목차

1. 기술 스택
2. 빌드 상세내용
3. 배포 환경 세팅
4. 배포 명령어
5. 기술 스택
   1. 이슈관리: Jira
   2. 형상관리: Gitlab
   3. 커뮤니케이션: Matermost
   4. 개발환경
      1. OS : Windows 10
      2. IDE
         1. Spring Tool Suit 3.9.14
         2. Visual Studio Code 1.70.1
         3. UI/UX : Figma
      3. Database : MySQL Workbench 8.0.21
      4. Server : AWS EC2 (MobaXterm)
         1. Ubuntu 20.04.2
         2. Docker 20.10.17
   5. 상세사용
      1. Backend
         1. OpenJDK 1.8.0\_192 (Zulu 8.33.0.1)
         2. Spring Boot Gradle 7.5
         3. Kurento 6.15.0
         4. Lombok 1.18.24, Swagger2 3.0.0, QueryDsl 5.0.0
      2. Frontend
         1. HTML5, CSS3, JavaScript(ES6)
         2. Vue 3.2.13, Vuex 4.0.2
         3. 프론트 쓰고싶은거
6. 빌드 상세내용
   1. Frontend 빌드
      1. front 폴더로 이동

: package.json, jsconfig.json, vue.config.js 등 설정파일이 위치한 곳으로 이동

* + 1. Node\_modules를 위한 기본 install

|  |
| --- |
| npm install |

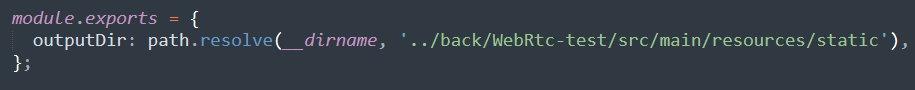
* + 1. 빌드하기

|  |
| --- |
| npm run build |

back/WebRtc-test/src/main/resources/static 폴더에 빌드 된 파일들이 생성됨

* + 1. 빌드 위치를 바꾸고싶다면…

vue.config.js 파일의 하단 부분을 수정하여 원하는 위치에 빌드



* 1. RestAPI 서버 빌드
     1. back/B310\_Back 폴더로 이동

: build.gradle, settings.gradle 등 gradle 설정파일이 위치한 곳으로 이동

* + 1. gradle 을 사용한 빌드

: gradle 설치가 되어있지 않다면 설치 후 빌드 진행

* + - 1. Windows 10 환경

|  |
| --- |
| gradlew build --exclude-task test |

* + - 1. ubuntu 환경

: gradle 권한 설정 후, 빌드 진행

|  |
| --- |
| chmod +x gradlew  ./gradlew build --exclude-task test |

* + 1. 빌드 파일 확인

back/B310\_Back/build/libs 위치에

B310\_Back-0.3.jar 파일 생성 확인

* 1. WebRTC 서버 빌드

: 빌드 된 front 파일이 WebRTC 서버 static 폴더에 저장되어 있어 이번 빌드 과정이 통합 빌드 처리가 됨

* + 1. back/WebRtc-test 폴더로 이동

: pom.xml 이 위치한 곳으로 이동

* + 1. maven 을 사용한 빌드

: maven 설치가 되어있지 않다면 설치 후 빌드 진행

|  |
| --- |
| mvn package |

* + 1. 빌드 파일 확인

back/WebRtc-test/target 위치에

WebRTC\_Server-6.15.0-exec.jar 파일 생성 확인

1. 배포 환경 설정
   1. Nginx 설정
      1. 기존 Nginx 설정 제거

|  |
| --- |
| sudo rm /etc/nginx/sites-enabled/default |

* + 1. Nginx 설정 파일 접근

|  |
| --- |
| sudo vi /etc/nginx/sites-available/default |

* + 1. 설정 값 다음과 같이 변경

|  |
| --- |
| # Default server configuration  #  server {  listen 80 default\_server;  listen [::]:80 default\_server;  root /var/www/html;  # Add index.php to the list if you are using PHP  index index.html index.htm index.nginx-debian.html;  server\_name \_;  location / {  # First attempt to serve request as file, then  # as directory, then fall back to displaying a 404.  try\_files $uri $uri/ =404;  }  }  server {  root /var/www/html;  # Add index.php to the list if you are using PHP  index index.html index.htm index.nginx-debian.html;  server\_name i7b310.p.ssafy.io; # managed by Certbot  location / {  # First attempt to serve request as file, then  # as directory, then fall back to displaying a 404.  proxy\_pass https://localhost:8443;  proxy\_set\_header Origin "";  proxy\_http\_version 1.1;  #proxy\_set\_header Upgrade $http\_upgrade;  #proxy\_set\_header Connection $connection\_upgrade;  proxy\_set\_header Upgrade "websocket";  proxy\_set\_header Connection "Upgrade";  proxy\_read\_timeout 120000; # 2 \* 60 \* 1000  proxy\_send\_timeout 120000; # 2 \* 60 \* 1000  proxy\_set\_header X-Real-IP $remote\_addr;  proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  proxy\_set\_header Host $http\_host;  #try\_files $uri $uri/ =404;  }  listen [::]:443 ssl ipv6only=on; # managed by Certbot  listen 443 ssl; # managed by Certbot  ssl\_certificate /etc/letsencrypt/live/i7b310.p.ssafy.io/fullchain.pem; # managed by Certbot  ssl\_certificate\_key /etc/letsencrypt/live/i7b310.p.ssafy.io/privkey.pem; # managed by Certbot  include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot  ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot  }  server {  if ($host = i7b310.p.ssafy.io) {  return 301 https://$host$request\_uri;  } # managed by Certbot  listen 80 ;  listen [::]:80 ;  server\_name i7b310.p.ssafy.io;  return 404; # managed by Certbot  } |

* + 1. 작성한 파일 적용

: sites-enabled 에 같은 파일이 옮겨지고 nginx 를 재실행 하면 설정이 적용됨

|  |
| --- |
| sudo ln -s /etc/nginx/sites-available/default /etc/nginx/sites-enabled/ |

* + 1. nginx 재실행

|  |
| --- |
| sudo service nginx restart |

* 1. DB 설정
     1. MySQL 접속

: MySQL이 설치되지않았다면 설치 후 진행

|  |
| --- |
| sudo mysql -u root -p |

\* 비밀번호 입력의 경우 초기에는 없거나 ‘root’

* + 1. 로컬 계정 생성

: 설치한 곳에서만 사용 가능한 계정

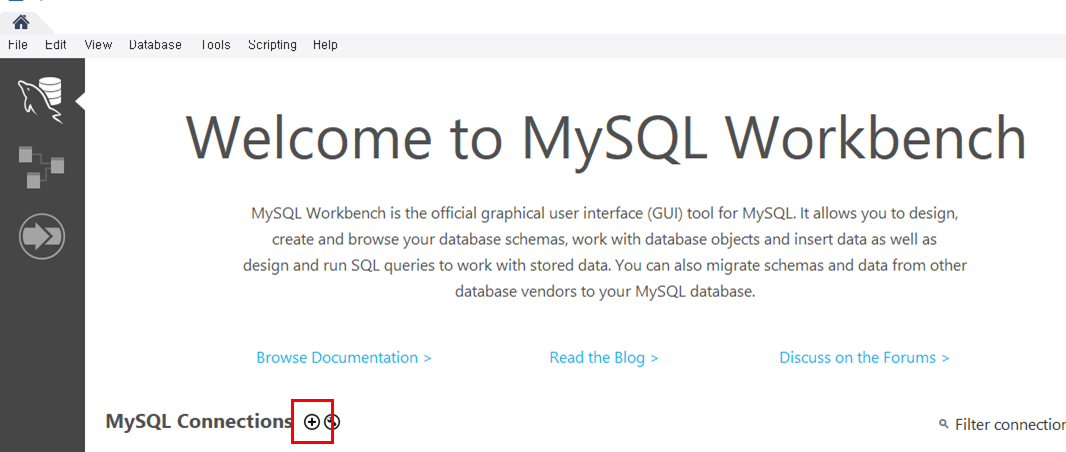
|  |
| --- |
| create user ‘ssafy’@’localhost’ identified by ‘ssafy’; |

* + 1. 외부 접근 계정 생성

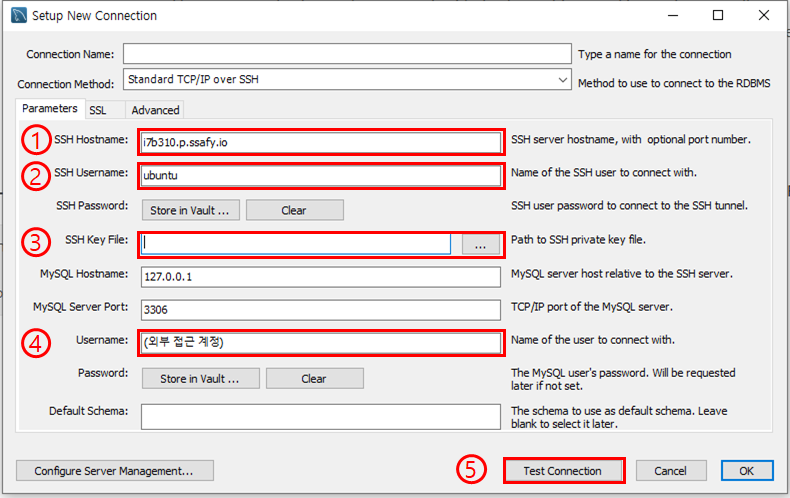
: 외부에서 DB를 관리할 때 사용할 계정

|  |
| --- |
| create user ‘(사용할 계정)’@’%’ identified by ‘(사용할 비밀번호)’; |

* + 1. Workbench로 서버 DB 연결
       1. 새 연결 추가 버튼 클릭

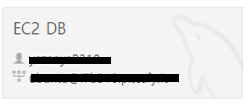


* + - 1. 연결에 필요한 값 입력



1. SSH Hostname 입력 (EC2 주소 @ 뒷부분)
2. SSH Username 입력 (EC2 주소 @ 앞부분)
3. SSH Key File 파일 경로 입력 (EC2 의 .pem 키 위치)
4. Username 입력 (생성한 외부 접근 계정)
5. Test Connection 클릭 후 접속 확인
6. OK 눌러 등록
   * + 1. 새로운 연결 등록 확인

아래와 같은 연결 블록이 생겼다면 성공



* + 1. 기본 더미 데이터 입력

더미데이터 SQL 실행

* 1. 기본 더미 데이터 파일 입력

/home/ubuntu/ 위치에 더미 폴더가 저장된 img 폴더 옮기기

* 1. Kurento Media Server
     1. Docker 설치

|  |
| --- |
| curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg |

|  |
| --- |
| echo \  "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \  $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null |

|  |
| --- |
| sudo apt-get update  sudo apt-get install  sudo apt-get install docker-ce docker-ce-cli containerd.io |

|  |
| --- |
| sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose |

|  |
| --- |
| sudo chmod +x /usr/local/bin/docker-compose |

* + 1. Kurento Media Server 이미지 받기

|  |
| --- |
| sudo docker pull kurento/kurento-media-server:latest |

* + 1. Kurento Media Server 컨테이너 실행

: kms 라는 이름으로 기본 설정포트(8888) 로 실행

|  |
| --- |
| sudo docker run -d --name kms --network host \  kurento/kurento-media-server:latest |

* + 1. Coturn Server 설치

: WebRTC 시그널링을 위한 STUN/TURN 서버

* + - 1. 설치

|  |
| --- |
| sudo docker run -d --name kms --network host \  kurento/kurento-media-server:latest |

* + - 1. 설정
         1. /etc/default/coturn 파일 수정

|  |
| --- |
| TURNSERVER\_ENABLED=1 |

* + - * 1. /etc/turnserver.conf 파일 수정

|  |
| --- |
| listening-port=3478  tls-listening-port=5349  listening-ip=(EC2 Private IPv4)  external-ip=(EC2 Public IPv4)/(EC2 Private IPv4)  relay-ip=(EC2 Private IPv4)  fingerprint  lt-cred-mech  user=myuser:mypassword  realm=myrealm  log-file=/var/log/turn.log  simple-log |
|  |

* + - 1. Coturn 재 실행

|  |
| --- |
| sudo service coturn restart |

1. 배포 명령어
   1. EC2에 빌드한 2개의 jar 파일 업로드
   2. 배포 전 구동 중인 서버 확인

: 서버의 포트는 8081, 8443 으로 설정되어있음

* + 1. 8081, 8443 포트로 실행중인 서버 PID 확인

|  |
| --- |
| netstat -ntlp | grep :8081  netstat -ntlp | grep :8443 |

* + 1. 실행 중인 PID 가 있을 경우 종료

|  |
| --- |
| sudo kill <PID> |

* 1. 서버 실행

|  |
| --- |
| nohup java -jar (실행할 빌드 파일.jar) & |