## Spring 1차과제(일정관리)

- ▼ 테이블 정보 [ERD]
  - ▼ schedule
    - 일정 테이블
  - ▼ writer
    - 작성자 테이블
  - ▼ 캡쳐본참고

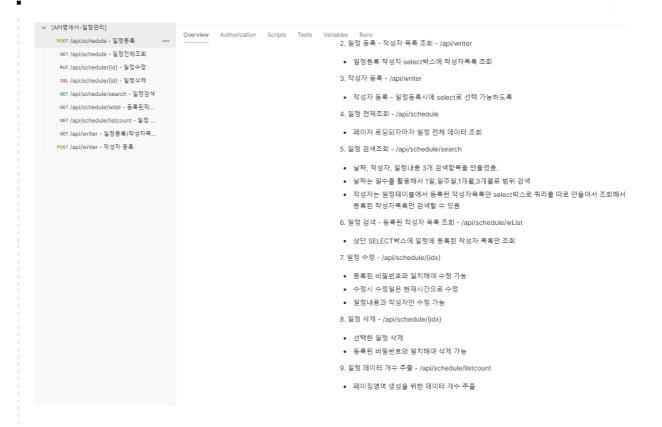
□ name /\* 이름 \*/ varchar(45) ■ email /\* 이메일 \*/ varchar(45) □ reg\_date /\* 작성일↓ □ 작성되는 동시 자동 \*/ □ edit\_date /\* 수정일과 도시에 저장과 수후에 수정할때 수정하는 시점으로 UPDATE \*/ schedule /\* 일정관리 테이블 \*/ [☑ w\_id] /\* 작성자 고유번호 ↓ □ 작성자 테이블과 연동 \*/ ☐ memo /\* 일정 내용 \*/ varchar(150) Ⅲ w\_name /\* 작성자 이름√ ㄴ 작성자 테이블과 연동 \*/ varchar(50) □ pw /\* 비밀번호 \*/ longtext D pw\_check /\* 비밀번호 확인√ 느 임시 (스크립트로 처리할지 DB로 처리할지 고민) \*/ longtext ∏ reg\_date /\* 작성일↓∟ 작성되는 동시 자동 \*/ "Dedit\_date → 수정일과 도시에 저장과 수후에 수정할때 수정하는 시점으로 UPDATE \*/ 

## ▼ API명세서

- ▼ 일정 테이블
  - ▼ 일정 등록

- POST
- /api/schedule
- 요청 body
- 등록정보
- 200 정상등록
- ▼ 일정 전체조회
  - GET
  - /api/schedule
  - 요청 param
  - 다건 응답정보
  - 200 정상조회
- ▼ 일정 검색
  - GET
  - /api/schedule/search
  - 요청 param
  - 다건 응답정보
  - 200 정상조회
- ▼ 일정검색 등록된 작성자 목록만 조회
  - GET
  - /api/schedule/wList
  - 요청 param
  - 다건 응답정보
  - 200 정상조회
- ▼ 일정 수정
  - PUT
  - /api/schedule/{id}
  - 요청 body
  - 수정 정보
  - 200 정상수정
- ▼ 일정 삭제

- DELETE
- /api/schedule/{id}
- 요청 param
- . .
- 200 정상삭제
- ▼ 현재 데이터 개수(페이징세팅)
  - GET
  - /api/schedule/listcount
  - 요청 param
  - 단건 응답정보
  - 상태 200 정상조회
- ▼ 작성자 테이블
  - ▼ 일정등록 작성자 목록 조회
    - GET
    - /api/writer
    - 요청 param
    - 다건 응답정보
    - 200 정상조회
  - ▼ 작성자 등록
    - POST
    - /api/writer
    - 요청 body
    - 등록정보
    - 200 정상등록
- ▼ 캡쳐본참고



## ▼ 요구사항

- ▼ 레벨1
  - API명세서작성
  - ERD작성
  - SQL작성
- ▼ 레벨2
  - ▼ 일정 생성
    - 할일, 작성자명, 비밀번호, 작성일, 수정일 저장
    - 작성 수정일은 DATETIME형태
    - ▼ 최초입력시 작성일과 수정일은 동일한 시간으로 남도록
      - 등록될때 수정일도 함께 등록
    - ▼ 일정 ID는 자동으로 생성 및 관리
      - 기본키로 관리
  - ▼ 일정 조회
    - ▼ 전체일정조회
      - 수정일, 작성자명 목록에 보이도록

- 수정일 기준 내림차순 정렬
- ▼ 선택일정조회
- 일정 ID를 기준으로 선택 조회되도록
- ▼ 레벨3
  - ▼ 일정 수정
    - 비밀번호 입력하면 수정할 수 있도록
    - 내용과 작성자명만 수정 가능
  - ▼ 일정 삭제
    - 비밀번호 입력하면 삭제할 수 있도록
- ▼ 레벨4
  - 작성자와 일정연결
- ▼ 레벨5
  - 페이징 기능 넣기
  - Paging 객체 활용해보기