend/uniqon/src/main/resources/application-API-KEY.properties`
    echo "$value"
    cd backend/uniqon/
    chmod +x gradlew
    ./gradlew build -x test
artifacts:
    paths:
      - ./backend/uniqon/build/libs/*.jar
    expire_in: 60 min
  only:
    - master
    - develop

spring-dockerize:
  image: docker:latest
  stage: spring-dockerize
  variables:
    DOCKER_TLS_CERTDIR: ""
  services:
    - docker:dind
  before_script: |
    cd backend/uniqon
    echo $BACK_DOCKER_HUB_PW | docker login -u $BACK_DOCKER_HUB_USER --password-stdin
script: |
    docker build -t $BACK_IMAGE_NAME .
    docker push $BACK_IMAGE_NAME
after_script: |
    docker logout
only:
    - master
    - develop

react-test:
  image: node:16.16
  stage: react-test
  script: |
    cd frontend/
    yarn install
    yarn test:ci
only:
    - master
    - develop
    - frontend

react-dockerize:
  image: docker:latest
  stage: react-dockerize
  variables:
    DOCKER_TLS_CERTDIR: ""
  services:
    - docker:dind
  before_script: |
```

```
    echo $FRONT_DOCKER_HUB_PW | docker login -u $FRONT_DOCKER_HUB_USER --password-stdin
script: |
    echo -e $FRONT_DOTENV_LOCAL > ./frontend/.env.local
    docker build -t $FRONT_IMAGE_NAME ./frontend/
    docker push $FRONT_IMAGE_NAME
after_script: |
    docker logout
when: on_success
  only:
    - master
    - develop

deploy:
  image: docker:latest
  stage: deploy
  variables:
    DOCKER_TLS_CERTDIR: ""
  tags:
    - deploy
  before_script: |
    mkdir -p ~/.ssh
    eval $(ssh-agent -s)
    echo $SSH_KNOWN_HOSTS >> ~/.ssh/known_hosts
    chmod 644 ~/.ssh/known_hosts
    chmod 600 $SSH_KEY
    ssh-add $SSH_KEY
script: |
    ssh ubuntu@"$DEPLOY_SERVER_IP" sudo bash deploy.sh
when: on_success
  only:
    - master
    - develop
```

AWS EC2

- deploy.sh

```
#!/bin/bash

echo "dahamkei11!" | docker login -u "hanwool77" --password-stdin

cd /home/ubuntu/uniq.on/
docker-compose down
docker rmi hanwool77/private:uniqon-frontend
docker rmi hanwool77/private:uniqon-backend
docker-compose up -d
# sudo bash init-letsencrypt.sh

docker logout
```

AWS S3

- 보안 정책
  - IAM 사용자 생성
    - AWS 자격 증명 유형: 엑세스 키 - 프로그래밍 방식 엑세스
    - IAM 사용자 S3 접근 권한 추가 - AmazonS3FullAccess
  - S3 버킷 생성
    - 버킷 정책 편집
      - Select Type of Policty: S3 Bucket Policy
      - Effect: Allow
      - Principal: *
      - Actions: GetObject, PutObject, DeleteObject
- aws.yml

```
cloud:
  aws:
    credentials:
      accessKey: ${AWS_ACCESS_KEY_ID}      # AWS IAM AccessKey
      secretKey: ${AWS_SECRET_ACCESS_KEY}  # AWS IAM SecretKey
    s3:
      bucket: ${bucket_name}
    region:
      static: ap-northeast-2  # 서울 region
    stack:
      auto: false
```