



4월 4주차 회의록

팀	MAC
참석자	최필환(한) 한결 민(상) 상연 안
장소	원흥관 3층 아리수
시간	2023년 4월 30일 12:00~18:00

팀 주제

동국대학교 E-class 내 파일 스토리지 기능 구현 → 협업 능력 향상

주요 안건

- 간트 차트 작성
- 진행 사항 파악
- 코드 리뷰 후 머지
- 통합 테스트

회의 내용

1. 진행도 파악

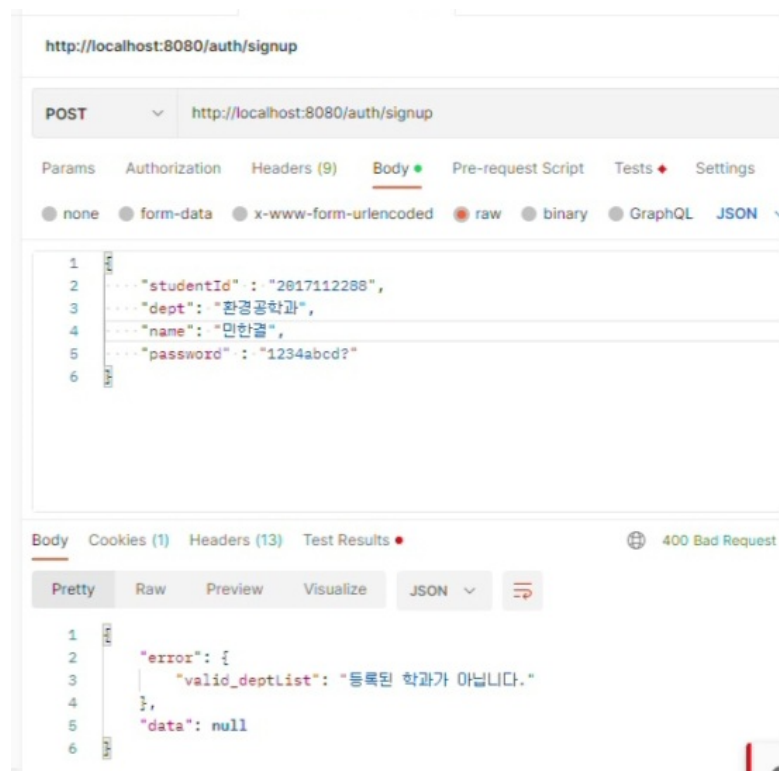
분류	내용	담당	4월4주	5주	5월1주	2주	3주	4주	5월5주	6월1주	6월2주
웹컴 페이지	로그인 기능 구현	민한결									
	프론트 템플릿 수정	안상연									
파일 스토리지	AWS 설정/파일 업로드 구현	최필환									
팀 활동 페이지	팀 구성 기능 구현	민한결									
웹컴 페이지	로그인 요청 및 응답 처리	안상연									
파일 스토리지	버전별 파일 관리/다운로드	최필환									
팀 활동 페이지	공지사항 기능 구현	민한결									
	팀 구성 요청 및 응답 처리	안상연									
	파일 업로드, 다운로드, 삭제	최필환									
	예외 처리, 배포	민한결									
	파일 스토리지 요청/응답	안상연									
	통합 테스트	최필환									
최종 수정 및 발표		모 두									

2. 진행 사항 파악

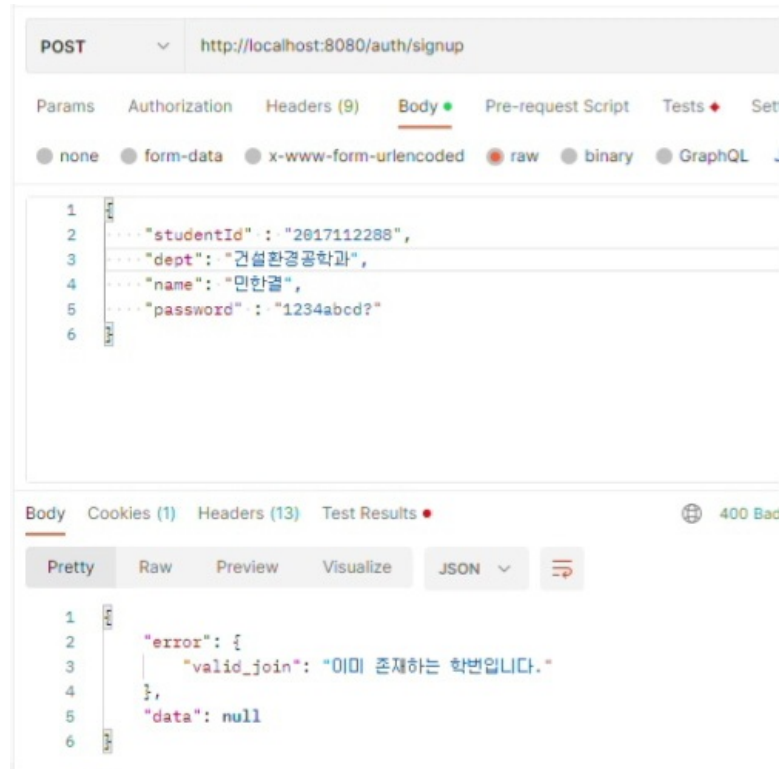
1. 민한결(백엔드: 회원가입, 로그인 기능 구현)

1-1. 회원가입

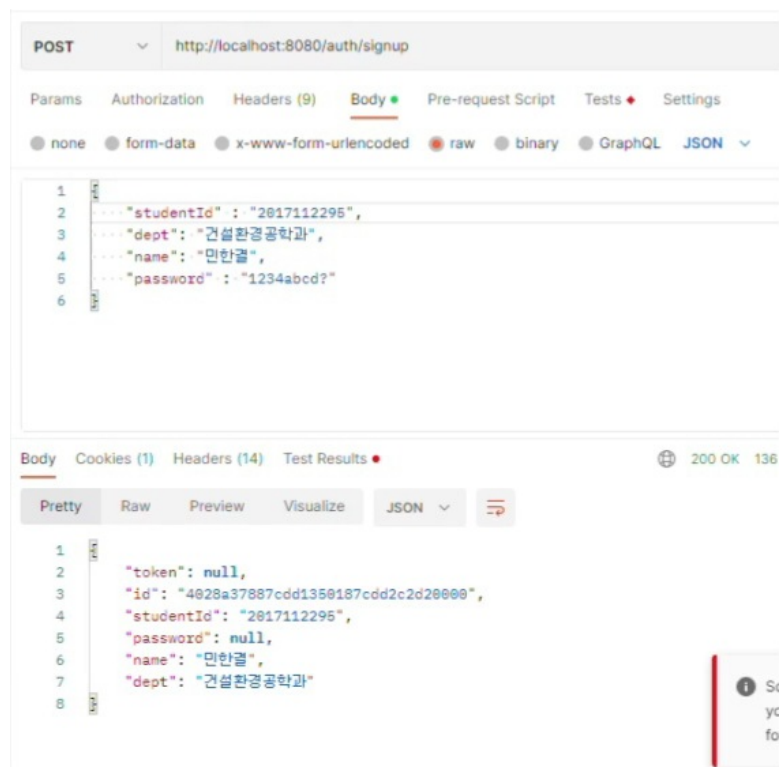
- a. Spring Validation 사용 유효성 검증 → userJoinService의 validationHandling 로 예외 발생시 메시지를 ResponseDto의 Error에 담아 client로 전달



- b. 가입 데이터가 유효한지, 학번이 이미 존재하는지 검증

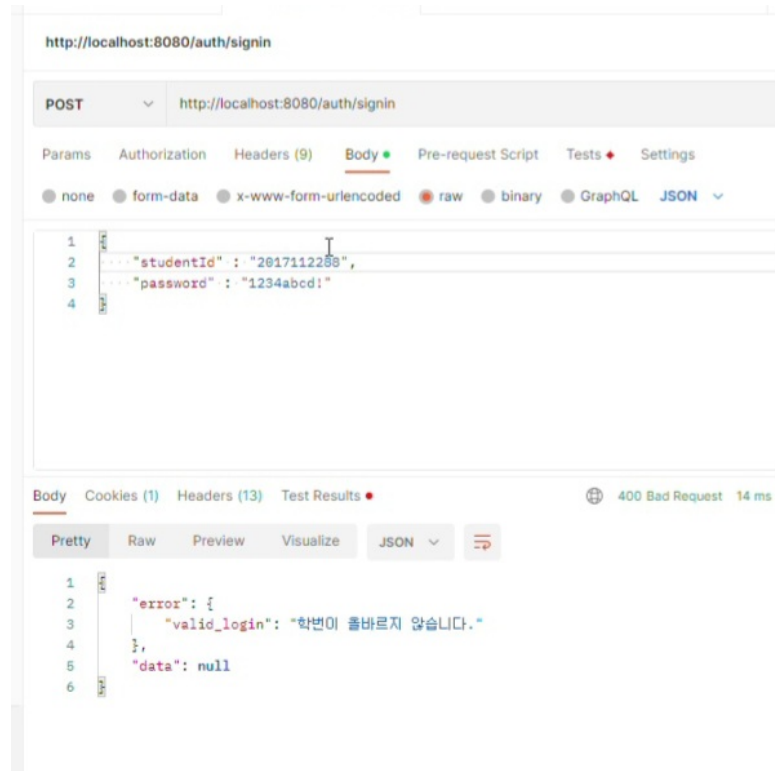


- c. 유효한 가입 요청인 경우 Spring Security 라이브러리 BcryptPasswordEncoder의 encode 메서드 이용 SHA-2 이상의 알고리즘, 랜덤하게 생성된 솔트로 password 암호화해서 저장하고 client에 http 200 응답코드와 가입 결과 반환

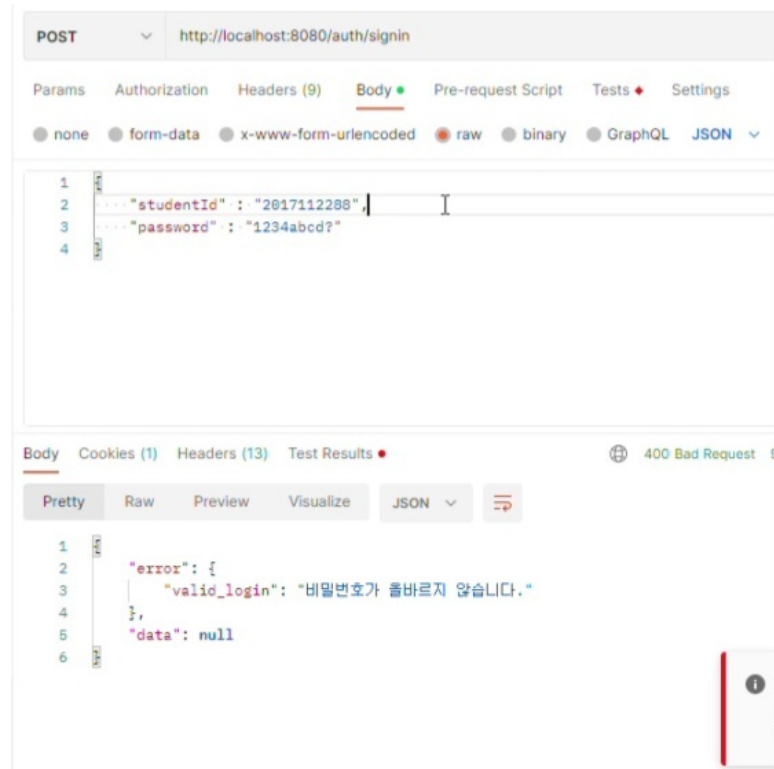


1-2. 로그인

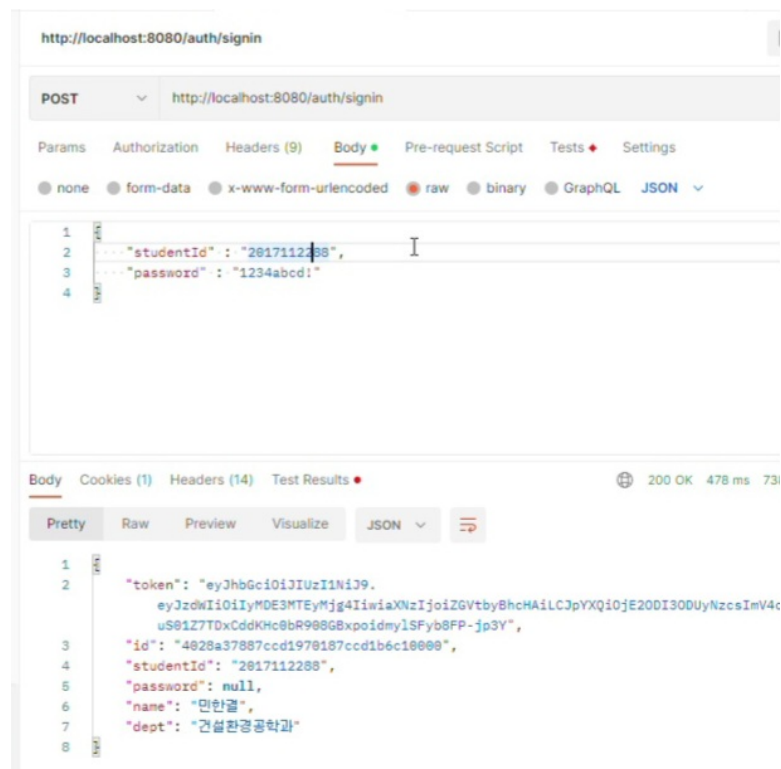
a. 학번이 올바른지 검증



b. 학번이 올바를 때 비밀번호가 올바른지 검증



- c. 유효한 로그인 요청이면 토큰 발급하여 userDto에 담아 build하고 client에 http 200 응답코드와 함께 전달



1-3. JWT 토큰 발급

- a. Spring jjwt 라이브러리 사용 → application.yml의 secretKey 통해 studentId와 expiration 정보 담은 JWT 토큰 생성(application.yml은 .gitignore로 비공개)

```
@Slf4j
@Service
public class TokenProvider {
    // application.yml에 설정한 jwt.secretKey를 가져옴
    @Value("${jwt.secret}")
    private String secretKey;

    // 토큰 생성
    public String create(User user) {
        Date now = new Date(); // 현재 시간
        Date validity = new Date(now.getTime() + 1000 * 60 * 30); // 30분 뒤 만료
        return Jwts.builder()
            .signWith(SignatureAlgorithm.HS256, secretKey) // HS256 알고리즘, secretKey로 서명
            .setSubject(user.getStudentId())
            .setIssuer("demo app")
            .setIssuedAt(now)
            .setExpiration(validity)
            .compact();
    }
}
```

- b. 토큰 유효성, 토큰 만료 여부 검증 로직 구현
- c. request에서 토큰 파싱하여 검증하는 JwtAuthenticationFilter 구현
- d. WebSecurityConfig에서 메인 페이지, 로그인, 회원가입 api 토큰 검증 예외 처리, 서버에 인증정보 저장하지 않으므로 csrf disable

```
http.cors()//cors 허용
    .and()
    .csrf().disable() //csrf 토큰 비활성화
    .httpBasic().disable() //기본 설정 사용 안함
    .sessionManagement().sessionCreationPolicy(Session
    .and()
    .authorizeRequests().antMatchers("/", "/auth/**",
    .anyRequest().authenticated(); //그 외의 url은 인증 필
http.addFilterAfter(jwtAuthenticationFilter, CorsFilter.cl
```

- e. 이외 api 요청시 마다 토큰 통해 로그인 검증

1-4. 프론트 통합 테스트

로그인 기능 테스트시 Cross Origin Error 해결 위해 webConfig에서 react 로컬 서버 cors 요청 허용

```
package com.test.userTest.domain.user.service.config;

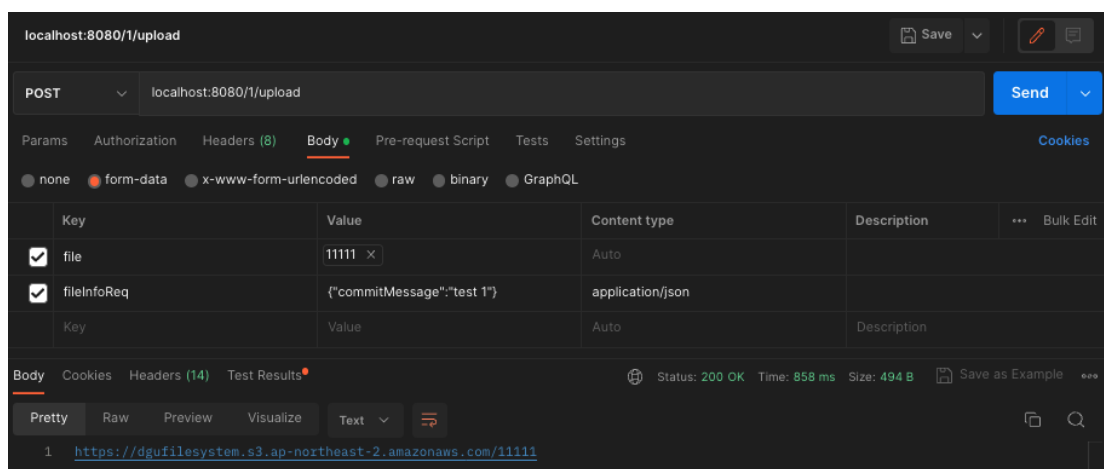
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
@EnableWebMvc
public class WebMvcConfig implements WebMvcConfigurer {

    @Override
    public void addCorsMappings(CorsRegistry registry) {
        registry.addMapping(pathPattern: "**")
            .allowedOrigins("http://localhost:3000")
            .allowedMethods("POST");
    }
}
```

2. 최필환(백엔드: 파일 업로드 기능 구현)

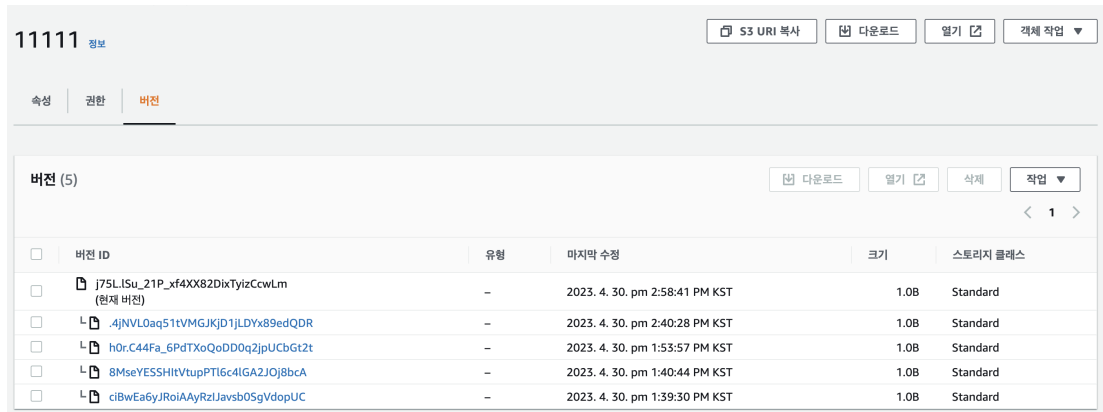
a. POSTMAN을 통해 file 업로드



- form-data를 이용하여 MultipartFile 타입의 파일과, application/json 형태로 file업로드시 추가정보(commit_message)를 POST 방식으로 전송하면 url을

응답받을 수 있다.

b. S3 파일 관리

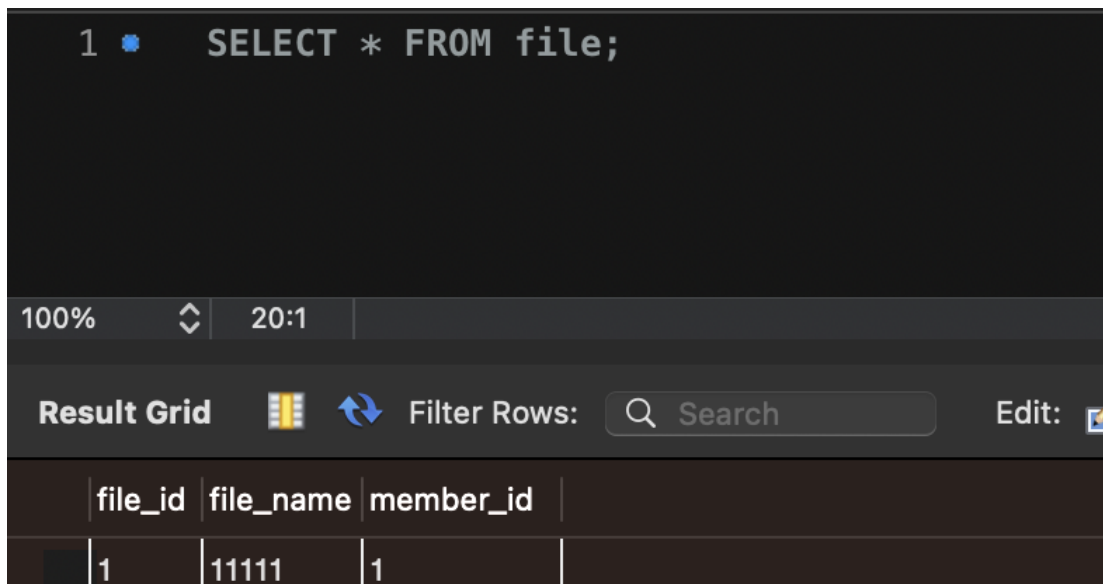


The screenshot shows the AWS S3 console interface for a bucket named '11111'. It displays a list of file versions under the '버전 (5)' tab. The table includes columns for version ID, type, last modified date, size, and storage class.

버전 ID	유형	마지막 수정	크기	스토리지 클래스
j75LlSu_21P_xf4XX82DixTyzCcwLm (현재 버전)	-	2023. 4. 30. pm 2:58:41 PM KST	1.0B	Standard
.4jNVL0aq51tVMGJKjD1jLDYx89edQDR	-	2023. 4. 30. pm 2:40:28 PM KST	1.0B	Standard
h0r.C44Fa_6PdTXoQoDD0q2jpUCbGt2t	-	2023. 4. 30. pm 1:53:57 PM KST	1.0B	Standard
8MseYESSHtVtupPTI6c4lGA2JOj8bcA	-	2023. 4. 30. pm 1:40:44 PM KST	1.0B	Standard
ciBwEa6yJRoiAAyRzJavsb05gVdopUC	-	2023. 4. 30. pm 1:39:30 PM KST	1.0B	Standard

- S3의 버킷 버전 관리를 통해 이름.확장자 가 같은 파일이 업로드 될시, 위의 사진과 같이 최신순으로 파일이 저장된다.

c. 로컬 DB 파일 저장

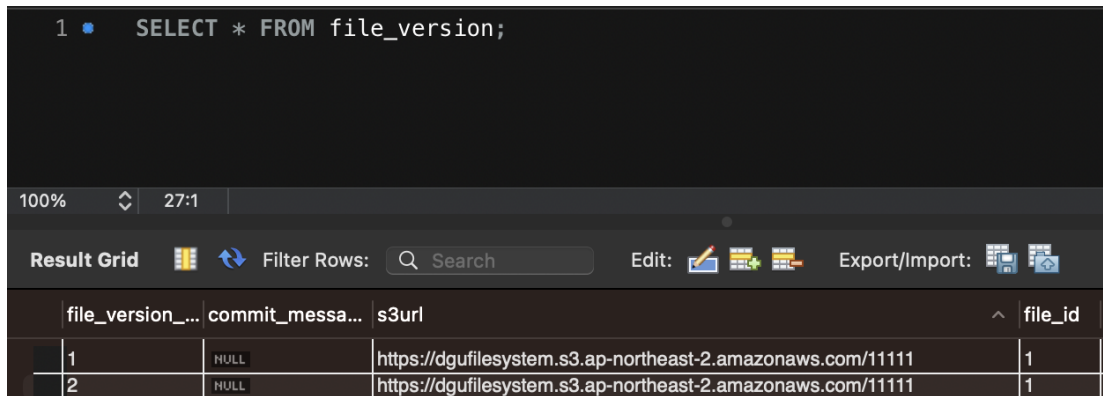


The screenshot shows a SQL query 'SELECT * FROM file;' being executed in a database client. The result is displayed in a grid format with columns: file_id, file_name, and member_id. The data shows a single row with file_id 1, file_name 11111, and member_id 1.

file_id	file_name	member_id
1	11111	1

- 위와 같이 member_id가 1인 객체를 만들어 놓고, file을 저장했을때 db 테이블 목록이다.
- S3에 올린 file_name과 동일한 것으로 보아, 같은 파일임을 알 수 있다.

d. 로컬 DB 파일 버전 관리



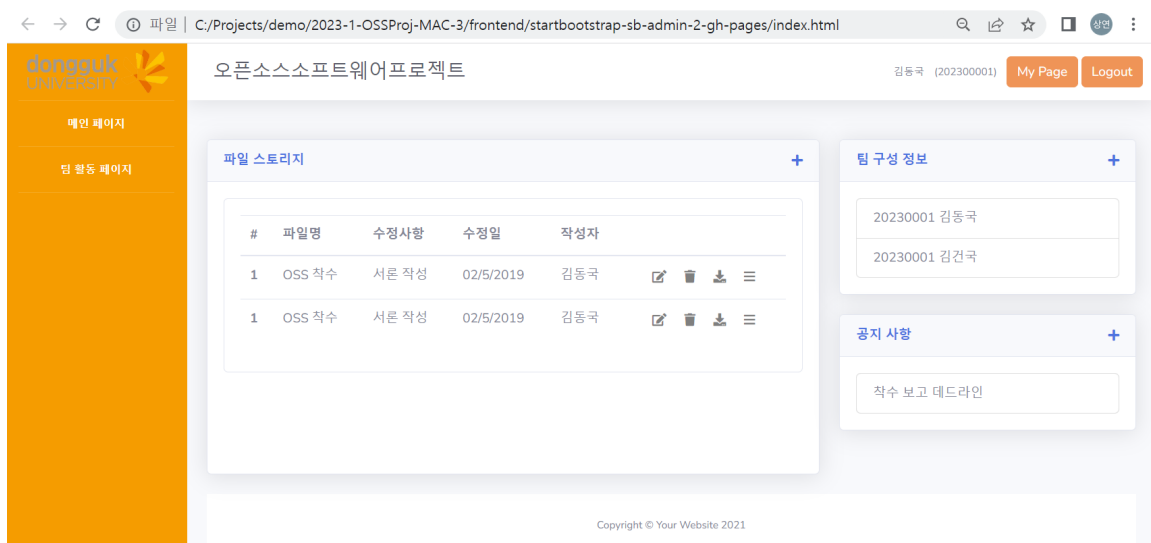
The screenshot shows a database query result for the 'file_version' table. The query is 'SELECT * FROM file_version;'. The result grid displays two rows of data. The columns are 'file_version_id', 'commit_message', 's3url', and 'file_id'.

file_version_id	commit_message	s3url	file_id
1	NULL	https://dgufilesystem.s3.ap-northeast-2.amazonaws.com/11111	1
2	NULL	https://dgufilesystem.s3.ap-northeast-2.amazonaws.com/11111	1

- file 업로드시, 업로드 파일과 같은 이름의 파일이 있다면, 버전관리를 위해 추가적으로 file을 생성하는 것이 아닌, file_version 테이블에 commit message와 url을 저장하여 버전 관리
- 현재 commit_message가 누락되는 문제와 s3url이 첫 upload할때의 url로 저장되는 문제가 발견됨.
- 따라서 commit_message 저장과 개별 버전의 s3url 저장을 수정 및 구현할 예정.

3. 안상연(프론트엔드: 팀활동 페이지 구현)

a. 팀 활동 페이지 구현



- 템플릿 오픈소스 코드 수정하여 팀 활동 페이지 구현

b. 로그인 화면 리액트 환경으로 변경 후 서버 연동 테스트 진행

- 리액트 환경 구축 후, 로그인 페이지인 login.html 코드를 리액트 코드로 변환

```
fetch("http://localhost:8080/auth/signin", {
  method: "POST",
  headers: {
    "Content-Type": "application/json",
  },
  body: JSON.stringify(data),
})
  .then((response) => response.json())
  .then((data) => console.log(data))
  .catch((error) => console.error(error));
```

- fetch 함수를 사용하여 http://localhost:8080/auth/signin 주소로 POST 요청을 보냄

```
{ } package.json M ×
reactTest > demo > { } package.json > { } browserslist > [ ] deve
35      "last 1 safari version"
36    ]
37  },
38  "proxy": "http://localhost:8080/"
39  }
40
```

- CORS(Cross-origin-resource-sharing) 에러를 리액트 환경 내 package.json 수정(+IntelliJ 코드 수정)으로 해결
- 결과

최필환: 버전별 url 관리, 다운로드 기능 구현