

2023-하계방학 자기주도학습동아리 활동보고서

팀명	JUMP		
학습일시	2023년 07월 18일 (화요일)	진행차수	4주차
학습시간	20시 00분 ~ 22시 00분 (2시간)		
참석자	이 영 교 (서명)	김 태 현 (서명)	손 민 재 (서명)
	이 호 영 (서명)	최 재 신 (서명)	진 정 원 (서명)
불참자	X		
구 분	내 용		
학습목표	<ul style="list-style-type: none"> - 프론트엔드와 백엔드의 HTTP 통신 REST API를 디자인할 수 있다. - 서버와의 통신에서 적절한 상태값과 응답값을 전달하는 방법과, 프론트엔드에서 요청하는 URL을 설계할 수 있다. 		
학습 방법 및 학습 내용	<p>0. 이번 주차는 노선에 내용을 정리해서 세션처럼 설명하는 방식으로 진행했습니다. 기본적으로 클라이언트와 서버가 통신하는 방법에 대해 백엔드, 프론트엔드 개발 인원들이 모두 알아야 하는 공통 주제로 설정하여, 사이트를 개발할 때 기능들을 조금 더 구체화할 수 있는 시간을 갖게 되었습니다.</p> <p>이번 세션에서는 웹 서비스를 운영하며 프론트엔드에서 필요한 자원(Resource)을 백엔드 서버와 통신하기 위해 정하는 규칙(REST API)의 올바른 디자인 가이드를 살펴보는 시간을 갖게 되었습니다.</p> <p>REST API는 백엔드 개발자 인원만 생각하고 바꾸는 것이 아닌, 프론트에도 영향을 주는 설계이므로 같이 신중하게 검토와 소통을 통해 재구성해야 하는 단계였습니다.</p> <p>- 노선 링크 :</p> <p>1) REST API 설계 : https://citrine-situation-94c.notion.site/REST-API-03cb739493d94ef287af4bb1a3fdd92e?pvs=4</p> <p>1. REST란?</p> <p>REST는 웹 아키텍처가 웹의 본래 설계의 우수성을 많이 사용하지 못하고 있다고 판단하고, 웹의 장점을 최대한 활용할 수 있는 네트워크 기반의 아키텍처를 소개한 것을 말합니다.</p> <p>웹에 존재하는 모든 자원(이미지, 동영상, DB 등)에 고유한 URL을 부여하여 이를 필요로 하는 프론트엔드에게 URL을 전달하는 방법으로 활용한다.</p> <p>RESTful이란, REST 원리를 따르는 시스템으로 REST 특징을 지키면서 API를 제공한다.</p> <p>2. 용어 정리</p> <p>리소스(Resource) : 데이터의 일부</p> <p>컬렉션(Collection) : 리소스의 집합</p> <p>URL : 리소스 혹은 컬렉션을 식별할 수 있는 경로(Ex. ./company/google/employee/12)</p> <p>3. REST API 디자인 가이드</p> <p>수많은 디자인 가이드가 있지만, 실제로 개발하며 고려할 사항들만 정리했다.</p> <p>1) URL에 케밥케이스(kebab-case)를 사용하자</p> <p>orders의 목록을 반환하는 API를 만든다고 가정하면,</p>		



Bad :

```
[GET] /systemOrders
[GET] /system_orders
```

Good :

```
[GET] /system-orders
```

2) Path Variable에는 카멜케이스 사용하자

: 주문내역 조회 URL을 만든다고 가정할 때, 주문 내역이라는 Orders 하위에 orderID가 있을 것이다. Path Variable에 해당하는 orderId는 카멜케이스를 사용하자

Bad :

```
[GET] /orders/{order_id}
[GET] /orders/{OrderId}
```

Good :

```
[GET] /orders/{orderId}
```

3) Collection에는 단수가 아닌 복수를 사용하자

: 사용자들 전체를 조회하는 URL에 User의 집합이므로 복수를 사용

Bad :

```
[GET] /user
[GET] /User
```

Good :

```
[GET] /users
```

4) Collection으로 시작해서 Identifier로 끝내자

: 리소스 대신에 price라는 속성을 가리키는 것을 좋지 못하므로, 상점 목록이라는 Collection으로 시작해서 하위에 속한 하나의 shopID라는 Identifier로 끝내는 것처럼 수정

Bad :

```
[GET] /user
[GET] /User
```

Good :

```
[GET] /users
```

5) HTTP 메서드로 표현하자

: HTTP 메서드를 통해서 CRUD 중 무엇을 수행하는지 알 수 있게 디자인을 해야 한다.

- GET : 리소스 정보를 조회
- POST : 새로운 리소스를 생성
- PUT : 존재하는 리소스를 수정
- PATCH : 제공된 리소스를 업데이트

예시.

- GET /shops/2/products: shop 2에서 판매하는 모든 상품을 조회합니다.
- GET /shops/2/products/31: shop 2에서 판매하는 31번 상품을 조회합니다.
- DELETE /shops/2/products/31: shop 2에서 판매하는 31번 상품을 삭제합니다.
- PUT /shops/2/products/31: shop 2에서 판매하는 31번 상품의 정보를 수정합니다.
- POST /shops : 새로운 shop을 생성합니다.



6) ** 적절한 상태값을 응답하자 **(중요)

: 잘 설계된 REST API는 URL만 잘 설계된 것이 아니라, 리소스에 대한 응답을 잘 전달하는 것에 목적을 두고 설계해야 한다. 정확한 응답의 상태코드는 많은 정보를 전달할 수 있다. 응답 코드는 크게 5가지로 분류되며, 코드별로 제공하는 응답 종류가 다르다.

- Informational 1XX: 정보를 제공하는 응답
- Successful 2XX : 성공적인 응답
 - 200 OK: 요청이 성공적으로 수행
- Redirection 3XX
 - 메시지 클라이언트의 요청을 완료하기 위해서 추가적인 행동이 필요한 경우
- Client Error 4XX
 - 클라이언트가 서버에게 잘못된 요청을 하는 경우
 - 401 Unauthorized: 인증이 필요한 페이지를 요청한 경우
 - 403 Forbidden: 허용되지 않은 메소드가 있을 때
 - 406 Not Acceptable: 허용 불가능
- Server Error 5XX
 - 서버에서 오류가 발생하여 정상적으로 요청을 처리할 수 없는 경우
 - 500 Internal Server Error: 웹 서버가 처리할 수 없음
 - 503 Service Unavailable: 서비스 제공불가, 서버 과부하, 서버 폭주

4. 총정리

: 이번 세션에서는 통신 API를 디자인하는 방법에 대해 알아보는 시간을 갖게 되었습니다. 클라이언트 측에서 정보를 요청하는 URL을 디자인할 때 어떤 주의사항들을 고려해야 하는지, 서버에서 응답할 때 적절한 상태 코드를 담아주는 것을 생각해보도록 세션을 준비해보았습니다.

스터디를 진행하며, JSON 형식과 XML 형식으로 정보를 제공하는 것조차 모르는 팀원이 있었지만, 모르는 부분을 서로에게 질문하며 새로운 정보를 얻어갈 수 있는 시간이었습니다.

이제 5-6주차에는 실제 개발하고자 하는 웹 서비스에 대한 기능을 생각해보고, 지금까지 공부한 내용들을 활용하는 주차가 되고자 합니다.

이번 스터디에서 가장 어려웠던 부분은, API 설계를 하는 방법에 대해서는 이제 이해할 수 있었지만, 문서화를 하는 도구 Swaager 등을 백엔드 파트에서 코드에 적용하는 과정을 가져오지 않아 실제로 문서를 제작하는 과정이 어떻게 되는지 이해하기 어려웠다는 점입니다.

<div> 활동 후기 * 본인이 맡은 역할 활동과 학습한 내용에 대한 후기 작성 </div>	팀장 / 팀원 이름	작성란
	이영교	다른 팀원들의 회의 내용을 기록하며 해당 주차에 준비해온 프론트엔드 개발 지식들을 공유하는 역할을 했습니다. 이번 주차는 REST API 디자인 가이드를 주제로, 팀원이 개발을 진행하면서 한번쯤은 고려해보면 좋을 것 같은 내용을 가져왔습니다.
	김태현	이번 주차는 정적 콘텐츠와 MVC, 템플릿 엔진, API에 대해 공부하였습니다. 다른 팀원들과는 다르게 백엔드 개발 파트에 들어가게 되었는데 큰 어려움은 없었습니다.
	손민재	리엑트를 공부하기 전 자바스크립트를 공부해야 했는데, 변수/연산자/배열 등등 학교 수업을 통해 학습했던 C언어와 비슷한 점이 꽤 있어 수월하게



