BackEnd 개발

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/ec9ff8ab-166e-4703-b263-35f50a365df5/BackEnd__.pptx

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/a66885ea-1c7e-4ce6-9039-9288f0b0a828/pxd-study_API.md

Database

Database 는 데이터의 저장소 입니다.

간혹 프론트 프로젝트에서 API가 만들어지기 전 Fake Server 로 Json 파일을 key+value 형태로 만들고 사용하실 때가 있는데, Database는 Json의 Value의 값들을 갖고 있습니다.

기본적으로 CRUD 라고 하여 INSERT(등록), SELECT(조회) UPDATE(수정) DELETE(삭제) 라는 쿼리 명령어를 통해 데이터 작업이 일어나게 됩니다.

Table 예시

Aa User	# ID	NAME	AGE
제목 없음	1	아무개	17
제목 없음	2	김개발	20
제목 없음	3	이개발	21

DB에서 값을 조회할때는

FROM > WHERE > GROUP BY > HAVING> SELECT > ORDER BY

의 순서로 실행됩니다.

여기서 **FROM WHERE SELECT** 에 대해 안내 하자면

FROM 은 어디서 조회할지를 지정합니다. 내가 찾고자 하는 정보의 위치(테이블) 입니다.

WHERE 은 어떤 조건으로 할 것이냐 입니다. 조회 수정 삭제의 경우 조건에 따라 작업이 이루어집니다.

SELECT 는 어떤 정보를 가져올 지 지정합니다. user 라는 테이블에서 이름만 필요할 수도 있기 때문 입니다.

조회

SELECT ID, NAME, AGE FROM **USER** WHERE ID = 1;

내가 조회하고자 하는 테이블을(**볼드**) 지정하고 조회하고자 하는 필드(키)를(파란글자) 지정합니다

마지막으로 조건을(오렌지 배경) 지정합니다.

등록

INSERT INTO **USER** (NAME, AGE) VALUE('이개발', 17);

혹은

INSERT INTO **USER** VALUE(3, '이개발', 17);

내가 등록하고자 하는 값을 넣을 테이블을(**볼드**) 지정하고 등록 하고자 하는 필드를(파란글자) 지정합니다

이후 등록하고자 하는 필드의 갯수와 타입에 맞게 등록할 값을(보라색글자) 지정합니다

- *등록하는 값이 컬럼의 테이블에 만들어진 필드 갯수와 동일하다면 값만 넣어주면 됩니다.

수정

UPDATE **USER** SET AGE = 18, name='박개발' WHERE ID = 1;

내가 수정하고자 하는 테이블을(**볼드**) 지정하고 수정하고자 하는 필드(키)를(파란글자) 지정합니다

마지막으로 조건을(오렌지 배경) 지정합니다.

삭제

DELETE FROM **USER** WHERE ID = 1;

내가 삭제하고자 하는 테이블을(**볼드**) 지정하고 조건을(오렌지 배경) 지정합니다.

API

API 는 프론트엔트 프로젝트에서 데이터 바인딩을 위해 서버에서 호출하게되는 정보 입니다.

대개 Backend 개발팀에서 Database 에서 정보를 조회 및 가공 후 전달하게 됩니다.

형태나 사용 용도에 따라 GET, PATCH(PUT), POST, DELETE 등의 방식으로 요청하는데 이를 REST 방식 이라 합니다.(그중 많이 접해본 4가지에 대해 안내 드립니다)

GET: 조회를 위한 용도의 요청 (queryString으로 요청합니다)

PATCH(PUT): 수정을 위한 용도의 요청 (body에 값을 담아 요청합니다)

POST: 등록을 위한 용도의 요청 (body에 값을 담아 요청합니다)

DELETE: 삭제를 위한 용도의 요청 (queryString으로 요청합니다)

PATCH 와 PUT은 둘다 수정을 위한 용도인데, 차이점으로는 PATCH는 대상의 수정되어야 할 부분만 요청하면 되지만, PUT은 변경되지 않은 정보도 요청해야합니다.

PATCH / PUT 비교

사용자정보

{name: ‘아무개’, age: 17}

위 정보를 받아서 나이를 18세로 변경할때

PATCH: {

age: 18

}

PUT: {

name: ’아무개’,

age: 18

}

위 형태처럼 PUT에서는 수정요청 시 모든 값을 지정해야하며 유지되는 정보라도 값이 누락되어 있으면 해당 정보의 값을 null 로 처리하게 됩니다

4가지의 REST 요청 방식에 대해 안내하였지만, 실제 개발팀에서 API 전달해주는 것을 보면 GET 방식과 POST방식으로만 제공하는것을 볼 수 있습니다.

이는 여러가지 이유가 있으나, 그중 하나로는 서버 관리자가 GET 과 POST 요청에 만 보안설정을 하는 경우가 있기 때문입니다.(특정요청에 대해서만 서버에서 응답)

그러니 꼭 PATCH, DELETE가 없다고 잘못된 API는 아니라는 의미 입니다.

대개 backend 개발자와 협업하여 API를 제공받을경우 API를 어떻게 요청하고 어떻게 전달해야하는지에 대한 명세를 제공받게 됩니다,

API명세서 라고 하며, JAVA(spring)등의 프로젝트에서는 Swagger 라는 라이브러리를 통해 서버에서 API 목록과 직접 테스트해볼 수 있는 환경을 제공하기도 합니다.

필수 항목의 경우 Require 혹은 '*' 등의 한눈에 알 수 있는 표시로 안내하게 되며 해당 정보를 누락하게 되면 서버에서는 에러를 반환하게 됩니다.

(간혹 서버에서 예외처리 코드가 없을 경우 서버가 다운되기도 합니다)

에러코드

에러는 4xx, 5xx 에러가 존재합니다.

4xx 에러는 요청시 에러로 API 주소를 잘못 입력하거나 필수값을 누락할시 반환하는 종류 입니다

대표적으로 403 forbidden 서버가 허용하지 않은 요청을 할 경우 반환합니다.

5xx 에러는 서버에서 문제가 발생하여 반환하는 에러 입니다.

대표적으로 500 에러가 있으며, 접속자 폭주 등으로 서비스가 일시적으로 중단될 때 반환됩니다.

다음 링크를 참고하면 더 많은 에러코드 정보를 확인하실 수 있습니다.

https://developer.mozilla.org/ko/docs/Web/HTTP/Status

필드 추가 시 API 추가의 과정은?

사용자 정보 중 키(신장) 정보가 신규로 추가되어야 한다면 Backend 에서는

1.Database의 사용자 테이블에 키(신장) 정보를 담을 필드 추가

2.서버에서 사용자정보를 담는 type(변수를 객체 파일에 따로 지정한 파일 이라고 생각하시면 됩니다)을 찾아 키 (신장)변수 추가

3.서버에서 사용자 정보를 관리하는 메서드(함수)를 찾아 쿼리에서 키(신장) 정보를 CRUD 할 수 있도록 수정

4.변경된 API에 대해 명세 수정 후 Front 개발자에게 전달

API Backend 개발자 작업 요약

1.서버에서 db값을 조회하여 값을 담을 class(파일) 생성 및 타입과 변수 선언

2.db 에서 값을 CRUD 후 1번에 만든 변수에 대입 및 가공

3.front 에서 api 호출에 필요한 url 과 함수 매핑

마무리

제공받은 API가 명세서대로 했는데 에러가 날 경우 개발자에게 바로 문의하시면 시간을 절약할 수 있습니다.

개발자가 명세서에 작성하지않은 부분이 있을수도 있습니다.

보안적인 이유로 특정 서버에서만 요청이 되는것일 수도 있습니다.

(CORS 많이들 겪어보셨죠?ㅎㅎㅎ)

혹은 API호출에 대해 적응되지 않아 모를수도 있습니다.

이런 상황은 본인이 혼자 해결하려고 시간을 끄는 것보단 주변에 물어보고 개발자에게 도움을 받는게 효율이 높다 할 수 있습니다.