Water Melon

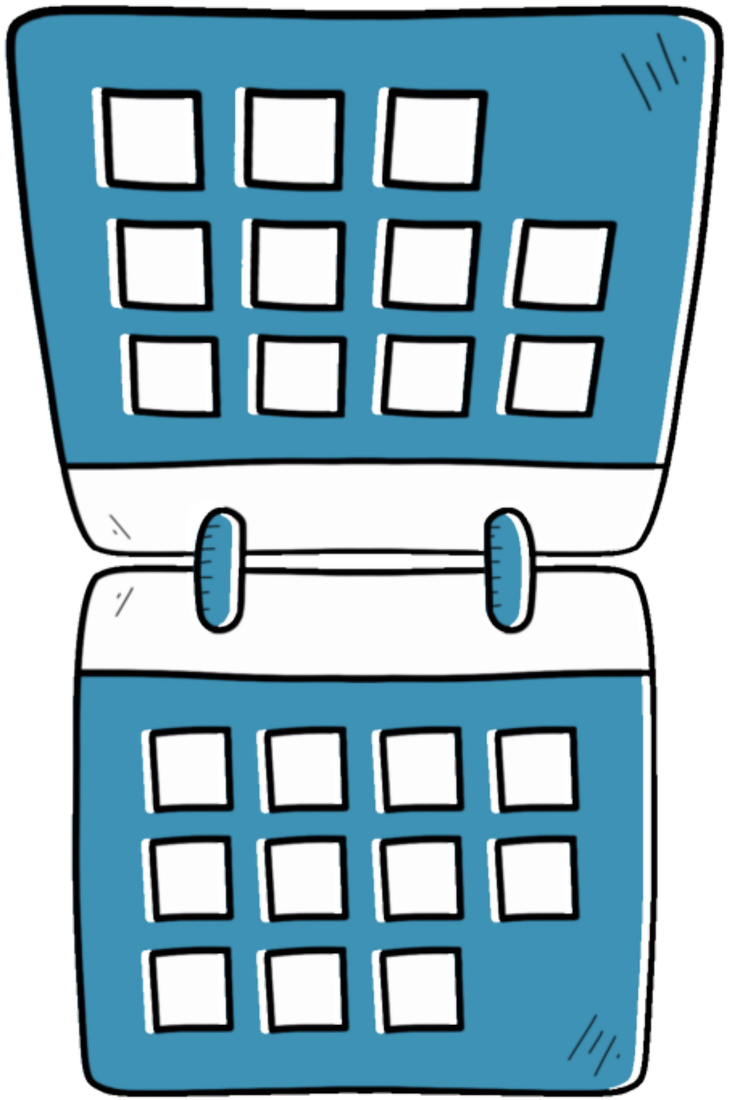
team Jodongary

음악이 필요할 땐

목차

개요

감정에 대해 공유하지 않는 최근 현대인들의 성향에 위로가 되어주고 공감해주는 노래를 추천해주는 봇을 만들고 싶었다.



과정

Day 01~02 주제선정

Day 02~06 코드제작

Day 06~08 Ppt제작

Day 08~11 검토/수정

Day 12 😆발표

Tool

API

팀원 소개

최효정

정인혁

권병진

코드

```
1     import EMOTION
2     import TITLE
3     from flask import Flask, request, jsonify
4     import random
5     import requests
6     import THUMBNAIL
7
8     application = Flask(__name__)
9
10    myreq1 = None
11    myreq2 = None
12    emo = EMOTION.EMOTION()
13    tit = TITLE.TITLE()
14    thu = THUMBNAIL.THUMBNAIL()
15
```

사용한 라이브러리와 호출값 간소화

코드

1

```python
@application.route("/hello", methods=["POST"])
def handle_request1():
    global myreq1
    req = request.get_json()
    print(req["userRequest"]["utterance"])
    myreq1 = req["userRequest"]["utterance"]

    if request.path == "/hello":
        message_text = "오늘은 어떤 음악을 들으실 건가요?"
        quick_replies = [
            {"messageText": "힙합", "action": "message", "label": "힙합"},
            {"messageText": "아이돌", "action": "message", "label": "아이돌"},
            {"messageText": "인디", "action": "message", "label": "인디"},
            {"messageText": "발라드", "action": "message", "label": "발라드"},
            {"messageText": "검색할래", "action": "message", "label": "직접입력으로 검색하기"}
        ]
        # else:
        #     return None
```

2

```python
        res = {
            "version": "2.0",
            "template": {
                "outputs": [
                    {
                        "simpleText": {
                            "text": message_text
                        }
                    }
                ],
                "quickReplies": quick_replies
            }
        }
        return jsonify(res)
    else:
        return jsonify({"error": "Invalid request path"})
```

hello스킬이 발화 시 quick_replies에 넣어둔 발화키를 선택지로 던져줌

코드

```
@application.route("/searching",methods=["POST"])
def seachingtap():
    req = request.get_json()
    myreq = req["userRequest"]["utterance"]

    res = {
            "version": "2.0",
            "template": {
                "outputs": [
                    {
                        "simpleText": {
                            "text": "검색어를 입력하세요."
                        }
                    }
                ]
            }
    }
    return jsonify(res)
```

3

hello스킬에서 던져준
선택지 중 검색하기를 선택시
searching스킬이 작동
검색어를 입력하라는
메시지를 발송

코드

```python
def youtube_search():
    req = request.get_json()
    search_query = req["userRequest"]["utterance"]
    search_query = '/youtube ' + search_query
    if search_query.startswith('/youtube '):
        search_term = search_query.replace('/youtube ', '')
        youtube_api_key = 'YOUTUBE DATA V3 API KEY'
        try:
            response = requests.get('https://www.googleapis.com/youtube/v3/search', params={
                'part': 'snippet',
                'q': search_term,
                'key': youtube_api_key
            })
            search_results = response.json()['items']
            if search_results:
                cards = []
                for item in search_results:
                    video_title = item['snippet']['title'].replace('&quot;','"')
                    video_thumbnail = item['snippet']['thumbnails']['default']['url']
                    video_id = item['id']['videoId']

                    card = {
                        'title': video_title,
                        'description': 'YouTube Video',
                        #'thumbnail': {
                        'imageUrl': video_thumbnail,
                        #},
                        'link': {
                            'web': f'https://www.youtube.com/watch?v={video_id}'
                        }
                    }
                    cards.append(card)
```

4

앞서 설명한 '3번 코드'에서
검색어를 입력 시
youtube api를 사용해
해당 내용을 유튜브에
검색해 결과 값을 카드 형태로
발송해줌

발송 내용에 관한 코드는
다음 장에

코드

5

```
            res = {
                "version": "2.0",
                "template": {
                    "outputs": [
                        {
                            "listCard": {
                                "header": {
                                    "title": f"\"{search_term}\" 검색 결과"
                                },
                                "items": cards
                            }
                        }
                    ]
                }
            }
            print(res)
            #res = res.replace('&quot;','"')
            return jsonify(res)
        else:
            return jsonify({'message': '검색 결과가 없습니다.'})
    except Exception as e:
        print('검색엔진 호출 중 오류:', e)
        return jsonify({'message': '검색엔진 호출 중 오류가 발생했습니다.'}), 500

return jsonify({'message': ''})
```

결과 값의 출력 형태와 앞서 설명한 '4번 코드'의 에러 시 출력 형태에 관한 코드

코드

```python
@application.route("/category", methods=["POST"])
def handle_request2():
    global myreq2
    req = request.get_json()
    print(req["userRequest"]["utterance"])
    myreq2 = req["userRequest"]["utterance"]

    if request.path == "/category":
        message_text = "오늘의 기분은 어떠신가요?"
        quick_replies = [
            {"messageText": "기쁨", "action": "message", "label": "기쁨"},
            {"messageText": "슬픔", "action": "message", "label": "슬픔"},
            {"messageText": "분노", "action": "message", "label": "분노"},
            {"messageText": "불안", "action": "message", "label": "불안"},
            {"messageText": "심심", "action": "message", "label": "심심"}
        ]
        # else:
        #     return None

        res = {
            "version": "2.0",
            "template": {
                "outputs": [
                    {
                        "simpleText": {
                            "text": message_text
                        }
                    }
                ],
                "quickReplies": quick_replies
            }
        }
        return jsonify(res)
    else:
        return jsonify({"error": "Invalid request path"})
```

6

'1번 코드'에서 검색하기가 아닌 다른 선택지를 선택했을 경우
발화 값을 global char변수인 myreq2에 저장하고
Category 스킬이 작동
6번 코드가 실행되어 현 감정을 묻는 5개의 선택지를 '2번 코드'와 같은 형태로 제공

코드

7

```
@application.route("/wm", methods=["POST"])
def handle_request3():
    global myreq1
    global myreq2
    req = request.get_json()
    print(req["userRequest"]["utterance"])
    myreq = req["userRequest"]["utterance"]
    rn = random.randint(0, 4)
    if myreq2 == '인디':
        if myreq == '기쁨':
            getUrl = emo.indiJoy(rn)
            getTitle = tit.indiJoy(rn)
            getImage = thu.indiJoy(rn)
            res = {
                "version": "2.0",
                "template": {
                    "outputs": [
                        {
                            'basicCard': {
                                'title': '이 노래는 어떠세요?',
                                'description': getTitle,
                                'thumbnail': {
                                    'imageUrl': getImage
                                },
                                'buttons': [
                                    {
                                        'action': 'webLink',
                                        'label': '들어보기',
                                        'webLinkUrl': getUrl
                                    }
```

'6번 코드'에서 제공한 감정 선택지에서 선택 시 WM스킬이 작동 '6번 코드'가 발화되며 저장한 myreq2의 값으로 이전 선택지의 선택 항목을 검수 후 해당하는 내용에 대해 차후 보여줄 Emotion.py에 list의 내용을 받아와 카드형태로 발송해줌

코드

```
elif myreq == '슬픔':
    getUrl = emo.indiSad(rn)
    getTitle = tit.indiSad(rn)
    getImage = thu.indiSad(rn)
    res = {
        "version": "2.0",
        "template": {
            "outputs": [
                {
                    'basicCard': {
                        'title': '이 노래는 어떠세요?',
                        'description': getTitle,
                        'thumbnail': {
                            'imageUrl': getImage
                        },
                        'buttons': [
                            {
                                'action': 'webLink',
                                'label': '들어보기',
                                'webLinkUrl': getUrl
                            }
                        ]
```

'7번 코드'의 연장
myreq의 값에 따라
다른 내용의 리스트를
Emotion.py에서 가져옴

다른 선택지 또한
이와같은 형태기에 생략

또한 모두 같은 형태기에 이후 업데이트를 통해
메서드화를 진행 예정

Emotion.py

```python
class EMOTION:
    def __init__(self):
        return None

    def indiJoy(self, rn):
        return indiJoySong[rn]

    def indiSad(self, rn):
        return indiSadSong[rn]

    def indiAnger(self, rn):
        return indiAngerSong[rn]

    def indiUnrest(self, rn):
        return indiUnrestSong[rn]

    def indiBoring(self, rn):
```

```python
    # 힙합
    def hipJoy(self, rn):
        return hipJoySong[rn]

    def hipSad(self, rn):
        return hipSadSong[rn]

    def hipAnger(self, rn):
        return hipAngerSong[rn]

    def hipUnrest(self, rn):
        return hipUnrestSong[rn]

    def hipBoring(self, rn):
```

```python
    # 아이돌
    def idolJoy(self, rn):
        return idolJoySong[rn]

    def idolSad(self, rn):
        return idolSadSong[rn]

    def idolAnger(self, rn):
        return idolAngerSong[rn]

    def idolUnrest(self, rn):
        return idolUnrestSong[rn]

    def idolBoring(self, rn):
```

```python
    # 발라드
    def balJoy(self, rn):
        return balJoySong[rn]

    def balSad(self, rn):
        return balSadSong[rn]

    def balAnger(self, rn):
        return balAngerSong[rn]

    def balUnrest(self, rn):
        return balUnrestSong[rn]

    def balBoring(self, rn):
        return balBoringSong[rn]
```

설명은 뒷장을 참고

Emotion.py

```
    def indiJoy(self, rn):
        indiJoySong = [
            # 완벽해 - 레터플로우
            'https://www.youtube.com/watch?v=56vTXXk5PSQ&pp=ygUZ66CI7YSw7ZSM66Gc7JqwIOyZhOuyve2VtA%3D%3D',
            # 예쁜 말 - 어나니머스 아티스트
            'https://www.youtube.com/watch?v=0Bk_8tEEYzA&pp=ygUZ7JiI7IGc66eQIOyWtOuCmOuLiOuouOyKpA%3D%3D',
            # UP DOWN - 어나니머스 아티스트
            'https://www.youtube.com/watch?v=nw_o_qymHoM&pp=ygUl7Ja064KY64uI66i47IqkIOyVhO2LsOyKpO2KuCB1cCBkb3duIA%3D%3D',
            # 뭘 믿고 그렇게 이쁜거니 - 레터플로우
            'https://www.youtube.com/watch?v=DnVm2LENTec&pp=ygUw662Y66-_6rOgIOq3uOugh-qyjCDsnbTsgZzqsbDri4gg66CI7YSw7ZSM66Gc7Jqw',
            # ocean view - 로시
            'https://www.youtube.com/watch?v=5wiW60inhgw&pp=ygUN66Gc7IucIOyYpOyFmA%3D%3D'
        ]
        return indiJoySong[rn]
```

앞서 보여준 Emotion.py의 코드는 위와 같은 형태로 장르와 감정으로 나뉘어져 리스트 형태로 제작되어 있다.

이를 통해 '6번 코드'에서 랜덤 변수를 매게 변수로 던져주며 리스트의 값 중 하나를 랜덤하게 가져가게 된다.

Title.py

```python
class TITLE:
    def __init__(self):
        return None

    def title(self, song, rn):
        # 인디
        print('song:' + song)
        print('rn:')
        print(rn)

    def indiJoy(self, rn):
        indiJoy = [
            '완벽해 - 레터플로우',
            '예쁜 말 - 어나니머스 아티스트',
            'UP DOWN - 어나니머스 아티스트',
            '뭘 믿고 그렇게 이쁜거니 - 레터플로우',
            'ocean view - 로시'
        ]
        return indiJoy[rn]
```

Thumbnail.py

```python
class THUMBNAIL:
    def __init__(self):
        return None

    def indiJoy(self, rn):
        indiJoy = [
            'https://image.bugsm.co.kr/album/images/500/200888/20088894.jpg',
            'https://image.bugsm.co.kr/album/images/500/203321/20332177.jpg',
            'https://image.bugsm.co.kr/album/images/500/203030/20303087.jpg',
            'https://image.bugsm.co.kr/album/images/500/5371/537150.jpg',
            'https://mblogthumb-
phinf.pstatic.net/MjAyMDA4MjBfMTI5/MDAxNTk3OTMzNzI0MTQz.O1YJ0z6M0Sozdku2GX6tGLR
no121/1.PNG?type=w800'
        ]
        return indiJoy[rn]
```

Title.py와 Thumbnail.py 또한 Emotion.py와 같은 형태임으로

설명은 앞선 페이지를 참조해주시기 바랍니다.

같은 형태이기에 예시 리스트 하나만 보여드리는 점 양해 바랍니다.

카카오 챗봇

챗봇의 블럭
및
엔티티 지정

타 블럭 또한 비슷한
형태이기에 생략

카카오 챗봇

챗봇의 엔티티 설정

gohome	집
travel	여행
rest	휴식
party	파티
rain	비
spring	봄
summer	여름
authmn	가을
winter	겨울

엔티티 예시

카카오 챗봇

스킬 목록

API를 연결하여 동작 파라미터에 반응하는 응답형식을 설계합니다.

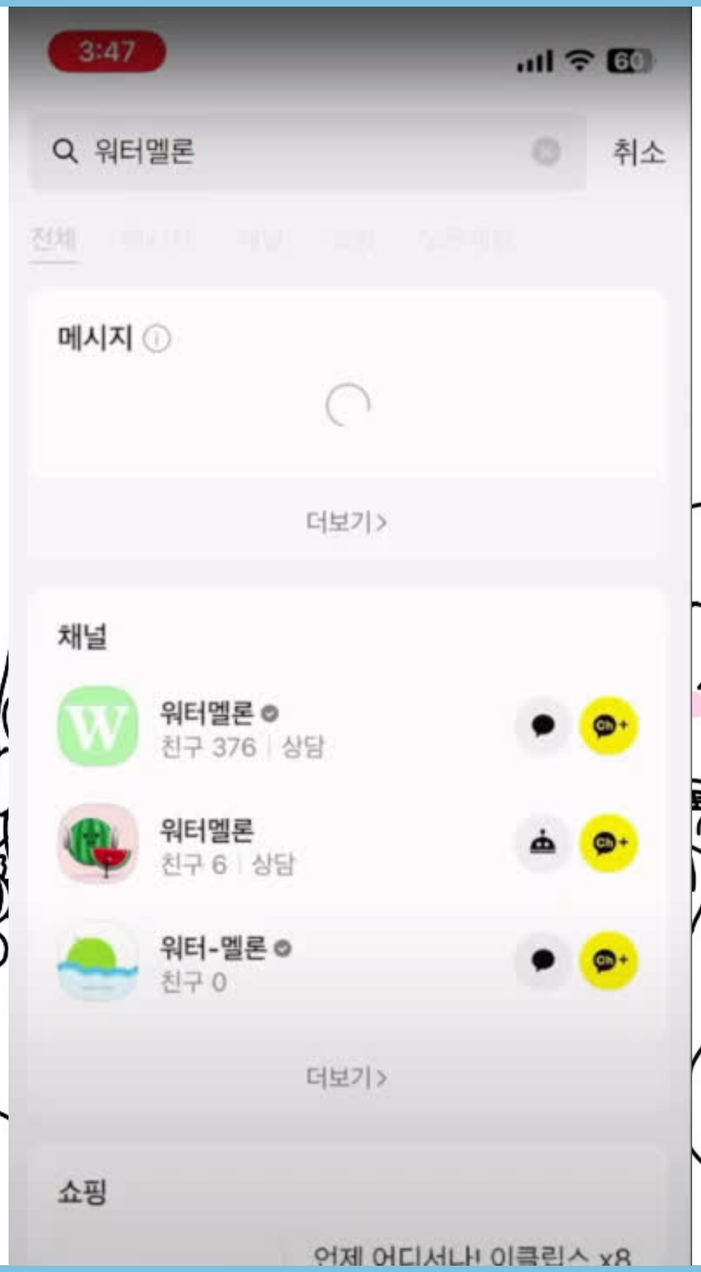
전체

스킬명	버전	적용 블록수
searching	1	1
youtube	1	2
wm	1	1
sayCategory	1	1
sayHello	1	1

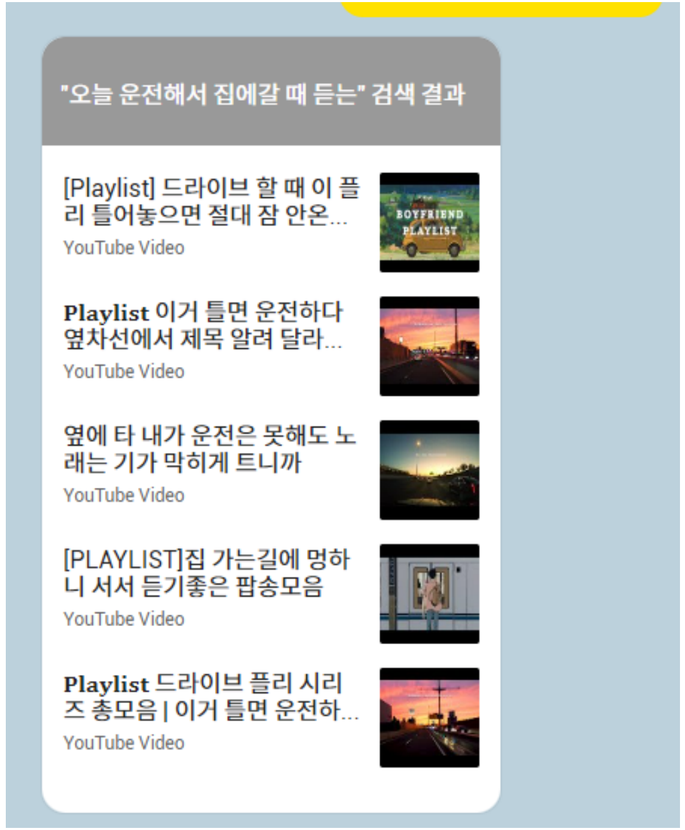
챗봇의 스킬
각 역할에 맞춰
블럭을 지정해 할당해
주었다.

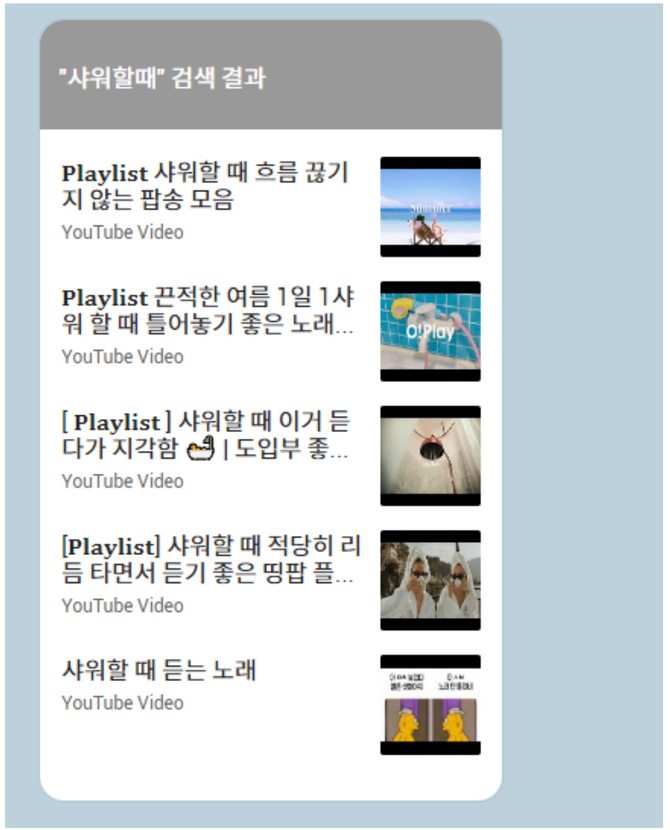
시연 영상


병진


3:47
워터멜론
취소
전체
메시지
더보기>
채널
워터멜론
친구 376 | 상담
워터멜론
친구 6 | 상담
워터-멜론
친구 0
더보기>
쇼핑

누락 자료(영상) 사진으로 대체





youtube data api v3의 사용 할당량 초과로 인해 다시 촬영이 불가능하여 사진으로 대체함을 양해 부탁드립니다.

※ 타 계정으로 발급받고 바꿔주면 가능하니 궁금하신 분은 후에 개발자들 git이 있으니 문의바람

차후 계획

앞서 서술한 선택지 호출의 메서드화와
제작해둔 openAI API와 Asyncia API를 사용한 gpt에게
대신 질문해주고 그 답을 웹훅 기능을 구현하여 받아오는 코드를
추가할 예정

제작해둔 코드는 다음 페이지에

추가 예정 코드

```python
@application.route("/webhook/", methods=["POST"])
def webhook():
    global a
    request_data = json.loads(request.get_data(), encoding='utf-8')
    a[request_data['user']] = request_data['result']['choices'][0]['message']['content']
    return 'OK'

@application.route("/question", methods=["POST"])
def get_question():
    global a
    request_data = json.loads(request.get_data(), encoding='utf-8')
    response = { "version": "2.0", "template": { "outputs": [{
        "simpleText": {"text": f"GPT에게 물어보는 중입니다.: {request_data['action']['params']['question']}"}
    }]}}
    a[request_data['userRequest']['user']['id']] = '아직 GPT가 처리중입니다.'
    try:
        api = requests.post('https://api.asyncia.com/v1/api/request/', json={
            "apikey": "<sk-[redacted]>",
            "messages" :[{"role": "user", "content": request_data['action']['params']['question']}],
            "userdata": [["user", request_data['userRequest']['user']['id']]]},
            headers={"apikey":"[redacted]"}, timeout=0.3)
    except requests.exceptions.ReadTimeout:
        pass
    return jsonify(response)
```

OPEN AI API KEY

ASYNCIA API KEY

추가 예정 코드

```
@application.route("/ans", methods=["POST"])
def hello2():
    request_data = json.loads(request.get_data(), encoding='utf-8')
    response = { "version": "2.0", "template": { "outputs": [{
        "simpleText": {"text": f"답변: {a.get(request_data['userRequest']['user']['id'], '질문을 하신적이 없어보여요. 질문부터 해주세요')}"}
    }]}}
    return jsonify(response)
```

제작해둔 코드를 해당 프로젝트에 추가하지 않은 이유는
Asyncia API가 악의적 공격을 막기 위해 초기 상용 데이터의 양을
제한해두어 용량이 작기에 주기적으로 사용하며 추가 데이터를
신청해야 하는 방식이기에 사용 가능한 데이터가 현재 너무 적어
추가하지 않았음