데이터베이스 프로젝트

중간보고서

(팀 12)

제출일: 2020.11.15

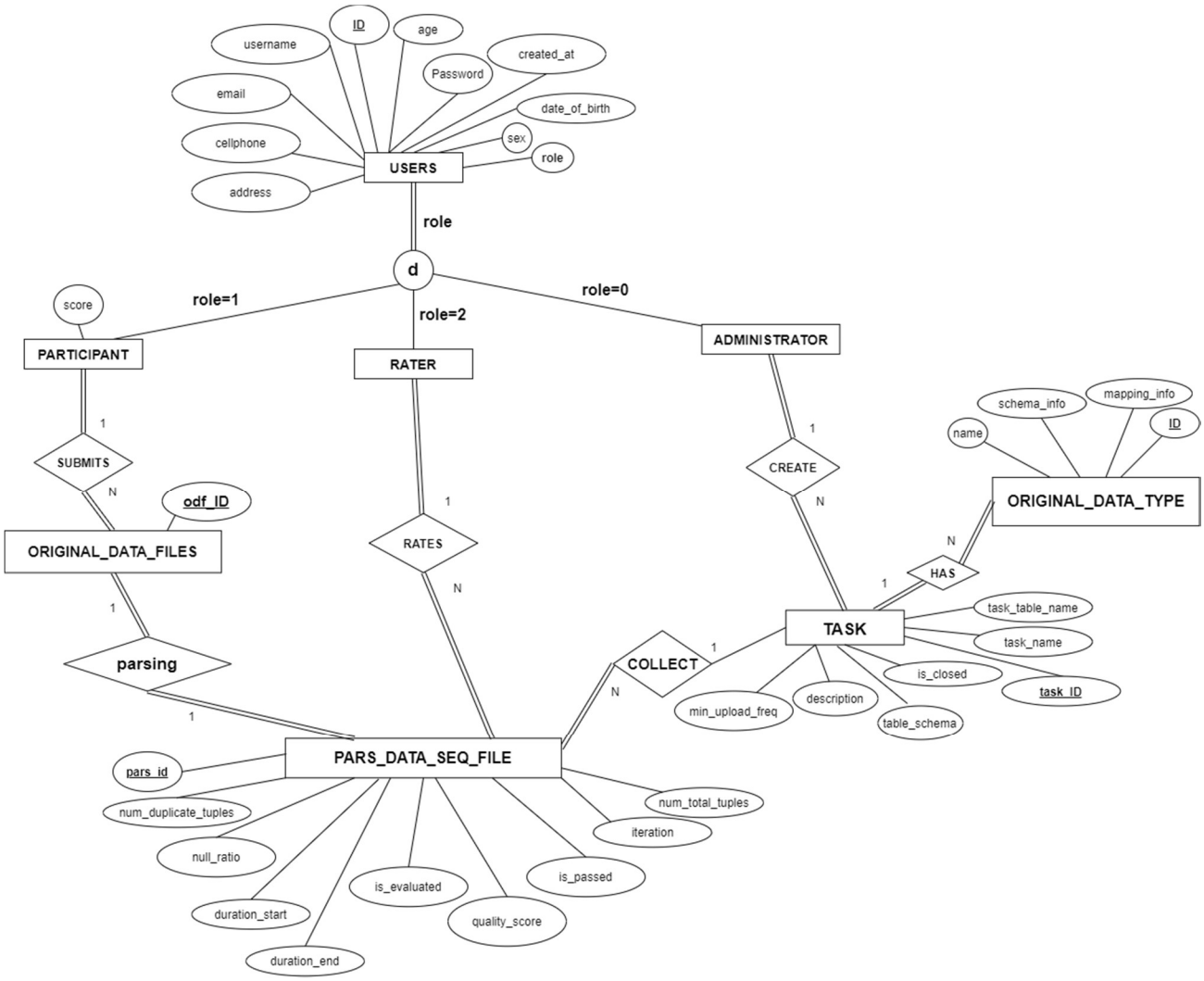
서브 1 팀: 권호정 / 곽동우 / 마준영

서브 2 팀: CUI,LIN / HAN,YAXI / ZHAN,XITONG

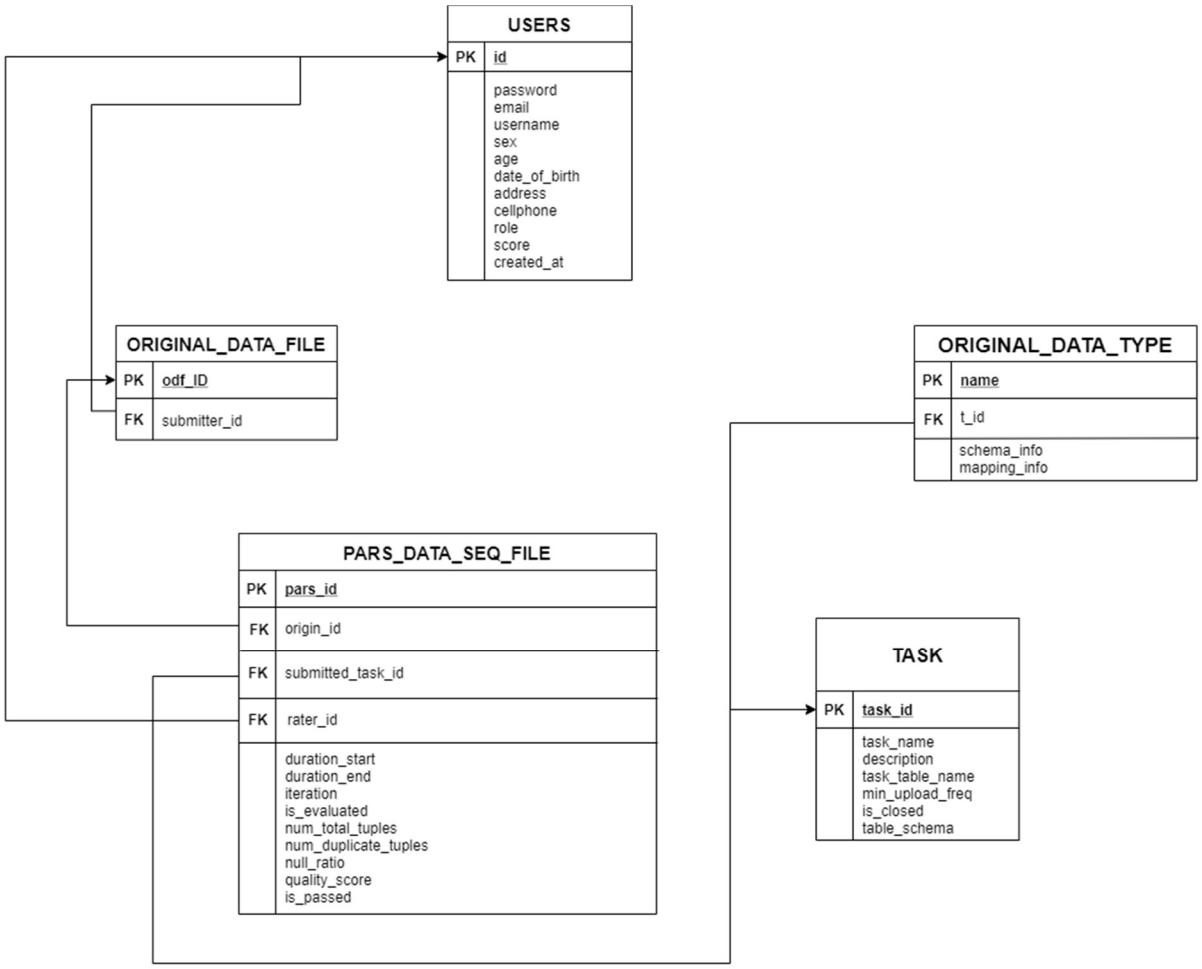
1. ER-Diagram & Mapping ERD to Relational Schema

ERD 작성은 서브 팀 별로 주제별로 먼저 작성한 뒤, 전체회의를 통해 수정 및 보완하여 최종 ERD 를 완성하였다. 최종 ERD 를 가지고 relational schema 로 mapping 하는 과정을 거쳤다. ERD 의 relational schema 로의 mapping 은 서브 1 팀(마준영)과 서브 2 팀(CUI,LIN)의 데이터베이스 담당자가 회의하여 작성하였다. ERD 와 ERD 의 relational schema 로의 mapping 결과는 아래와 같다.

<ERD>



<Mapping ERD to Relational Schema>



2. 역할 분담 및 일정

개발명세에 따라 세부 구현일정을 정하였으며, 프론트엔드/백엔드/데이터베이스 파트별로 팀원을 나누어 진행할 계획이다. 구체적인 역할 분담 및 일정은 아래와 같다.

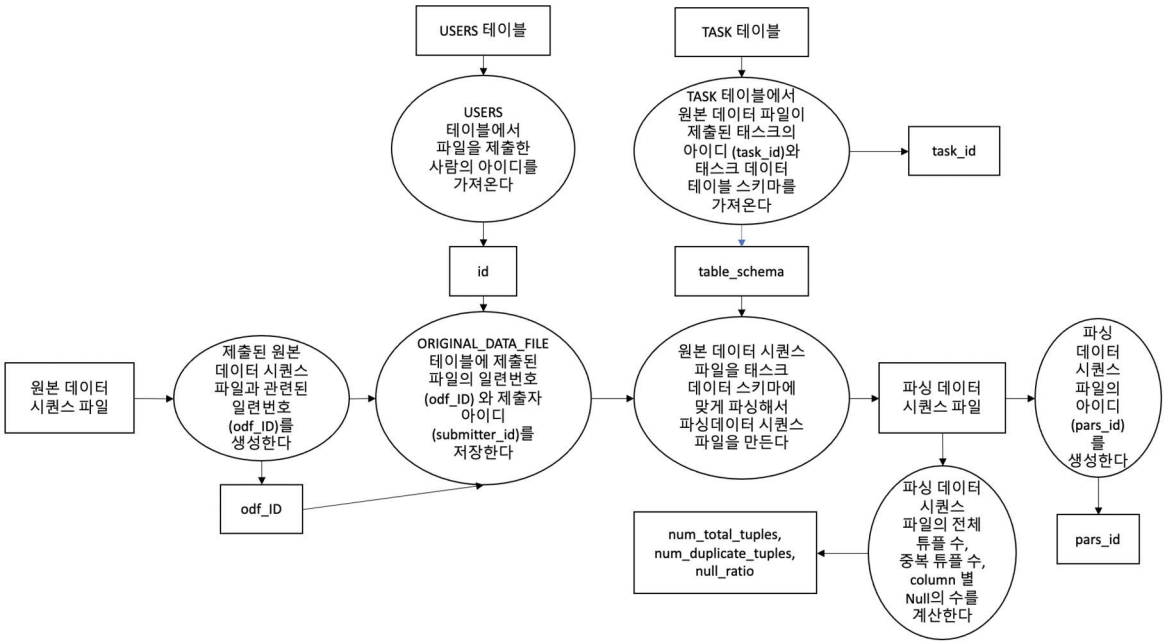
프론트 엔드	곽동우 / HAN,YAXI
백 엔드	권호정 / ZHAN,XITONG
데이터베이스	마준영 / CUI,LIN

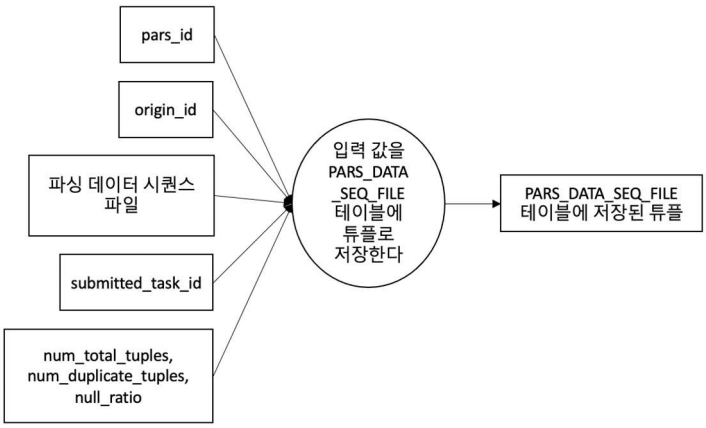
기능별 구현은 각 파트별로 1 팀의 각 파트담당자와 협업하여 진행하며, 최대한 정해진 프로젝트 수행일자까지 완료하도록 한다. 12 월 6 일에 최종제출을 목표로 11 월 말까지는 모든 필수기능의 구현을 목표로하며, 최종제출까지의 시간은 버그수정 또는 추가기능 구현을 위해 사용할 것이다.

ERD 그리기, ERD to relational mapping	11/3 까지완료 (팀별로 같이진행)
Database script 작성+보고서 DB 내용추가	11/10 까지 완료
구현 목표 기능 평가자 기능: 평가자 메인 페이지 완성 제출자 기능: 제출자 메인 페이지 완성 관리자 기능: 회원통계 기능 회원 공통 기능: 회원가입, 로그인, 탈퇴	11/13 까지 구현완료
구글폼을 통해 중간보고서 작성	11/15 제출
구현 목표 기능 평가자 기능 : 파일 평가 제출자 기능 : 태스크 참여 신청, 원본데이터(csv) 제출 관리자 기능 : 태스크 생성/추가, 원본데이터 타입 추가, 제출자 승인/거부	11/18 까지 구현완료
구현 목표 기능 평가자 기능 : 평가 점수 관리 제출자 기능 : 평가 점수 확인 관리자 기능 : 태스크 통계, 검색(파일수, 튜플수, 제출자 목록 등)	11/22 까지 구현완료
구현 목표 기능 평가자 기능 : 평가 내역 모니터링 제출자 기능 : 제출 현황 모니터링 관리자 기능 : 통계기능 보완, csv 파일 다운로드 기능	11/29 까지 구현완료
전체적인 구현완료, 버그수정, 시간에 따라 추가기능 구현	12/6 까지 프로젝트 제출

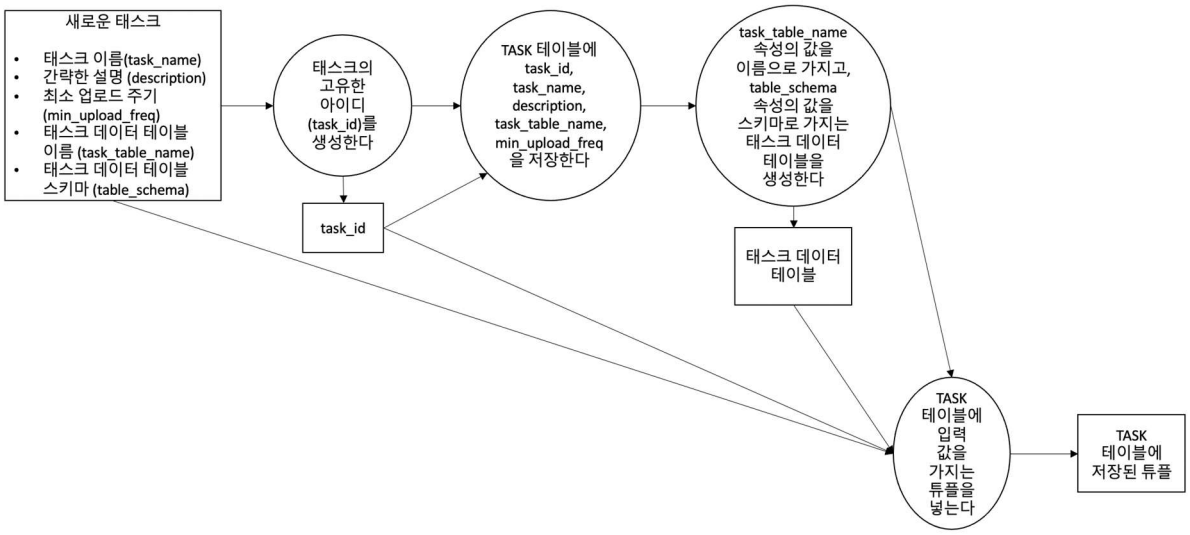
3. 시스템 설계 문서

본 시스템 설계 문서는 두 팀의 백 엔드 담당자분들(1 팀: 권호정, 2 팀: ZHAN,XITONG)께서 작성하셨습니다.

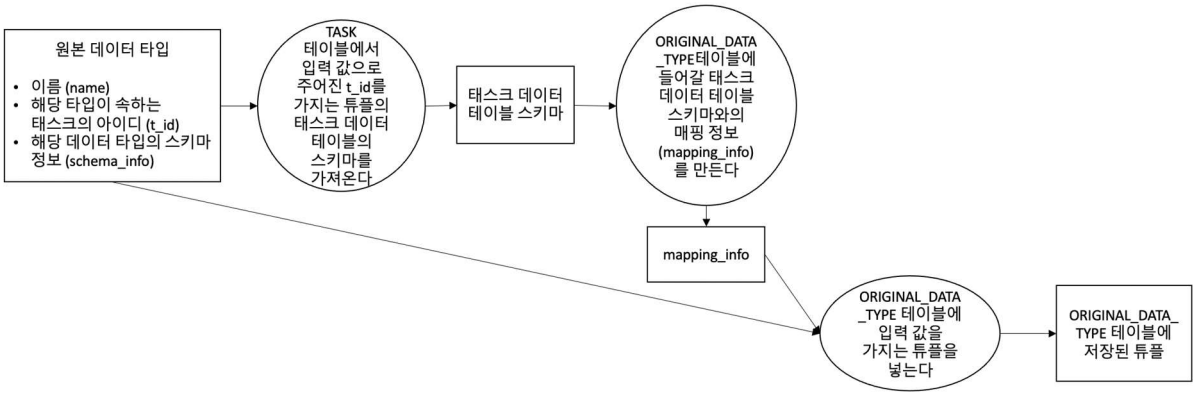




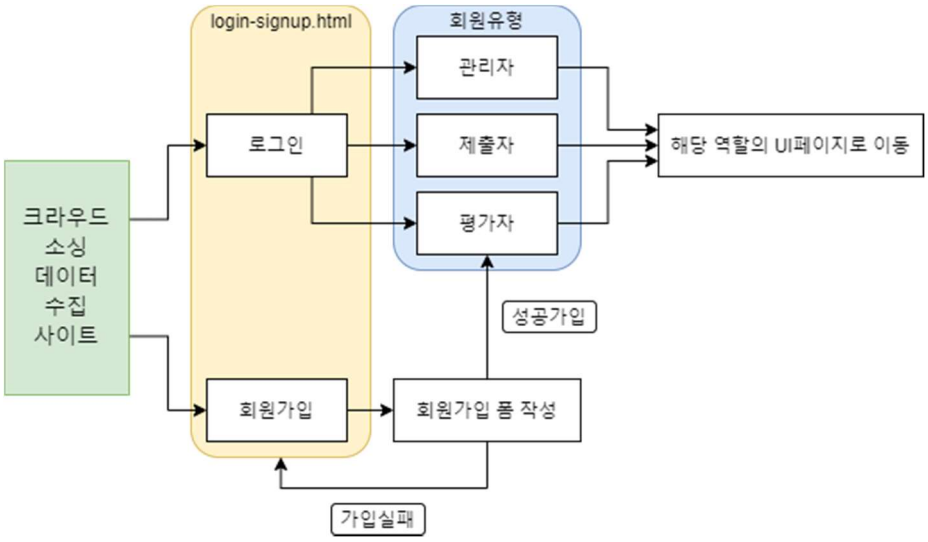
위의 두개의 그림은 제출자가 원본 데이터 시퀀스 파일을 제출했을 때에 일어나는 데이터, 백엔드 함수, DBMS 와 데이터 베이스 간의 상호작용을 표현한다. 입력과 출력 데이터는 직사각형으로 표현되었고, 백엔드 함수와 DBMS 의 역할은 타원으로 표현되었다. 아래쪽 그림에서 pars_id 는 위쪽의 그림에서 생성된 파싱 데이터 시퀀스 파일의 아이디이고, origin_id 는 제출된 원본 데이터 파일과 관련하여 생성된 일련번호 (odf_ID) 와 같은 값을 가진다. 또한, submitted_task_id 는 위쪽 그림에서와 같이 TASK 테이블에서 가져온 태스크의 아이디 (task_id) 와 같은 값을 가진다.



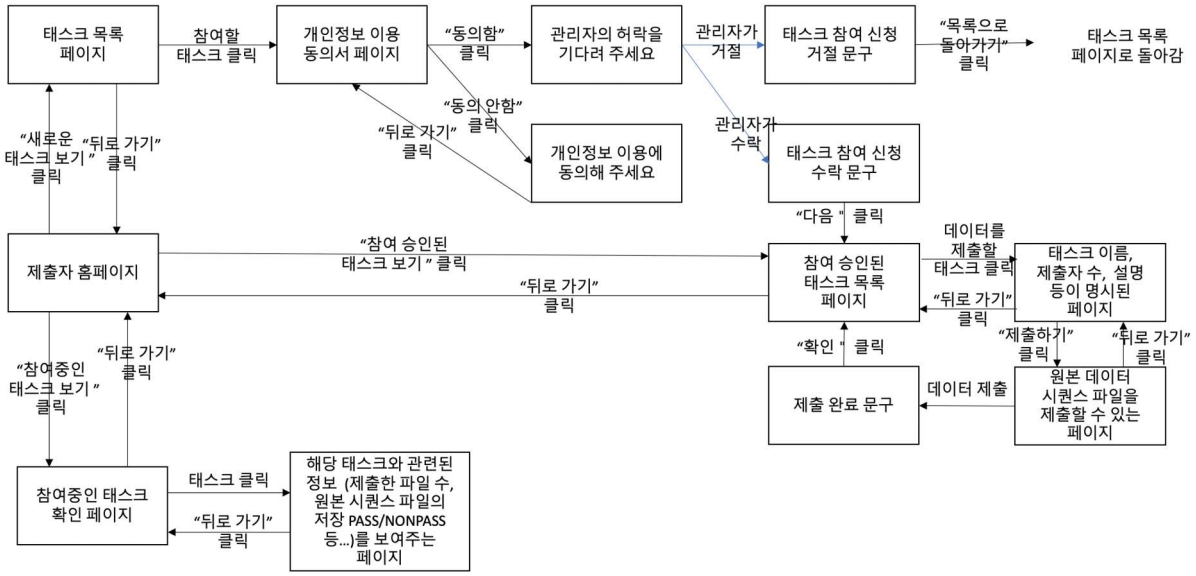
위의 그림은 관리자가 새로운 태스크를 생성할 때 일어나는 데이터, 백엔드 함수, DBMS 와 데이터 베이스 간의 상호작용을 표현한다.



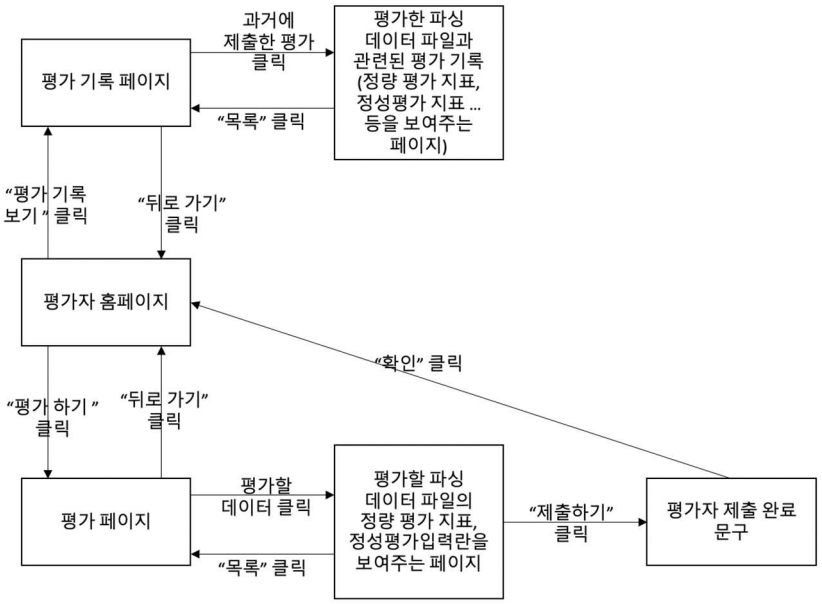
위의 그림은 관리자가 새로운 태스크를 생성한 다음 해당 태스크와 관련된 원본 데이터 타입을 정의할때 일어나는 데이터, 백엔드 함수, DBMS 와 데이터 베이스 간의 상호작용을 표현한다.



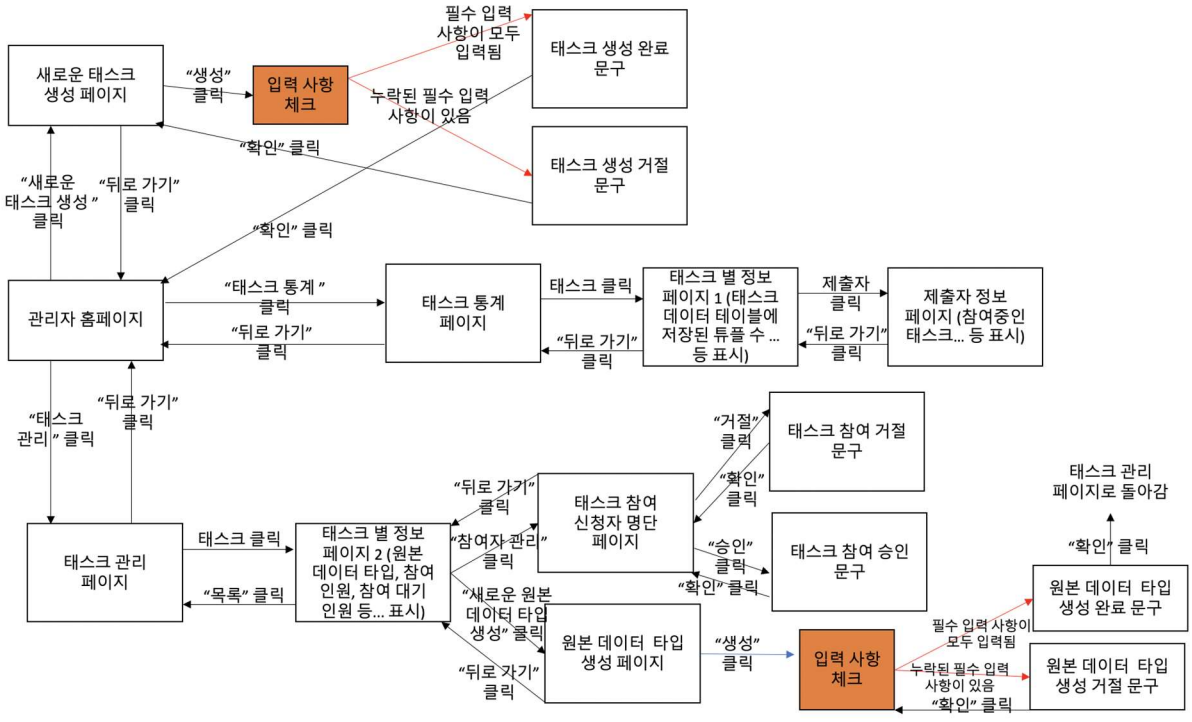
사이트 방문시 첫 화면 에서는 로그인과 회원가입기능을 보여준다. 로그인 시 에는 해당 역할의 UI 페이지로 넘겨주며, 회원가입 성공 시에는 회원 역할을 수행할 수 있으며, 실패 시 실패메시지를 보여주고 다시 첫 페이지로 돌아온다.



위의 그림은 사용자가 제출자로 로그인 했을때에 사용자와 웹사이트 유저 인터페이스 간의 상호작용을 보여준다. 웹사이트 안의 페이지는 직사각형으로 표현되어 있고, 사용자의 행동은 화살표로 표현되어 있다. 파란 화살표로 표시된 "관리자가 거절", "관리자가 수락"은 제출자가 아닌 관리자가 하는 역할이다.



위의 그림은 사용자가 평가자로 로그인 했을때에 사용자와 웹사이트 유저 인터페이스 간의 상호작용을 보여준다.



위의 그림은 사용자가 관리자로 로그인 했을때에 사용자와 웹사이트 유저 인터페이스 간의 상호작용을 보여준다. 주황색 직사각형 안의 "입력 사항 체크"는 사용자가 "생성" 버튼을 누른 뒤에 백엔드 함수가 하는 것으로 웹사이트 안에

나타나는 페이지가 아니다.

4. 중간 구현 내용

회원기능: 회원가입, 로그인, *탈퇴(미완성)*, 회원통계기능*(보완필요)* 웹사이트 디자인 구현완료 (HAN,YAXI/CUI,LIN)

평가자/제출자: 웹페이지 디자인 구현 (곽동우)

1.회원가입 기능 (DB 연동)

{"isreg":true,"result":{"code":200,"msg":"가입성공!"}}

회원가입 클릭 시: /add 경로에서 "가입성공" 메시지 호출

{"isreg":false,"result":{"code":1,"msg":"같은 이름의 사용자가 존재합니다."}}

회원가입 아이디 존재 시: /add 경로에서 실패메시지 호출

2.로그인

(개인정보수정+탈퇴기능 보완필요)

{"isLogin":true,"userInfo":{"id":"admin"},"result":{"code":200,"msg":"로그인되었습니다."}}

로그인 성공 시: /checkin 경로로 로그인 성공 메시지 전송

{"isLogin":false}

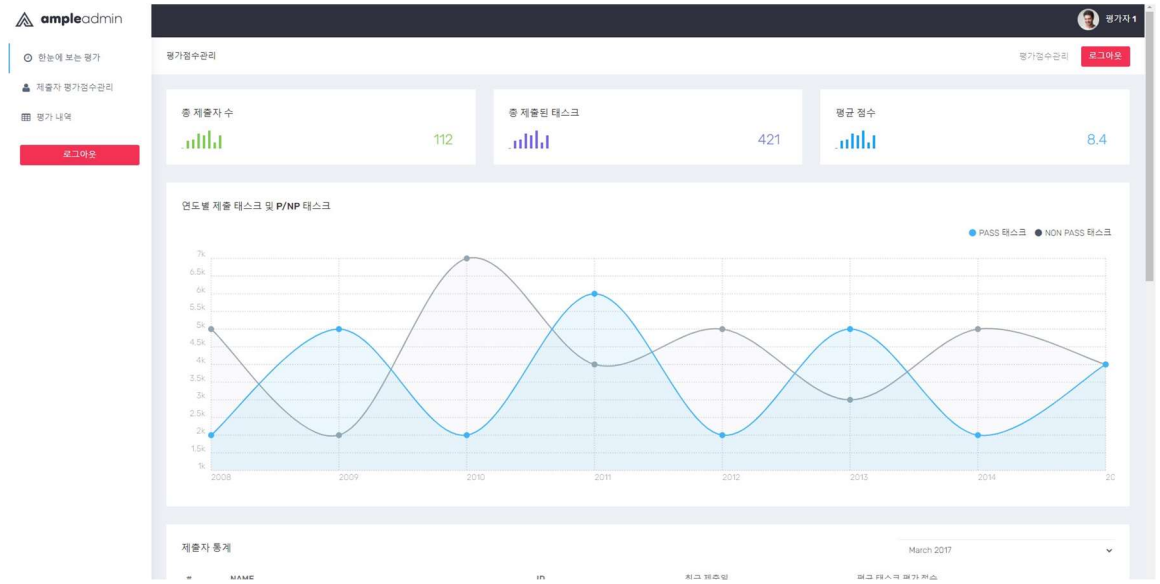
로그인 실패 시:/checkin 경로로 로그인 실패 메시지 전송

3.회원통계(DB 연동)

전체회원 목록:/users 경로로 mysql DB 와 연동하여 현재 users 테이블에 있는 모든 user 를 불러옴. 클릭시 해당 회원 정보 나타남 (뒤로가기 가능)

추가구현 필요사항: *권한 수정, 관리자에 한해서만 볼 수 있도록 권한 바꿀 예정.*
관리자에 한해 회원정보 수정권한(수정,삭제 등) 부여.

4. 제출자/평가자 측 메인 페이지 디자인



index.html 페이지

기타사항

DB 에 저장된 관리자 계정: ID/password - admin/admin

5. Database table 생성 script

DB table 생성 Script 작성은 서브 1 팀(마준영) 과 서브 2 팀(CUI,LIN)의 데이터베이스 담당자가 토론하고 수정하며 작성하였다.

script 파일위치: db_ver1115_final/database.sql

```
CREATE DATABASE IF NOT EXISTS db;
USE db;

/* 숫자형 INTEGER 형에서 길이를 제한할 경우 경고메세지 뜸 */
/* 해결: 숫자형에 대해서 길이제한을 삭제 */
CREATE TABLE IF NOT EXISTS users (
      id VARCHAR(20) NOT NULL UNIQUE,
    password VARCHAR(20) NOT NULL,
    email VARCHAR(255) DEFAULT "" NOT NULL,
    username VARCHAR(255) NOT NULL,
    sex VARCHAR(255) NOT NULL,
    age INTEGER,
    date_of_birth DATE,
```

```
    address VARCHAR(255),
    cellphone VARCHAR(255),
      /*role=0:관리자 role=1:제출자 role=2:평가자*/
    role INTEGER NOT NULL,
      score INTEGER DEFAULT 0 NOT NULL,
    created_at DATE NOT NULL,

      /*key*/
      CONSTRAINT USER_ID_PK
            PRIMARY KEY (id),

      /*check*/
      CONSTRAINT SEX_CONSTRAINT
            CHECK (sex="남자" OR sex="여자"),
      CONSTRAINT AGE_CONSTRAINT
            CHECK (age>=0),
      CONSTRAINT ROLE_CONSTRAINT
            CHECK (role=1 OR role=2 OR role=0),
      CONSTRAINT SCORE_CONSTRAINT
            CHECK (score>=0)
    );

/* 관리자 계정 생성 */
INSERT INTO
users(id,password,email,username,sex,role,score,created_at)
VALUES
('admin','admin','admin@admin.com','관리자','남자',0,0,STR_TO_DATE('2020-
11-01','%Y-%m-%d'));

CREATE TABLE IF NOT EXISTS original_data_files (
      submitter_id VARCHAR(20) NOT NULL,
      odf_ID INTEGER NOT NULL UNIQUE,

      /*key*/
      CONSTRAINT ODF_PK
            PRIMARY KEY (odf_ID),
      CONSTRAINT SUBMITTER_ID_FK
            FOREIGN KEY (submitter_id)
                  REFERENCES users (id)
                  ON DELETE CASCADE
                  ON UPDATE CASCADE
);

/* ERD 에서 task 부분에 participate_id 를 is_closed 로 바꿈 */
/* 참여자의 id 는 해당 파싱파일 table 에 저장됨 */
CREATE TABLE IF NOT EXISTS task(
      task_id INTEGER NOT NULL UNIQUE,
      task_name VARCHAR(255) NOT NULL,
      description VARCHAR(255),
      task_table_name VARCHAR(255) NOT NULL,
      min_upload_freq INTEGER,
      is_closed BOOLEAN DEFAULT (FALSE),
```

```
      table_schema VARCHAR(255) NOT NULL,
      /*key*/
      CONSTRAINT TASK_ID_PK
            PRIMARY KEY (task_id),
      /*check*/
      CONSTRAINT TASK_INFO_CONSTRAINT
            CHECK(task_id>=0),
            CHECK(min_upload_freq>=1)
);

CREATE TABLE IF NOT EXISTS pars_data_seq_file(
      /* Information */
      pars_id INTEGER NOT NULL,
    duration_start DATE NOT NULL,
    duration_end DATE NOT NULL,
    iteration INTEGER NOT NULL,
    orign_id INTEGER NOT NULL,
    submitted_task_id INTEGER NOT NULL,
    rater_id VARCHAR(20),
    is_evaluated BOOLEAN DEFAULT FALSE,

    /* Quantitative evaluation */
    num_total_tuples INTEGER,
    num_duplicate_tuples INTEGER,
    null_ratio REAL,

    /* Qualitative evaluation */
    quality_score INTEGER,
    is_passed VARCHAR(2),

    /* Keys */
    CONSTRAINT PARS_PK
            PRIMARY KEY (pars_id),

      CONSTRAINT PARS_ORIGN_FK
            FOREIGN KEY (orign_id)
                  REFERENCES original_data_files (odf_ID)
                  ON DELETE CASCADE
                  ON UPDATE CASCADE,

    CONSTRAINT PARS_TASK_FK
            FOREIGN KEY (submitted_task_id)
                  REFERENCES task (task_id)
                  ON DELETE CASCADE
                  ON UPDATE CASCADE,

    CONSTRAINT PARS_RATER_FK
            FOREIGN KEY (rater_id)
                  REFERENCES users (id)
                  ON DELETE CASCADE
                  ON UPDATE CASCADE,

    /* Checks */
```

```
    CONSTRAINT PARS_ID_CONSTRAINT
           CHECK (pars_id >= 0),

     CONSTRAINT PARS_DURATION_CONSTRAINT
           CHECK (duration_end >= duration_start),

    CONSTRAINT PARS_ITERATION_CONSTRAINT
           CHECK (iteration >= 1),

    CONSTRAINT PARS_QUANTI_CONSTRAINT
           CHECK (num_total_tuples >= 0),
           CHECK (num_duplicate_tuples >= 0),
           CHECK (null_ratio >= 0),

    CONSTRAINT PARS_QUALITY_CONSTRAINT
           CHECK (quality_score >= 0 AND quality_score <= 10),

    CONSTRAINT PARS_P_NP_CONSTRAINT
           CHECK (is_passed = "P" OR is_passed = "NP")
);

CREATE TABLE IF NOT EXISTS original_data_type(
     t_id INTEGER NOT NULL,
     name VARCHAR(255) NOT NULL,
     schema_info VARCHAR(255),
     mapping_info VARCHAR(255) NOT NULL,

     /*key*/
     CONSTRAINT ODT_NAME_PK
           PRIMARY KEY (name),
     CONSTRAINT TASK_ID_FK
           FOREIGN KEY (t_id)
                REFERENCES task (task_id)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);
```