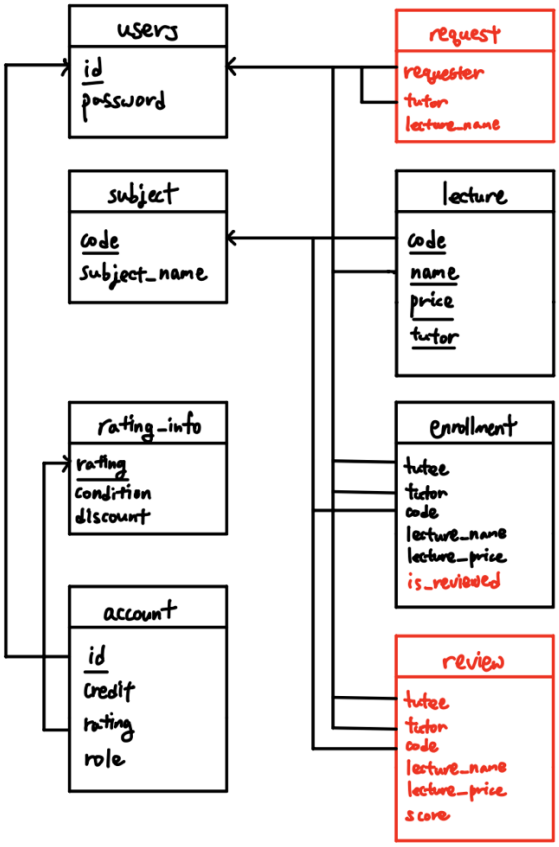
Term Project Report

2021320117 배민성

1. Schema Diagram



위 schema diagram에서 붉은색으로 표시되어 있는 부분은 기본으로 제공되었던 데이터베이스의 schema에 존재하지 않았던 추가 및 수정된 요소들로 강의 개설 요청 시스템의 구현을 위해 request 테이블을 추가하였으며 강의 평점 시스템을 구현하기 위해 review 테이블 추가 및 enrollment에 리뷰 참여 여부를 파악하기 위한 attribute인 'is_reviewed'를 추가하였다.
(review table의 경우 enrollment와 겹치는 attribute가 많아 redundant할 수 있으나, 평점 평균 점수를 계산하기 위한 과정에서 null 값의 처리나 default review 점수 처리 등의 문제가 있고 review 테이블을 따로 만든다면 해당 table만을 가지고 다양한 추가 기능 구현 및 데이터 분석이 용이하다고 생각해 테이블을 따로 추가하는 형태로 설계하였다.)

2. 구현한 함수 및 추가 기능

웹사이트의 실질적 구현을 위해 app.py에서는 다음과 같은 함수들을 선언하여 사용하였다.
- main() : login 이전의 화면인 main.html을 보여준다.

- login() : main.html에서 id와 password를 입력 받고 사용자가 login 버튼을 눌렀다면 id와 password가 정확하게 입력되었는지 확인한다. SQL query를 통해 user 정보를 가져와서 정확히 아이디와 비밀번호가 일치하는지 확인한다. 또한, user 정보에 따라 user의 role과 admin 여부, id를 로그인 이후 기능들에서 활용하기 위해 전역변

수로 저장한다. 해당 과정은 pay.html로 return할 때도 수행되어야 하므로 함수 형태로 구현하여 사용했다. 이후 정확하게 id, password가 입력되었다면 pay.html, 아니라면 login_fail.html 창으로 넘어간다. 만약 사용자가 sign up 버튼을 눌렀다면 회원가입 창으로 넘어간다.

- signup() : 사용자로부터 id, password, role 입력을 받고 입력된 id가 user table에 이미 존재하는지 SQL 문을 통해 user table 정보를 가져와 확인한다. 존재한다면 id collision 창으로 넘어가게 되고, 그렇지 않다면 SQL을 통해 users와 account에 해당 value를 갖는 tuple을 insert한다. 이후 signup 성공 창으로 넘어가도록 한다.

- admin() : 사용자의 admin 여부를 check하여 admin만 접근 가능한 user 정보 및 trade 정보 접근을 막는다. Login 과정에서 찾은 admin 여부 전역 변수를 활용해 admin이 아니라면 접근이 불가능하다는 창으로 넘어가게 되고, admin이라면 정상적으로 account / enrollment table 정보를 보여주는 페이지에 접근이 가능하다.

- logout() : login 과정에서 저장했던 전역 변수들을 다시 default 값으로 되돌리고, pay 화면에서 메인 화면으로 돌아간다.

- mylectures() : my info 버튼을 눌렀을 때 tutor / tutee 역할에 따라 보이는 창을 다르게 할 수 있도록 구현했다. Tutor는 본인이 개설한 강의의 등록 현황, 본인 등록 강의 및 본인에게 들어온 강의 개설 요청을 확인할 수 있으며, tutee는 본인이 등록한 강의를 표 형태로 확인 가능하고 해당 창에서 새롭게 구현된 기능인 review가 가능하다. 앞에서 설명한 정보들을 가져오기 위한 SQL 문을 실행하였다.

- add() : pay 화면에서 add lectures 버튼을 눌렀을 때 tutee라면 invalid한 접근이라는 창으로 넘어가게 되고, tutor라면 실제 강의 등록을 위한 새로운 창으로 넘어간다. 이때, 새로운 창에 subject 테이블을 그대로 보여줘야 해서 SQL문을 통해 찾은 해당 정보를 같이 넘겨준다.

- add_lecture() : 실제 강의 추가를 위한 함수이다. 강의 추가 창에서 tutor가 입력한 강의 code, 강의명, 강의 가격을 가져와서 해당 강의 추가가 가능한지 체크한다. 해당 강의 코드는 subject 테이블에 존재해야 하며, 가격은 음수일 수 없고 3가지 정보와 tutor까지 모두 같은 강의가 존재하지 않아야 한다. 3가지 중 하나라도 만족하지 못한다면 강의 추가가 불가능하다는 창을 보여주게 되고, 그렇지 않다면 실제로 SQL문을 통해 lecture table에 새로운 강의 tuple을 insert하고 강의 추가에 성공했다는 창을 보여준다.

- register() : 사용자가 pay 화면에서 강의 register 버튼을 눌렀을 때 해당 강의에 실제로 등록이 가능한지를 판단한다. 먼저 register를 원하는 강의의 정보를 html로부터 받아온 후 credit과 rating 정보를 알아내고, 3가지 등록 가능 조건을 확인한다. 먼저 본인이 개설한 강의가 아닌지 확인하고, rating_info table에서 rating에 따른 할인율 정보를 알아와서 할인 가격 및 최종 가격을 계산한다. 만약 최종 가격이 본인의 credit 미만이라면 등록이 불가능하고, 이미 enrollment table에 본인이 해당 강의에 등록했다는 정보가 존재한다면 등록이 불가능하다. 해당 조건을 모두 통과해 강의 등록이 가능하다면 등록 및 결제 confirm을 위한 창으로, 그렇지 않다면 등록 불가능 창으로 넘어간다.

- confirm() : 강의 등록 및 결제의 confirm을 위한 함수이다. 만약 결제 confirm 창에서 사용자가 confirm을 눌렀다면 먼저 html 파일로부터 강의 정보 및 현재 tutee의 credit 보유량 및 최종 결제 가격 등의 정보를 받아오고, SQL로 account 테이블에 접근해 tutor의 credit 보유량을 찾은 후 결제 후 tutee와 tutor의 credit 보유량을 계산한다. 이후 SQL로 account 테이블의 credit column을 새로운 credit 보유량 값으로 update한다. 또한, 새로운 credit 보유량에 따른 rating 조정도 진행한다. SQL로 읽어온 rating_info table을 활용해 새로운 credit 보

유량에 따른 두 사용자의 새로운 rating을 결정해서 account 테이블의 rating column을 update한다. 최종적으로, enrollment 테이블에 강의 등록 정보를 저장하고 등록에 성공했다는 창으로 넘어간다.

- review() : 추가 기능 중 하나인 강의 평점 시스템 구현을 위한 함수이다. 강의 평점은 최소 1점에서 최대 10점의 정수로 tutor가 다른 tutor에게 악의적 평점을 주는 것을 방지하기 위해 role이 tutee인 user만 남길 수 있다. 강의 평점을 남기기 위해서는 my info 창에 들어가서 본인이 등록한 강의 옆에 있는 review 버튼을 눌러야 한다. 해당 함수는 button을 누른 경우 호출되며 해당 강의 정보를 받아온 이후, enrollment table에서 해당 사용자의 강의 등록 정보를 찾아 review를 이미 작성한 상태인지 아닌지 확인한다. 평점은 사용자마다 각 강의당 한 번씩만 매길 수 있다. 만약 이미 본인이 평점을 남긴 강의라면 review fail 창으로, 아직 남기지 않은 강의라면 강의 평점을 매길 수 있는 창으로 넘어간다.

- review_confirm() : 강의 평점 창에서 사용자가 매긴 강의 평점 및 해당 강의 정보를 받아와서 SQL query를 통해 평점에 관한 정보를 review 테이블에 저장하고 enrollment 테이블에서 사용자의 해당 강의 리뷰 참여 여부를 변경한다. 이후 리뷰 성공 창으로 넘어간다.

- request_lec() : 또 다른 추가 기능 중 하나인 강의 개설 요청 시스템 구현을 위한 함수이다. 모든 user는 본인이 아닌 다른 tutor user에게 강의 개설을 요청할 수 있다. 개설 요청이 정상적으로 이루어지면 요청 정보 (요청자명 및 요청 강의명)는 해당 tutor의 my info 창에서 확인이 가능하다. 먼저 해당 함수는 pay page에 있는 request 버튼을 눌렀을 때 호출되며 request 작성을 위한 창으로 넘어가게 된다.

- request_submit() : request 작성 창에서 작성된 request 대상 tutor의 이름과 강의명을 받아오고, 정상적인 request인지에 대한 조건을 확인한다. 먼저, request는 본인에게는 할 수 없으며, account 테이블에 저장되어 있는 반드시 존재하는 tutor에게만 할 수 있다. 두 조건 중 하나라도 만족 못 한다면 request 실패 창으로, 만약 두 조건을 모두 만족한다면 먼저 요청자가 요청한 강의와 비슷한 강의명을 가진 강의가 있는지 확인한다. 강의명의 유사도는 두 문자열 간의 유사도를 측정하는데 사용하는 알고리즘 중 하나인 편집거리 알고리즘을 통해 구현하였고, 해당 코드는 코드 주석에 남겨져있는 reference site에서 가져왔다. 유사도가 일정 threshold를 넘는 강의들은 추천 강의로써, 요청을 제출하기 전 request_recom.html을 통해 먼저 사용자에게 추천 유사 강의를 보여주게 된다.
- request_final() : request_recom.html에서 사용자가 유사 강의 목록을 확인한 이후에도 요청을 진행하기를 원했다면, 최종적으로 요청을 처리한다. 단, 이때 같은 tutor에게 같은 강의명으로 이미 request를 한 적이 있다면 중복된 request를 할 수는 없다. 이를 request table을 통해 체크하여 중복된 request가 있다면 request 중복 창으로 넘어가게 되고, 그렇지 않다면 request 정보를 request table에 저장하게 된다. 그리고 request 성공 창으로 넘어간다.

- return_page() : HTML 페이지들에서 메인 페이지 혹은 결제 페이지로 돌아가기 위해 구현한 함수로 return 등의 버튼을 누를 시 main.html 혹은 pay.html로 돌아가도록 구현했다. 다만 pay.html로 돌아갈 때는 인기 강의 목록, 강의 및 account 정보가 update된 상황일수도 있기 때문에 항상 다시 함수를 통해 정보를 찾아서 전달해 준다.

위 함수들 내에서 사용하기 위해 구현한 추가적인 함수는
- get_popular_lst() : pay page에 나오는 등록 수가 가장 많은 subject, 강의, tutor를 찾기 위한 함수로서 enrollment table에서 각 attribute 별로 group by 한 후 count를 통해 해당 attribute 별 등록 인원 수를 찾고,

인원 수가 최대인 attribute 값들 중 하나를 선택하여 return하게 된다.
- get_account_info : user의 id를 이용해 account table에서 해당 user의 credit, rating, role을 알아낸다.
- get_score : pay page에 나오는 강의 list의 score는 review 테이블을 통해 계산된 해당 강의의 평균 평점이다. 이를 계산하기 위한 함수이다.
- get_lectures : lecture 테이블에서 강의 list를 읽어오고, 해당 강의의 평균 점수를 get_score를 통해 계산해서 강의 정보와 평균 점수 두 개를 합친 tuple list를 return한다.
- levenshtein_distance, compute_similarity : 강의명 유사도 계산을 위한 함수들이다.

3. 사용한 HTML 파일들

구현 과정에서 제작한 HTML 파일들은 다음과 같다.
- main.html : login을 위한 창으로 id, password 입력 칸과 login / sign up button이 있다.
- login_fail.html : 로그인에 실패했을 때 보여지는 창이다.
- signup.html : sign up을 위한 창으로 id, password 입력 및 role 선택이 가능하다.
- signup_success.html : 가입에 성공했을 때 보여지는 창이다.
- ID_collision.html : 동일한 id가 존재해 회원가입에 실패했을 때 보여지는 창이다.
- pay.html : 로그인 성공 시 보이는 창으로 admin 전용 기능, my info, 인기 강의, 강의 추가 및 개설 요청, 강의 목록 등 다양한 정보가 나오는 창이다.
- print_users.html : admin 전용 기능 중 하나인 user 정보를 모두 볼 수 있는 창이다.
- print_trades.html : admin 전용 기능 중 하나인 enrollment 정보를 볼 수 있는 창이다.
- admin_only.html : admin이 아닌 user가 admin 전용 기능 버튼을 누를 경우 보여지는 창이다.
- tutor_myinfo.html : tutor user가 my info를 눌렀을 때 보여지는 창이다. 본인 강의 등록 현황, 본인이 등록한 강의 목록 및 개설 요청 목록을 볼 수 있다.
- tutee_myinfo.html : tutee user가 my info를 눌렀을 때 보여지는 창이다. 본인이 등록한 강의 목록을 볼 수 있으며 review 버튼을 누르면 평점을 매길 수 있다.
- review.html : review button을 누를 경우 보여지는 창으로 해당 강의 정보를 볼 수 있으며 강의 평점을 제출할 수 있다.
- review_fail.html : 이미 review를 남긴 강의의 경우 review button을 누를 시 해당 창이 보여진다.
- review_success.html : review 성공 시 보여지는 창이다.
- add_lecture.html : tutor인 user가 add lecture 버튼을 눌렀을 때 보여지는 창으로, subject 테이블과 강의 코드 / 강의명 / 가격 입력이 가능하다.
- invalid.html : tutee인 user가 add lecture 버튼을 누를 경우 보여지는 창이다.
- lecture_add_success.html : 강의 추가에 성공한 경우에 보여지는 창이다.
- lecture_add_fail.html : 강의 추가에 실패한 경우에 보여지는 창이다. 강의 추가가 실패하는 경우에 대한 설명이 적혀있다.
- register.html : 강의 등록을 위해 register button을 누를 경우 보여지는 창으로 강의 정보, 원가 및 할인가, 최종 가격 등이 적혀있다.
- lecture_reg_fail.html : register button을 눌렀을 때 해당 user가 해당 강의에 등록이 불가능한 경우에 보여지는 창이다. 강의 등록이 불가능한 경우에 대한 설명이 적혀있다.
- lecture_reg_success.html : register.html에서 강의 등록에 대한 confirm 버튼을 누를 경우 보여지는 창이다.
- request.html : 강의 개설 요청을 위한 창으로 현재 강의 목록 확인이 가능하며 요청 대상 tutor 이름 및 요청 희망 강의명 입력이 가능하다.
- request_fail.html : 강의 요청에 실패한 경우에 보여지는 창이다. 강의 요청이 불가능한 경우에 대한 설명이 적혀있다.

- request_recom.html : 요청한 강의명과 강의명이 유사한 강의들 목록을 보여주고, 다시 한 번 사용자에게 요청을 할 것인지 질문한다.
- request_success.html : 사용자가 request_recom.html에서 요청을 희망하여 Yes 버튼을 눌렀을 때 요청에 성공한다면 보여지는 창이다.
- request_collision.html : 사용자가 request_recom.html에서 요청을 희망하여 Yes 버튼을 눌렀을 때 이미 같은 요청이 존재하여 요청에 실패한다면 보여지는 창이다.