

3주차 미션기록

▼ 요구사항 분석

홈화면

▼ 사용자 정보 불러오기 GET

1. 주소
2. 보유 포인트
3. 진행 완료 미션 개수

▼ ~~진행 중인 미션~~ GET

1. ~~미션~~ 아이디
2. ~~가게~~ 이름
3. ~~가게~~ 카테고리
4. ~~최소~~주문금액
5. 적립예정포인트
6. 마감기한

→ 미션 목록 조회 API를 같이 사용할 수도 있었지만,
홈 화면에서의 미션은 진행 중인 미션만 가져온다는 점과
같이 사용 하면 중복 되는 정보가 많지 않아 너무 많은 정보를 가져 온다는 점에서 분리
하는 것이 좋겠다고 생각함.

→ 생각을 다시 해보니,
두 API 모두 사용자별 미션이기 때문에 쿼리 스트링으로 구분하는 게 맞다고 생각
전체 미션을 가져와서 (진행중, 진행완료)로 구분

마이페이지 리뷰작성

▼ 리뷰작성 POST

1. 가게 아이디
2. 별점
3. 리뷰 내용
4. 사진

미션목록조회(진행중, 진행완료)

▼ 미션목록 조회 GET

1. 미션 아이디
2. 미션 상태(진행중, 진행완료, 실패)
3. 마감기한
4. 가게이름
5. 가게 카테고리
6. 최소 주문 금액
7. 적립 퍼센트
8. 적립예정포인트

미션 성공 누르기

▼ 미션 성공요청 GET

1. 미션 아이디
2. 사장님 구분 번호

▼ 미션 완료 POST

1. 미션 아이디

회원가입 하기(이메일)

▼ 약관 목록 불러오기 GET

1. 약관 아이디
2. 약관 제목
3. 약관 내용

▼ 회원 정보 입력 POST

1. 이름
2. 성별
3. 생년월일
4. 주소
5. 약관 동의 목록 [약관 아이디]

6. [선호 음식 카테고리]

▼ 음식 카테고리 불러오기 GET

1. 카테고리 아이디
2. 카테고리 이름

```
{
  "success": false,
  "code": "E5000",
  "message": "서버 내부 오류가 발생하였습니다",
  "result": null
}
```

```
{
  "success": true,
  "code": "SUCCESS200",
  "message": "성공입니다",
  "result": {

  }
}
```

API 명세서

🔍 분류	Aa Name	🔍 HTTP Method	≡ API Path	≡ Text
회원가입	<u>약관 목록 조회</u>	GET	/api/v1/terms	
	<u>음식 카테고리 목록 조회</u>	GET	/api/v1/food-category	
	<u>이메일 회원가입</u>	POST	/api/v1/users	
홈	<u>사용자 홈 조회</u>	GET	/api/v1/users/home	
미션	<u>미션 목록 조회</u>	GET	/api/v1/missions?status={status}	

🔍 분류	Aa Name	🔍 HTTP Method	≡ API Path	≡ Text
	<u>미션 성공 요청</u>	GET	/api/v1/missions/{missionId}/success	
	<u>미션 완료</u>	POST	/api/v1/missions/{missionId}/complete	
마이페이지	<u>리뷰 작성</u>	POST	/api/v1/reviews	
	<u>이미지 업로드</u>	POST	/api/v1/images	

Q. 쿼리스트링 별로 응답을 다르게 줄 수 있는가?

- 쿼리 스트링은 API의 Response Body 의 구조를 바꾸지 않음
- 쿼리스트링을 통해 API 내의 변화보다는, 서버 내부에서 SQL 쿼리를 다르게 주는 방식 등으로 **필터링** 을 하는 것

예시

```
GET /missions?status=inProgress
```

```
→ SELECT * FROM mission WHERE user_id = ? AND status = 'IN_PROGRESS';
```

```
GET /missions?status=completed
```

```
→ SELECT * FROM mission WHERE user_id = ? AND status = 'COMPLETED';
```

Q. 이미지를 클라이언트에서 서버로 보낼 경우


방법 1: 멀티파트로 내용과 이미지 함께 올리기

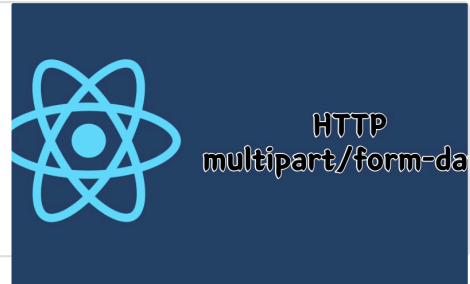
- 이미지 업로드를 API에 올리기 위해서는 JSON 으로는 불가능하고 multipart/form-data 를 사용해야함

- multipart/form-data 는 지정된 형식에 따라 메시지를 인코딩하여 보내는 방식
 - 이미지 파일을 파트별로 분리하여 그 개별 파일에 대한 정보를 보내는 것

HTTP multipart/form-data 란?

프로젝트를 진행하면서 프론트 -> 백엔드로 이미지를 전송하는 경우가 있었다.오늘은 HTTP, multipart, multipart/form-data 세 가지 키워드에 대해 알아보고, 그 중에서 중요한 개념중에 하나인

 <https://velog.io/@shin6403/HTTP-multipartform-data-%EB%9E%80>



POST /api/v1/reviews

Content-Type: multipart/form-data; boundary=----1234

-----1234

Content-Disposition: form-data; name="id"

1

-----1234

Content-Disposition: form-data; name="image"; filename="review.png"

Content-Type: image/png

(binary data)

-----1234--

방법 2: 이미지 먼저 업로드 후 URL 을 받은 뒤, JSON에 URL을 넣기

-- 이미지 업로드 요청 --

POST /api/v1/images

Content-Type: multipart/form-data

-----123

Content-Disposition: form-data; name="file"; filename="review.png"

Content-Type: image/png

(binary data)

-----123--

-- 이미지 업로드 응답 --

```
{
  "success": true,
  "code": "SUCCESS200",
  "message": "이미지 업로드 성공",
  "result": {
    "imageUrl": "https://cdn.myapp.com/uploads/123.png"
  }
}
```

-- 이미지와 함께 내용 올리기 --

```
{
  "name": "오현민",
  "images": [
    "https://cdn.myapp.com/uploads/123.png",
    "https://cdn.myapp.com/uploads/456.png"
  ]
}
```