포팅 메뉴얼

Backend

- JAVA 11
- Spring 2.7.14
- Docker : 24.0.5
- OpenVidu Deployment : 2.28.0
  - OPENVIDU 환경변수
    - OPENVIDU_URL : https://i9c203.p.ssafy.io
    - OPENVIDU_SECRET : openvidureonc203
- 인텔리제이 : 2023.1.13
- **Backend 환경변수 (Gradle)**
  - application.yml

```
spring:
  servlet:
    multipart:
      max-file-size: 50MB
      max-request-size: 50MB

  cloud:
    gcp:
      storage:
        credentials:
          location: classpath:ninth-botany-395113-5f24f3555342.json
        project-id: 5f24f35553425ba271e94c142c197a698e90f0dc
        bucket: reon-bucket

  datasource:
    driver-class-name: org.mariadb.jdbc.Driver
    url: jdbc:mariadb://stg-yswa-kr-practice-db-master.mariadb.database.azure.com:3306/S09P12C203?useUnicode=true&characterEncodir
    username: "S09P12C203"
    password: "3iNtyHIJCM"


  mvc:
    pathmatch:
      matching-strategy: ant_path_matcher

  jpa:
    hibernate:
      ddl-auto : create
      show-sql: true # DDL 출력
    generate-ddl: true
    properties:
      hibernate:
        show_sql: true
        format_sql: true

OPENVIDU_URL:
  https://i9c203.p.ssafy.io
OPENVIDU_SECRET:
  openvidureonc203


oauth:
  naver:
    secret: WRf9Tvwu7z
    client-id: EJLfPmAn1c9dXfThcDEl
    url:
      auth: https://nid.naver.com
      api: https://openapi.naver.com
jwt:
  secret-key: VvbHNpyb3JodG9kZbWhKZ29nby10Atam9vbmdhbmpaGFsZ2VveW8duaW0teWbS1zZXJ2ZXItZGxyamVvYW9290c3
```

- build.gradle

```
// QueryDSL
buildscript {
  ext {
    queryDslVersion = "5.0.0"
  }
}

plugins {
  id 'java'
  id 'org.springframework.boot' version '2.7.14'
  id 'io.spring.dependency-management' version '1.0.15.RELEASE'
  id "com.ewerk.gradle.plugins.querydsl" version "1.0.10"
}

  group = 'reon'
  version = '0.0.1-SNAPSHOT'

java {
  sourceCompatibility = '11'
}

configurations {
  compileOnly {
    extendsFrom annotationProcessor
  }
}

repositories {
  mavenCentral()
}

dependencies {
  //jpa 의존성
  implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
  //ouath 의존성
  implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'
  //security
  implementation 'org.springframework.boot:spring-boot-starter-security'
  // validation
  implementation 'org.springframework.boot:spring-boot-starter-validation'
  // 기본 web
  implementation 'org.springframework.boot:spring-boot-starter-web'
  //swagger
  implementation 'io.springfox:springfox-boot-starter:3.0.0'
  // jwt 관련
  implementation 'io.jsonwebtoken:jjwt-api:0.11.5'
  implementation 'io.jsonwebtoken:jjwt-impl:0.11.5'
  implementation 'io.jsonwebtoken:jjwt-jackson:0.11.5'
  //querydsl 추가
  implementation "com.querydsl:querydsl-jpa:${queryDslVersion}"
  annotationProcessor "com.querydsl:querydsl-apt:${queryDslVersion}"
  // 스토리지
  implementation group: 'org.springframework.cloud', name: 'spring-cloud-gcp-starter', version: '1.2.5.RELEASE'
  implementation group: 'org.springframework.cloud', name: 'spring-cloud-gcp-storage', version: '1.2.5.RELEASE'
  // web
  compileOnly 'org.projectlombok:lombok'
  runtimeOnly 'com.mysql:mysql-connector-j'
  annotationProcessor 'org.projectlombok:lombok'

  //객체 ToString을 위한 @ToStringBuilder
  implementation 'org.apache.commons:commons-lang3:3.12.0'

  //mariadb 의존성
  implementation group: 'org.mariadb.jdbc', name: 'mariadb-java-client'
  // openvidu
  implementation group: 'io.openvidu', name: 'openvidu-java-client', version: '2.28.0'

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

}

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

//queryDsl 추가
def querydslDir = "$buildDir/generated/querydsl"

querydsl {
  jpa = true
  querydslSourcesDir = querydslDir
}
```

```
sourceSets {
  main.java.srcDir querydslDir
}

configurations {
  compileOnly {
    extendsFrom annotationProcessor
  }
  querydsl.extendsFrom compileClasspath
}

compileQuerydsl {
  options.annotationProcessorPath = configurations.querydsl
}
```

Front

- Node.js : 18.16.1
  - `start: react-scripts –openssl-legacy provider start`
  - `build: react-scripts --openssl-legacy-provider build`
- NPM : 9.5.1
- create-react-app : 5.0.1
- React : 18.2.0
- Redux : 8.1.2
  - Redux-Toolkit : 1.9.5
- React-Router-Dom : 6.14.2
- OpenVidu-browser : 2.28.0
- Tailwind CSS: 3.3.3
- axios : 1.4.0
- ONNX Runtime Web : 1.15.1
- face-api.js : 0.22.2
- react-speech-recognition : 3.10.0
- Front 환경 변수
  - .env

```
REACT_APP_NAVER_CLIENT_ID=EJLfPmAn1c9dXfThcDEl
REACT_APP_REDIRECT_URI=https://i9c203.p.ssafy.io/login/redirect
REACT_APP_API=https://i9c203.p.ssafy.io
REACT_APP_STATE=jinsik
REACT_APP_ALTER_IMG_URL=https://storage.googleapis.com/reon-bucket/LoginDefaultImg.png
```

배포시 특이사항

EC2

- Frontend, Backend 수동 배포
  - 프로젝트 필드 후 Docker hub에 push
  - 서버 접속 후 이미지 pull, 실행
- Backend docker container는 8080port로 띄우기
- Frontend docker container는 3000 port로 띄우기
- /opt/openvidu/custom-nginx.conf에 location / 의 경우 3000번 포트로 연결

```
location / {
        #proxy_pass http://yourapp; # Openvidu call by default
        proxy_pass http://localhost:3000;

    }
```

- /opt/openvidu/custom-nginx.conf에 location /api 의 경우 8080번 포트로 연결

```
location /api {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        # try_files $uri $uri/ =404;
        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;
        proxy_set_header    X-NginX-Proxy true;
        proxy_pass http://localhost:8080;
        proxy_redirect  off;
        charset utf-8;
}
```

- 서버 OpenVidu Deployment 환경 변수 설정 필수
  - cd /opt/openvidu/.env에 오픈비두 환경 변수 설정
    - Openvidu에 설정된 변수와 백엔드에 설정된 변수가 동일해야함
    - OPENVIDU_URL : https://i9c203.p.ssafy.io
    - OPENVIDU_SECRET : openvidureonc203
- OpenVidu Deployment를 실행 후 Backend Container 실행 필요
  - Openvidu Deployment 재실행 시 Backend 재실행 필요

외부 서비스 정보 문서

- Openvidu
  - https://docs.openvidu.io/en/2.28.0/deployment/ce/on-premises/
- 네이버 로그인
  - REON 서비스는 네이버 로그인 후 이용 가능
  - naver_developers : https://developers.naver.com/main/
  - naver developers reon application에 등록된 사람(개발자)만 이용 가능
    - 네이버 승인 후 서비스 모드로 넘어가게 되면 모두 이용 가능
    - 연기 배틀의 기능의 경우 상대방과의 매칭은 랜덤으로 이루어짐
    - 최종 발표 시연을 위해 naver develops Applicaiton을 개발 모드로 설정

DB 덤프 파일 최신본

시연 시나리오

- 로그인
  - 오른쪽 상단 로그인 버튼을 눌러 네이버 소셜 로그인 진행.
  - 모든 서비스는 로그인을 해야 이용 가능
- 혼자하기
  - 혼자 연기 연습을 하는 곳으로 상단 혼자하기 버튼을 눌러 진행 가능
  - 영화 변경 버튼을 눌러 연습할 영상 변경 가능

    - 게임 시작 버튼을 눌러서 연기 연습 진행
    - 가운데 하단부분에 연기 영상의 대본이 주어진다.
    - 실시간으로 채점을 진행하여 연기 점수를 알려준다.
- 같이하기
    - 상단 같이하기 버튼을 눌러 벡스테이지에 입장한다.
    - 벡스테이지
        - 연기 배틀을 진행하기 전 대기실
        - 본인의 유저 정보, 전적 등을 볼 수 있다.
        - 최근 전적 10개를 볼 수 있다.
        - 현재 서비스에 가입된 유저중 랭크 Top5을 볼 수 있다.
        - 오른쪽 아래 튜토리얼 버튼을 눌러 연기 베틀 튜토리얼을 볼 수 있다.
        - 오른쪽 아래 게임 시작 버튼을 눌러 1대1 연기 베틀을 진행할 수 있다.
    - 배틀룸
        - 실제 배틀이 이루어 지는 곳
        - 본인 화면, 연기할 영상, 상대 영상의 순서로 화면이 보여진다.
        - 처음에 연기할 영상이 재생되고 랜덤한 차례로 연기가 진행된다.
            - 차례인 유저 화면에는 연기할 차례라고 표시된다.
            - 자신의 차례에 연기를 진행
            - 가운데 주어진 대본과 영상을 보고 연기를 진행한다.
        - 연기가 종료되면 AI가 자동으로 채점을 진행하고 결과를 알려준다
        - 자신의 연기 영상을 저장하고 싶다면 저장하기 버튼을 누른다
        - 게임 종료후 자동으로 백스테이지로 이동하게 된다.
        - 주의사항
            - 세션 입장 퇴장시 꼭 텀을 두어야한다.
            - 만약에 끌 수 없는 빨간 에러면 새로고침 후 세션 정리될 때까지 대기
- 마이페이지
    - 해당 유저의 정보, 연기 영상들을 볼 수 있는 페이지
    - 유저의 게시글이나 댓글에서 닉네임을 클릭하여 이동하고 자신의 페이지는 로그인 후 오른쪽 상단 마이 페이지를 눌러 이동이 가능하다.
    - 유저의 닉네임, 티어, 이메일, 자기소개, 투표해줘 게시판에 올린 영상들, 배틀 영상을 저장한 비공개, 내가 좋아요 누른 영상들 순서로 조회가 가능하다.
    - 자신의 페이지의 경우 수정 버튼을 눌러 정보 수정이 가능하다.
    - 비공개 메뉴의 경우 배틀 후 저장한 영상목록이 주어진다.
        - 사람들에게 평가받거나 공유하고 싶은 연기 영상의 경우 업로드할 수 있다.
            - 해당 영상 선택 후 제목과 내용을 입력하고 업로드해야한다.
        - 업로드 한 영상은 투표해줘 게시판에 보이게 되고 게시물 메뉴로 이동하게 된다.
    - 게시물 메뉴의 경우 해당 유저가 투표해줘 게시판에 올린 게시물 목록을 볼 수 있다.
    - 좋아요 메뉴의 경우 해당 유저가 좋아요를 누른 게시글 목록을 볼 수 있다.