배포 정리

프로젝트 세팅

JVM : openjdk 17.0.13

웹서버 : Tomcat

IDE : IntelliJ IDEA 2024.3.1.1 (Ultimate Edition)

node.js : v22.12.0

VScode : 1.97.2

환경 변수

각 파일들을 Jenkins Credential로 관리

1. application.yml (Backend)

```
spring:
  application:
    name: muinus
  front:
    url: https://i12a506.p.ssafy.io

  # MySQL - 운영 DB
  datasource:
    url: jdbc:mysql://3.39.235.66:3306/hexa?useSSL=false&allowPublicKeyRetr
    username: root
    password: ssafy
    driver-class-name: com.mysql.cj.jdbc.Driver

  jpa:
    hibernate:
      ddl-auto: validate # 운영 환경이라면 'none' 또는 'validate' 권장
    show-sql: true
    database-platform: org.hibernate.dialect.MySQL8Dialect
```

```
elasticsearch:
  cluster:
    name: my-cluster
  rest:
    uris: http://3.39.235.66:9200
    username: your_username # 필요 시 설정
    password: your_password # 필요 시 설정

data:
  redis:
    host: 3.39.235.66
    port: 6379

  elasticsearch:
    repositories:
      enabled: true

cloud:
  aws:
    credentials:
      access-key: AKIAXNGUVPCXDAZVWQPG
      secret-key: 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
    s3:
      bucketName: muinus-bucket-dkfjka
    region:
      static: ap-northeast-2
    stack:
      auto: false

security:
  oauth2:
    client:
      registration:
        kakao:
          client-id: 6fc81aea9290c3fb44078c43a90b2acb
          redirect-uri: https://i12a506.p.ssafy.io/api/users/kauth
          authorization-grant-type: authorization_code
          scope:
              - profile_nickname
              - account_email
            client-name: Kakao
            client-authentication-method: POST
        provider:
          kakao:
            authorization-uri: https://kauth.kakao.com/oauth/authorize
            token-uri: https://kauth.kakao.com/oauth/token
            user-info-uri: https://kapi.kakao.com/v2/user/me

jwt:
  secretKey: t0u2eRrD1jjZV3JCLBNQKZjLjYmDb59m68WIaLRRQwvjG7NtyJk0

  access:
    expiration: 3600000000 # 1시간(60분) (1000L(ms → s) * 60L(s → m) * 60L(

  refresh:
    expiration: 1209600000 #  (1000L(ms → s) * 60L(s → m) * 60L(m → h) * 24

logging:
  level:
    org.springframework.jdbc.datasource.init: TRACE

OPENVIDU_URL: http://i12a506.p.ssafy.io:5443/
OPENVIDU_SECRET: ssafy
```

2. application.yml (Batch)

```
spring:
  application:
    name: muinus_batch

  main:
    allow-bean-definition-overriding: true
  # meta DB
  datasource-meta:
    jdbc-url: jdbc:mysql://3.39.235.66:3306/hexa_batch?useSSL=false&allowPu
    username: root
    password: ssafy
    driver-class-name: com.mysql.cj.jdbc.Driver

  # data DB
  datasource-data:
    jdbc-url: jdbc:mysql://3.39.235.66:3306/hexa?useSSL=false&allowPublicKe
    username: root
    password: ssafy
    driver-class-name: com.mysql.cj.jdbc.Driver
    hikari:
      maximum-pool-size: 30
      minimum-idle: 5
      idle-timeout: 30000
      connection-timeout: 20000

  batch:
    jdbc:
      initialize-schema: always
    job:
      enabled: false

  data:
    redis:
      host: 3.39.235.66
      port: 6379
      timeout: 60000
```

3. .env (Frontend)

```
REACT_APP_KAKAO_REST_API_KEY=3303e9b4860c22e2191afe3f6db54308
REACT_APP_KAKAO_JS_API_KEY=c2080ba7c02b1010d06cde9669469ac3
```

```
REACT_APP_Service_API_KEY=7x%2Bmm2IsLfjaDeGbXaLjOe6Lq5E81CciJXDu
REACT_APP_BACKEND_API_URL=https://i12a506.p.ssafy.io
```

배포 방법

1. docker 설치
2. Backend 배포

```
git clone https://lab.ssafy.com/s12-webmobile1-sub1/S12P11A506.git
cd Backend
docker-compose up --build -d
```

3. Backend Batch 서버 배포

```
// gitlab master 브랜치 클론 이후
cd Batch
docker-compose up --build -d
```

4. Frontend 서버 배포

```
git clone -b Frontend --single-branch https://lab.ssafy.com/s12-webmobile1-s
cd Frontend
docker-compose up --build -d
```

5. Redis 배포

```
docker run -d --name muinus-redis -p 6379:6379 redis:latest
```

6. MySQL 배포

```
docker run -d --name muinus-mysql -e MYSQL_ROOT_PASSWORD=ssafy -p
```

7. Openvidu 배포

}

~~~
sudo su
cd /opt
curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_l
cd openvidu

# .env 파일에서 DOMAIN_OR_PUBLIC_IP, OPENVIDU_SECRET 작성 (EX. i12a506
vi .env

# nginx 커스텀
docker-compose exec nginx cat /etc/nginx/conf.d/default.conf > custom-ngir
docker-compose exec nginx cat /etc/nginx/nginx.conf > nginx.conf
vi custom-nginx.conf
# 아래 내용 추가
```
    location / {
        root /usr/share/nginx/html;  # React 빌드 파일이 위치할 디렉토리
        index index.html;
        try_files $uri $uri/ /index.html;
    }

    location /api {
        proxy_pass http://localhost:8000;
        proxy_set_header Host $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;

        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header X-Real-IP $remote_addr;
        proxy_redirect off;

        # 쿠키 관련 설정
        proxy_cookie_path /api "/; HttpOnly; Secure; SameSite=None";
    }

    location /openvidu {
        proxy_pass http://openviduserver;
~~~

```
   }
```
```
./openvidu start
```

8. Elastic Search, Logstash, Kibana 배포

[ES.zip]

해당 파일에서 `docker-compose up -d` 로 Elasticsearch, Logstash, Kibana 실행 및 index 자동 생성

9. S3

```
1. AWS 계정 생성 후, s3 생성하여 access-key와 secret-key 발급
2. s3에서 권한으로 버킷 정책을
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "PublicReadGetObject",
         "Effect": "Allow",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::muinus-bucket-dkfjka/*"
      }
   ]
}
다음과 같이 설정
3. 또한 CORS 정책으로
[
   {
      "AllowedHeaders": [
         "*"
      ],
      "AllowedMethods": [
         "GET",
         "PUT",
         "POST",
         "DELETE",
         "HEAD"
      ],
      "AllowedOrigins": [
         "https://localhost:3000",
         "http://localhost:3000",
         "https://your-production-domain.com",
         "https://i12a506.p.ssafy.io"
      ],
      "ExposeHeaders": [
         "ETag",
         "x-amz-request-id",
         "x-amz-id-2",
         "Content-Length",
         "Content-Type"
      ],
      "MaxAgeSeconds": 3000
   }
]
다음과 같이 설정 후에 yml 파일로 설정하면 접속 가능
```

배포 시 특이사항 및 DB 접속 정보 등

구조

ssafy ec2 ㅡ docker ㅡ Jenkins

ㄴ Backend

ㄴ Frontend

ㄴ Backend-Batch

ㄴ Redis

ㄴ Elastic Search

ㄴ Logstash

ㄴ Kibana

ㄴ MySQL

ㄴ Openvidu

AWS — S3

MySQL

- hostname : ec2 퍼블릭 ip (ex. 3.39.235.66)
- port : 임의 설정 (ex. 3306)
- username : 임의 설정 (ex. root)
- password : 임의 설정 (ex. ssafy)

Redis

- host : ec2 퍼블릭 ip (ex. 3.39.235.66)
- port : 임의 설정 (ex. 6379)

Elastic Search

- host : ec2 퍼블릭 ip (ex. 3.39.235.66)
- port : 임의 설정 (ex. 9200)

Openvidu

- url : http://{{domain}}:{{port}} (ex. http://i12a506.p.ssafy.io:5443/)