

FORTING MANUAL VER 1.0



팀명	Team. COGETHER
프로젝트명	COGETHER
팀장	김진희
팀원	고나령, 박홍철, 신성은, 유지연

목차

1. 프로젝트 기술 스택
2. 백엔드 빌드 방법
3. 프론트엔드 빌드방법
4. MySQL 워크벤치 사용방법
5. EC2 세팅 및 명령어 정리
6. AWS S3
7. 기타 설정 파일 코드 (build.gradle, properties)

1. 프로젝트 기술스택

(1) 백엔드

- Springboot 2.7.2
- JAVA 11
- MySQL
- MySQL WORKBENCH
- IntelliJ IDEA (Ultimate)
- JWT
- JPA (H2)
- Springboot Starter Mail
- Swagger
- STOMP
- SpringSecurity

(2) 프론트엔드

- Vue3 (Vuex, VueRouter)
- Bootstrap
- HTML5
- CSS3
- JavaScript
- STOMP
- sockJS
- sweetalert2

(3) 배포 서버

- MobaXTerm
- nginx
- certbot
- aws
- s3

2. 백엔드 빌드방법

1. 인텔리 제이에서 **Gradle -> Tasks -> build -> bootJar**를 클릭
2. 오류없이 **jar** 파일이 생긴다면 **Project -> build -> libs**에 **jar** 파일이 생성 됨. (이미 있는 경우 갱신 됨)
3. 해당 **jar** 파일을 드래그 앤 드롭으로 **/home/ubuntu**로 서버에 올림

3. 프론트 엔드 빌드방법

1. `vscode`에서 `npm run build`를 입력
2. 생성된 `dist`파일을 드래그 앤 드롭으로 `/home/ubuntu`로 서버에 올림

4. MySQL 워크벤치 사용방법

1. 프로젝트용 새로운 User 생성
2. MySQL Connections 에 + 클릭
3. Connection Name 작성
4. Hostname에 i7a801.p.ssafy.io 입력
5. Username에 생성한 User 이름 등록
6. Password : Store in Vault 클릭 후 생성한 User의 PW인증
7. Test Connection 클릭 후 Success 뜨면 OK

5. EC2 세팅 및 명령어 정리

- MobaXterm 세팅

- MobaXterm 설치
- 실행 후 Session -> SSH
- Remote host 에 ubuntu@i7a801.p.ssafy.io 입력
- Advanced SSH settings 클릭
- Use private key 선택 후 지급 받은 pem키 선택
- OK

- 1. NginX 세팅

- a. `sudo apt update` (우분투 패키지 업데이트)
- b. `sudo apt install nginx` (Nginx 설치)
- c. `sudo nginx -v` (Nginx 버전 확인)
- d. `sudo service nginx restart` (nginx 재시작)

- 2. Certbot 사용해서 Htps 세팅

- a. snapd 설치
- b. `sudo snap install --classic certbot` (Certbot 설치)
- c. `sudo ln -s /snap/bin/certbot /usr/bin/certbot`
- d. `sudo certbot --nginx`
- e. `sudo certbot certonly --nginx`
- f. `sudo vi /etc/nginx/sites-available/default`

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    root /home/ubuntu/dist;
    index index.html index.htm index.nginx-debian.html;
    server_name _;
    location / {
        try_files $uri $uri/ =404;
    }
    location /api {
        proxy_pass http://localhost:8080;
        proxy_redirect off;
    }
}

server {
    root /home/ubuntu/dist;
    index index.html index.htm index.nginx-debian.html;
    server_name i7a801.p.ssafy.io; # managed by Certbot
    location / {
        try_files $uri $uri/ =404;
    }
    location /api {
```

```

        proxy_pass http://localhost:8080;
        proxy_redirect off;
    }
    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/i7a801.p.ssafy.io/fullchain.pem; #
managed by Certbot
    ssl_certificate_key
/etc/letsencrypt/live/i7a801.p.ssafy.io/privkey.pem; # managed by
Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by
Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by
Certbot
}
server {
    if ($host = i7a801.p.ssafy.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot
    listen 80 ;
    listen [::]:80 ;
    return 404; # managed by Certbot
}

```

3. 백엔드 SSL 인증을 위한 세팅

- certbot을 이용해 발급받은 fullchain.pem과 privkey.pem을 스프링 부트에서 인식할 수 있는 PKCS12형식으로 변경
- 변환된 키 파일을 로컬로 가져와 백엔드 파일에 적용
- application.properties에 설정 코드 추가

```

server.ssl.key-store-type= PKCS12
server.ssl.key-store-password= A801
server.ssl.key-store= keystore.pl2

```

4. mysql 세팅

- apt-get install mysql-server
- mysql -u root p (설치 확인)

5. java 세팅

- yum list java
- yum install (해당 버전)
- java -version (버전 확인)

6. 백엔드 빌드 파일 jar와 프론트 엔드 빌드 파일 dist를 mobaxterm에 드래그 앤 드롭

7. 실행 명령어

- java -jar <jar 파일 이름>
- nohup java -jar <jar 파일 이름> &
- ps -ef | grep <jar 파일 이름>
- kill -9 <PID>

6. AWS S3

1. S3 버킷 생성

- a. Create Bucket
- b. 이름과 지역 설정
- c. Set Permissions 설정
- d. 퍼블릭 정책 활성화

2. IAM 사용자 추가

- a. 사용자에서 사용자 추가
- b. User 이름 설정
- c. AmazonS3FullAccess 권한 부여
- d. 태그 설정
- e. 유저 생성
- f. accessKey와, secretKey를 Download받고 백엔드 파일에 설정 코드 추가

```
# aws
cloud.aws.credentials.access-key=AKIAU6JIIZHFP LIKSCLP
cloud.aws.credentials.secret-key=ERpPix6SNho6JQk09qs9tVF7cz1JSNQ
nw/bzBxyY
cloud.aws.s3.bucket=cogethera801
cloud.aws.region.static=ap-northeast-2
cloud.aws.stack.auto=false
cloud.aws.credentials.instance-profile=true
```

7. 프로퍼티스

- build.gradle

```
plugins {  
    id 'org.springframework.boot' version '2.7.2'  
    id 'io.spring.dependency-management' version '1.0.12.RELEASE'  
    id 'java'  
}  
  
group = 'com.cogether.api'  
version = '0.0.1-SNAPSHOT'  
sourceCompatibility = '11'  
  
configurations {  
    compileOnly {  
        extendsFrom annotationProcessor  
    }  
}  
  
repositories {  
    mavenCentral()  
}  
  
dependencies {  
  
    //lombok  
    compileOnly 'org.projectlombok:lombok'  
    annotationProcessor 'org.projectlombok:lombok'  
  
    //Spring Data JPA  
    implementation  
    'org.springframework.boot:spring-boot-starter-data-jdbc'  
    implementation  
    'org.springframework.boot:spring-boot-starter-data-jpa'  
    runtimeOnly 'mysql:mysql-connector-java'  
  
    //jwt  
    implementation 'io.jsonwebtoken:jjwt:0.9.1'  
  
    //SpringBoot  
    implementation  
    'org.springframework.boot:spring-boot-starter-security'  
    implementation 'org.springframework.boot:spring-boot-starter-web'  
  
    //aws  
    implementation  
    'org.springframework.cloud:spring-cloud-starter-aws:2.2.6.RELEASE'  
  
    //swagger  
    implementation group: 'io.springfox', name:  
    'springfox-boot-starter', version: '3.0.0'  
}
```

```

    testImplementation
'org.springframework.boot:spring-boot-starter-test'
    testImplementation
'org.springframework.security:spring-security-test'

    //email
    implementation 'org.springframework.boot:spring-boot-starter-mail'

    //socket
    implementation
'org.springframework.boot:spring-boot-starter-websocket'
    implementation group: 'org.webjars', name: 'stomp-websocket',
version: '2.3.3-1'
}

tasks.named('test') {
    useJUnitPlatform()
}

bootJar{
    archivesBaseName = 'cogether'
    archiveFileName = 'cogether'
    archiveVersion = "0.0.0"
}

```

- application-aws.properties

```

# aws
cloud.aws.credentials.access-key=AKIAU6JIIZHFLIKSCLP
cloud.aws.credentials.secret-key=ERpPix6SNho6JQk09qs9tVF7cz1JSNQnw/bzB
xyY
cloud.aws.s3.bucket=cogethera801
cloud.aws.region.static=ap-northeast-2
cloud.aws.stack.auto=false
cloud.aws.credentials.instance-profile=true

```

- application-db.properties

```

# Mysql ??
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# DB Source URL
spring.datasource.url=jdbc:mysql://i7a801.p.ssafy.io:3306/cogether?use
SSL=false&useUnicode=true&serverTimezone=Asia/Seoul

# DB username
spring.datasource.username=cogether

# DB password
spring.datasource.password=cogetherA801

# JPA
spring.jpa.show-sql=true

# DDL(create, alter, drop) ? ???? ??

```

```

spring.jpa.hibernate.ddl-auto=update

# JPA? ???? Hibernate? ????? ??? SQL? ???? ????.
spring.jpa.properties.hibernate.format_sql=true

#swagger
spring.mvc.pathmatch.matching-strategy=ant_path_matcher

# file upload max size (?? ??? ?? ??)
spring.servlet.multipart.max-file-size=20MB
spring.servlet.multipart.max-request-size=20MB

server.ssl.key-store-type= PKCS12
server.ssl.key-store-password= A801
server.ssl.key-store= keystore.p12

```

- application-email.properties

```

spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=cogether.service@gmail.com
spring.mail.password=gkyyedkudeozbpeb
spring.mail.properties.mail.smtp.socketFactory.class=javax.net.ssl.SSL
SocketFactory
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true

mail.smtp.auth=true
mail.smtp.starttls.required=true
mail.smtp.starttls.enable=true
mail.smtp.socketFactory.class=javax.net.ssl.SSLSocketFactory
mail.smtp.socketFactory.fallback=false
mail.smtp.port=465
mail.smtp.socketFactory.port=465
mail.username=cogether.service@gmail.com
mail.smtp.password=gkyyedkudeozbpeb
#admin ?? ??
AdminMail.id =cogether.service@gmail.com
AdminMail.password =gkyyedkudeozbpeb
#spring.mail.password=gkyyedkudeozbpeb

```