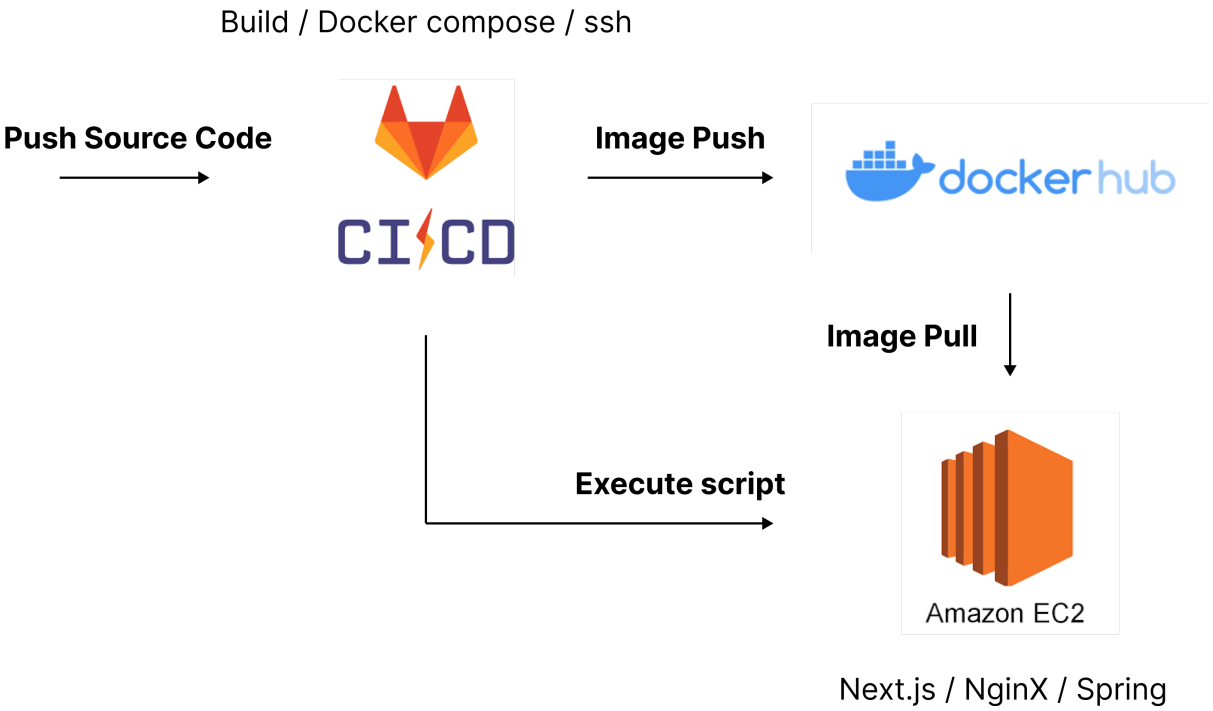
# 빌드/배포

## GitLab CI/CD




- Backend
  - SpringBoot: 2.7.3
    - project Metadata
      - Group: com.ssafy
      - Artifact: nuguri
      - Name: nuguri
      - Package Name: com.ssafy.nuguri
  - jdk: zulu-openjdk:11
  - mysql: 8.0.29
  - Intellij: 2022.1.3
  - Dockerfile

```
FROM azul/zulu-openjdk:11
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: {}
  # react_dist_vol: {}

services:
  nginx:
    container_name: nginx
    image: nginx
    volumes:
     # - react_dist_vol:/data/client/dist
      - ./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
    # volumes:
    #   - react_dist_vol:/app/dist
    command: |
      yarn start

  spring:
    image: $BACK_IMAGE_NAME
    container_name: spring
    restart: always
    ports:
      - 8080:8080
    links:
      - mongo
    depends_on:
      - mysql
      - redis
      - mongo
    environment:
      SPRING_DATASOURCE_URL: jdbc:mysql://mysql:3306/${MYSQL_DATABASE}?useSSL=false&serverTimezone=UTC&useLegacyDatetimeCode=false
      SPRING_DATASOURCE_USERNAME: root
      SPRING_DATASOURCE_PASSWORD: ${MYSQL_ROOT_PASSWORD}
      SPRING_DATA_MONGODB_HOST: mongo
      SPRING_DATA_MONGODB_PORT: 27017
      SPRING_DATA_MONGODB_DATABASE: chatdb
      SPRING_DATA_MONGODB_USERNAME: root
      SPRING_DATA_MONGODB_PASSWORD: ${MONGODB_ROOT_PASSWORD}
      SPRING_DATA_MONGODB_AUTHENTICATION_DATABASE: admin
      SPRING_REDIS_HOST: redis
      SPRING_REDIS_PORT: 6379
      TZ: Asia/Seoul
```

```
  redis:
    image: redis
    container_name: redis
    volumes:
      - redis_cache_vol:/data
    ports:
      - 6379:6379

  mysql:
    image: mysql:8.0.29
    container_name: mysql
    environment:
      - MYSQL_DATABASE=${MYSQL_DATABASE}
      - MYSQL_ROOT_HOST=%
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - TZ=Asia/Seoul
    command: ['--character-set-server=utf8mb4', '--collation-server=utf8mb4_unicode_ci']
    volumes:
      - mysql_db_vol:/var/lib/mysql
    ports:
      - 3306:3306


  mongo:
    image: mongo:latest
    container_name: mongo
    environment:
      - MONGO_INITDB_ROOT_USERNAME=root
      - MONGO_INITDB_ROOT_PASSWORD=${MONGODB_ROOT_PASSWORD}
      - MONGO_INITDB_DATABASE=chatdb
      - TZ=Asia/Seoul
    restart: always
    command: mongod
    ports:
      - 27017:27017
    volumes:
      - ./mongodb:/data/db
```

- 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:11
  stage: spring-build
  script: |
    echo -e $BACK_APIKEY > ./backend/nuguri/src/main/resources/application-API-KEY.properties
    value=`cat ./backend/nuguri/src/main/resources/application-API-KEY.properties`
    echo "$value"
    cd backend/nuguri/
    chmod +x gradlew
    ./gradlew build -x test
artifacts:
    paths:
      - ./backend/nuguri/build/libs/*.jar
    expire_in: 60 min
  only:
    - master
    - develop
    - backend

spring-dockerize:
  image: docker:latest
  stage: spring-dockerize
  variables:
    DOCKER_TLS_CERTDIR: ""
```

```
  services:
    - docker:dind
  before_script: |
    cd backend/nuguri
    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
    - backend

# 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: |
    docker build -t $FRONT_IMAGE_NAME ./frontend/
    docker push $FRONT_IMAGE_NAME
after_script: |
    docker logout
only:
    - master
    - develop
    - frontend

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
    - frontend
    - backend
```

## AWS EC2

- deploy.sh

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

cd /home/ubuntu/nuguri/
docker-compose down
docker rmi hanwool77/private:nuguri-frontend
docker rmi hanwool77/private:nuguri-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
```

## NGINX

- nginx.conf

```
upstream frontend {
    server react:3000;
}

upstream backend {
    server spring:8080;
}

server {
  listen 80;
  listen [::]:80;

  server_name k7a702.p.ssafy.io www.k7a702.p.ssafy.io;

  location /.well-known/acme-challenge/ {
    allow all;
    root /var/www/certbot;
  }

  location / {
    return 301 https://k7a702.p.ssafy.io$request_uri;
  }
}

server {
  client_max_body_size 128M;
  listen 443 ssl;
  server_name k7a702.p.ssafy.io;

  # ssl 인증서 적용하기
  ssl_certificate /etc/letsencrypt/live/k7a702.p.ssafy.io/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/k7a702.p.ssafy.io/privkey.pem;
  include /etc/letsencrypt/options-ssl-nginx.conf;
```

```
  ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

  location / {
    # root /data/client/dist;
    # index index.html index.htm;
    # try_files $uri /index.html;
    proxy_pass http://frontend;
    proxy_set_header Host $http_host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
  }

  location /auth {
    proxy_pass http://backend/auth;
    proxy_set_header Host $http_host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
  }

  location /app/sse {
    proxy_pass http://backend/app/sse;
    proxy_set_header Host $http_host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-forwarded-Proxo $scheme;
    proxy_buffering off;
    proxy_set_header Connection '';
    proxy_http_version 1.1;
  }

  location /app {
    proxy_pass http://backend/app;
    proxy_set_header Host $http_host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
  }

}
```