



파이썬 프로젝트

챗봇을 활용한 도서추천 챗봇

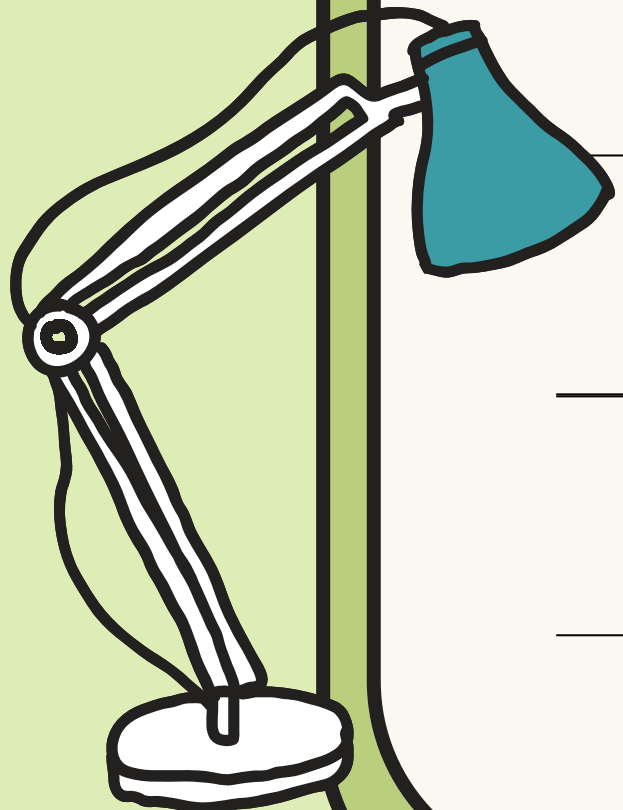
이.유.백

PageTalk



프로젝트 목차

목차	목차 내용	PAGE
01	팀원 구성원 및 역할	3쪽 ~ 4쪽
02	프로젝트 개요	5쪽 ~ 6쪽
03	프로젝트 절차	7쪽 ~ 26쪽
04	프로젝트 결과	27쪽 ~ 30쪽
05	팀원 자체평가	31쪽 ~ 32쪽



01

팀원 구성원 및 역할



PageTalk

열심히 들어봐요~!

팀구성원 및 역할



이현오



코드 총괄
및
엔티티 / 발화 작업

백정운



챗봇/ 채널 디자인
및
PPT 제작

유지웅



챗봇 기능 추가
및
PPT 발표

프로젝트 개요



프로젝트 개요

주제 선정 이유

인공지능 챗봇이 시간이 갈수록 무궁무진하게 활용되어 가고 있는 추세 속에 챗봇을 어렵게 만드는 것이 아닌 남녀노소 누구나 '편리하게' 사용할 수 있도록 제작 하고있다.

그래서 우리는 실생활에서 자주 접할 수 있는 책과 챗봇을 혼합하여 현재 사용자의 감정을 파악하여 그에 맞는 책들을 사용자한테 '편리하게' 추천을 해줄 수 있는 챗봇을 제작하기로 했다.

PageTalk



03

프로젝트 절차

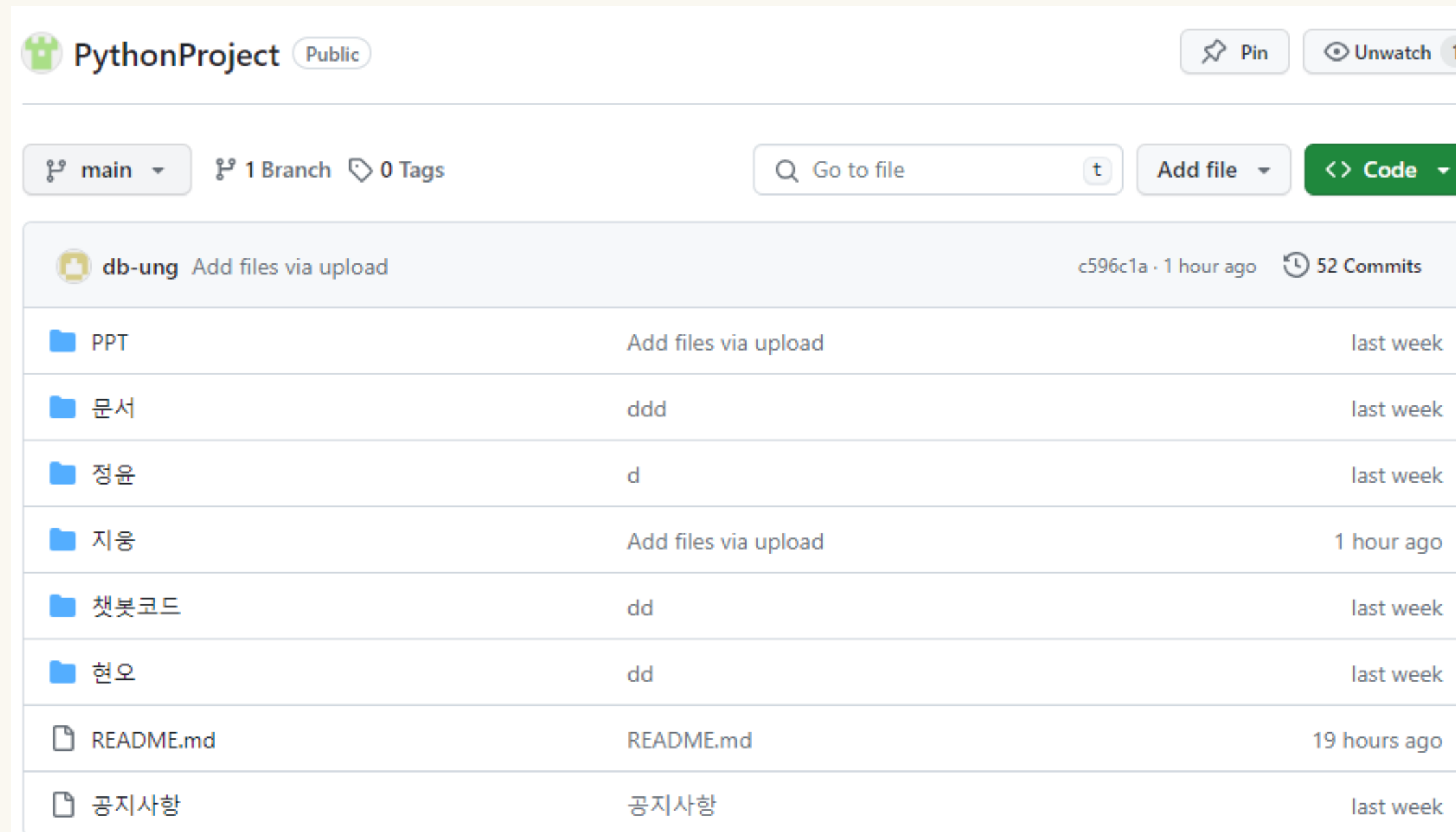


PageTalk

참고 프로그램 및 API



참고 프로그램 및 API



**코드 내용 업로드 및
파일 업로드 협업은 깃허브를
활용하였습니다.**

<https://github.com/JeongYoonBaek/PythonProject>

프로젝트 절차

12	DECEMBER					2023
Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
				1	2	3
4	5	6	7	8	9	10
주제 선정		소스코드 작성 및 챗봇 서버 구축				
11	12	13	14	15	16	17
코드 및 기능 수정		발표자료 준비		발표		

프로젝트 절차

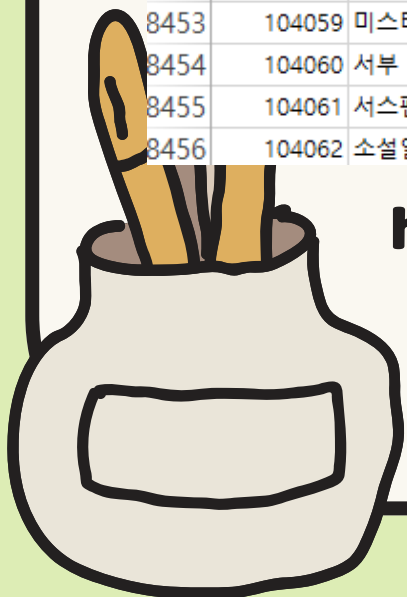
카테고리 분류 작업

8431	97612	코미디	외국도서	예술/대중문화	코미디		
8432	97613	혼합미디어	외국도서	예술/대중문화	혼합미디어		
8433	97614	TV/라디오	외국도서	예술/대중문화	TV/라디오		
8434	104055	라디오	외국도서	예술/대중문화	TV/라디오	라디오	
8435	105461	역사/비평	외국도서	예술/대중문화	TV/라디오	라디오	역사/비평
8436	105462	일반	외국도서	예술/대중문화	TV/라디오	라디오	일반
8437	105463	참고자료	외국도서	예술/대중문화	TV/라디오	라디오	참고자료
8438	105464	가이드/리뷰	외국도서	예술/대중문화	TV/라디오	텔레비전	가이드/리뷰
8439	105465	시나리오	외국도서	예술/대중문화	TV/라디오	텔레비전	시나리오
8440	105466	역사/비평	외국도서	예술/대중문화	TV/라디오	텔레비전	역사/비평
8441	105467	연출/제작	외국도서	예술/대중문화	TV/라디오	텔레비전	연출/제작
8442	105468	일반	외국도서	예술/대중문화	TV/라디오	텔레비전	일반
8443	105469	참고자료	외국도서	예술/대중문화	TV/라디오	텔레비전	참고자료
8444	104056	텔레비전	외국도서	예술/대중문화	TV/라디오	텔레비전	
8445	90849	오디오북	외국도서	오디오북			
8446	97615	건강/운동	외국도서	오디오북	건강/운동		
8447	97616	기타 논픽션	외국도서	오디오북	기타 논픽션		
8448	97617	명상	외국도서	오디오북	명상		
8449	97618	비즈니스/직업	외국도서	오디오북	비즈니스/직업		
8450	97619	소설/시/희곡	외국도서	오디오북	소설/시/희곡		
8451	104057	로맨스	외국도서	오디오북	소설/시/희곡	로맨스	
8452	104058	문학고전	외국도서	오디오북	소설/시/희곡	문학고전	
8453	104059	미스터리	외국도서	오디오북	소설/시/희곡	미스터리	
8454	104060	서부	외국도서	오디오북	소설/시/희곡	서부	
8455	104061	서스펜스/스릴러	외국도서	오디오북	소설/시/희곡	서스펜스/스릴러	
8456	104062	소설일반	외국도서	오디오북	소설/시/희곡	소설일반	

[https://docs.google.com/document/d/1mX-WxuoGs8Hy-](https://docs.google.com/document/d/1mX-WxuoGs8Hy-QalHhcvuV17n50uGI2Sg_GHofgiePE/edit)

[QalHhcvuV17n50uGI2Sg_GHofgiePE/edit](https://docs.google.com/document/d/1mX-WxuoGs8Hy-QalHhcvuV17n50uGI2Sg_GHofgiePE/edit)

**알라딘에서는 20000개에 가까운
도서 카테고리가 존재하는데,
팀원들은 여러 감정에 적합한 카테고리 분류
작업을 하였고 그 중에서도 최소한의 책
(데이터)들이 등록되어 있어야
챗봇에 활용할 수 있어서 분류 작업에
특히 신경을 썼음.**



프로젝트 절차

챗봇 코드

```
import json # json 라이브러리 추가
import urllib.request # url라이브러리요청
import sys
import random

class HappyRandomAPI:
    def __init__(self):
        return None
    def ranRandomHappy(self):
        # happyUrlList에 API request URL을 넣는다
        base_url = 'https://www.aladin.co.kr/ttb/api/ItemList.aspx?ttbkey=ttbbjy837414'

        # 다른 부분인 categoryID만 다르기때문에 따로 리스트로 만들어 저장해준다
        categoryIds = ['103712', '103721', '53515', '53536', '8546']

        # 그후 한 리스트에 넣어준다
        happyUrlList = [base_url + category_id for category_id in categoryIds]

        # 랜덤중 하나 뽑기
        randomHappyUrl = random.choice(happyUrlList)
```

HappyRandomAPI.py

**알라딘 API Key는 URL 형식에
맞게 작성한 후 책의 카테고리 ID를
감정 별로 리스트에 저장하고,
랜덤함수를 사용하여 임의로
데이터 하나를 출력한다.**



프로젝트 절차

챗봇 코드

```
# response 변수에 요청한 URL 열고 변수에 대입
response = urllib.request.urlopen(randomHappyUrl)

# 읽을때 UTF8로 디코딩하는이유 한글이 깨질 수있음
response = response.read().decode('utf8')

# response 데이터를 json 형태로 바꿈
data = json.loads(response)

happyres = data['item']

return happyres
```

HappyRandomAPI.py

**응답한 값의 URL을 변수에 대입하고,
데이터를 읽을 때 한글을 깨지는 것을
방지하기 위해 UTF8 디코딩 설정 후
데이터 형태를 JSON으로 변환 시킨다.**



프로젝트 절차

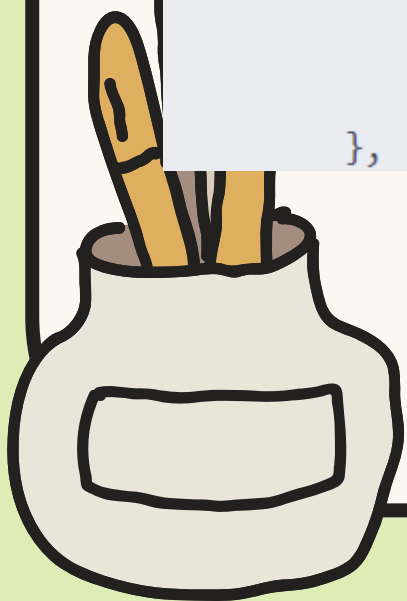
챗봇 코드

```
num = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
i = random.choice(num)
```

```
"listCard": {
  "header": {
    "title": "행복/기쁨에 따른 책 추천."
  },
  "items": [
    {
      "title": happydata[i]['title'], #happydata['item']['title']
      "description": happydata[i]['author'], # happydata['item']['author']
      "imageUrl": happydata[i]['cover'], #happydata['item']['cover']
      "link": {
        "web": happydata[i]['link'] #happydata['item']['link']
      }
    }
  ],
}
```

Application.py

**카테고리별 20권중 3권을 추천하기
때문에 리스트에 값을 저장한 후 랜덤
변수 생성하고, 리스트 카드 형태로
사용자에게 랜덤으로 책의 정보 및 구매
링크들을 출력해준다.**



프로젝트 절차

챗봇 코드

**국립중앙도서관을 크롤링하여
책 제목과 저자 책 커버를
Book 리스트에 저장 하여서
반환시켜준다.**

```
def search_books(self, keyword):
    url = f"https://www.nl.go.kr/NL/contents/search.do?srchTarget=total&pageNum=1&pageSize=10&insiteschStr=&schQuery=&mainSrchField=18"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4849.75"}

    original_html = requests.get(url, headers=headers)
    soup = BeautifulSoup(original_html.text, "html.parser")

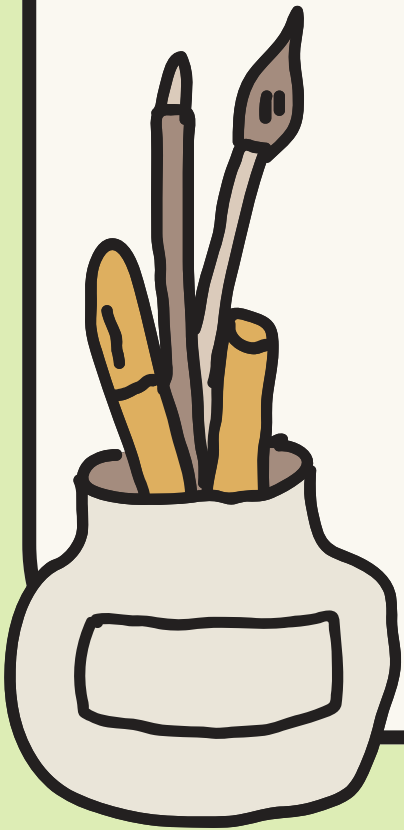
    book = {}
    article = []
    mytag = {}

    try:
        mytag = soup.select("div.cont_list.list_type > div.row")[0]
        article = mytag.select("span.txt_left.row_txt_tit > a")[0]
    except Exception as e:
        print(e)
    return book
```

```
book['href'] = article['href']
book['title'] = article.get_text(strip=True)
author = mytag.select("span.mr.txt_grey")
author = author[0].get_text()
cover = mytag.select("div.row_img_wrap>img")[0]
cover = cover['src']
book['author'] = author
book['cover'] = cover

return book
```

SearchBook.py



프로젝트 절차

챗봇 코드

```
# 크롤링해서 국립중앙도서관 자료검색
@appapplication.route('/search', methods=['POST'])
def chatbot_search_books():

    data = request.get_json()
    user_input = data['userRequest']['utterance']

    # 국립중앙도서관 도서검색
    d = BookSearcher()
    searchBookData = d.search_books(user_input)
    book = searchBookData
```

Application.py

**SearchBook.py에서
search_books함수를 호출 한 후
사용자가 발화한 것을 매개변수로 받는다**



프로젝트 절차

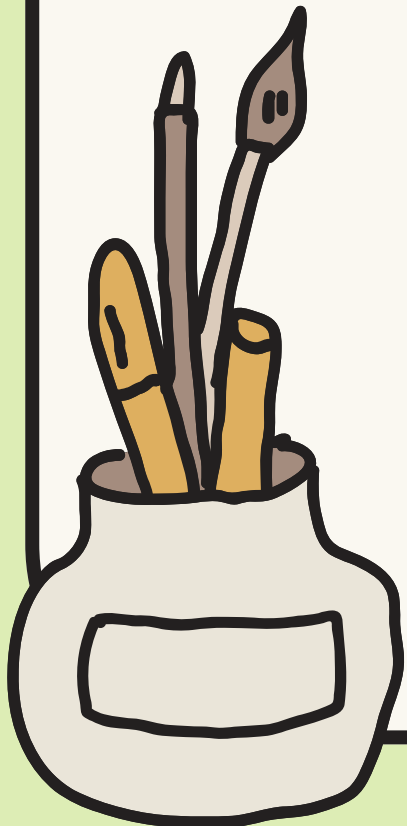
챗봇 코드

**사용자 발화를 크롤링하여 검색했을 때
검색결과가 없을 경우는 텍스트형식으로
“검색 결과가 없습니다”를 출력해주고
검색결과가 있을 때는 기본카드형식으로
책제목,저자,책커버이미지를 출력해준다**

```
if not book:
    # 검색 결과가 없을 경우 응답
    res = {
        "version": "2.0",
        "template": {
            "outputs": [
                {
                    "simpleText": {
                        "text": "검색 결과가 없습니다."
                    }
                }
            ]
        }
    }
```

```
else:
    # 기본 카드로 변경
    res = {
        "version": "2.0",
        "template": {
            "outputs": [
                {
                    "basicCard": {
                        "title": book["title"],
                        "description": book["author"],
                        "thumbnail": {
                            "imageUrl": book["cover"],
                        }
                    }
                }
            ]
        }
    }
    return jsonify(res)
```

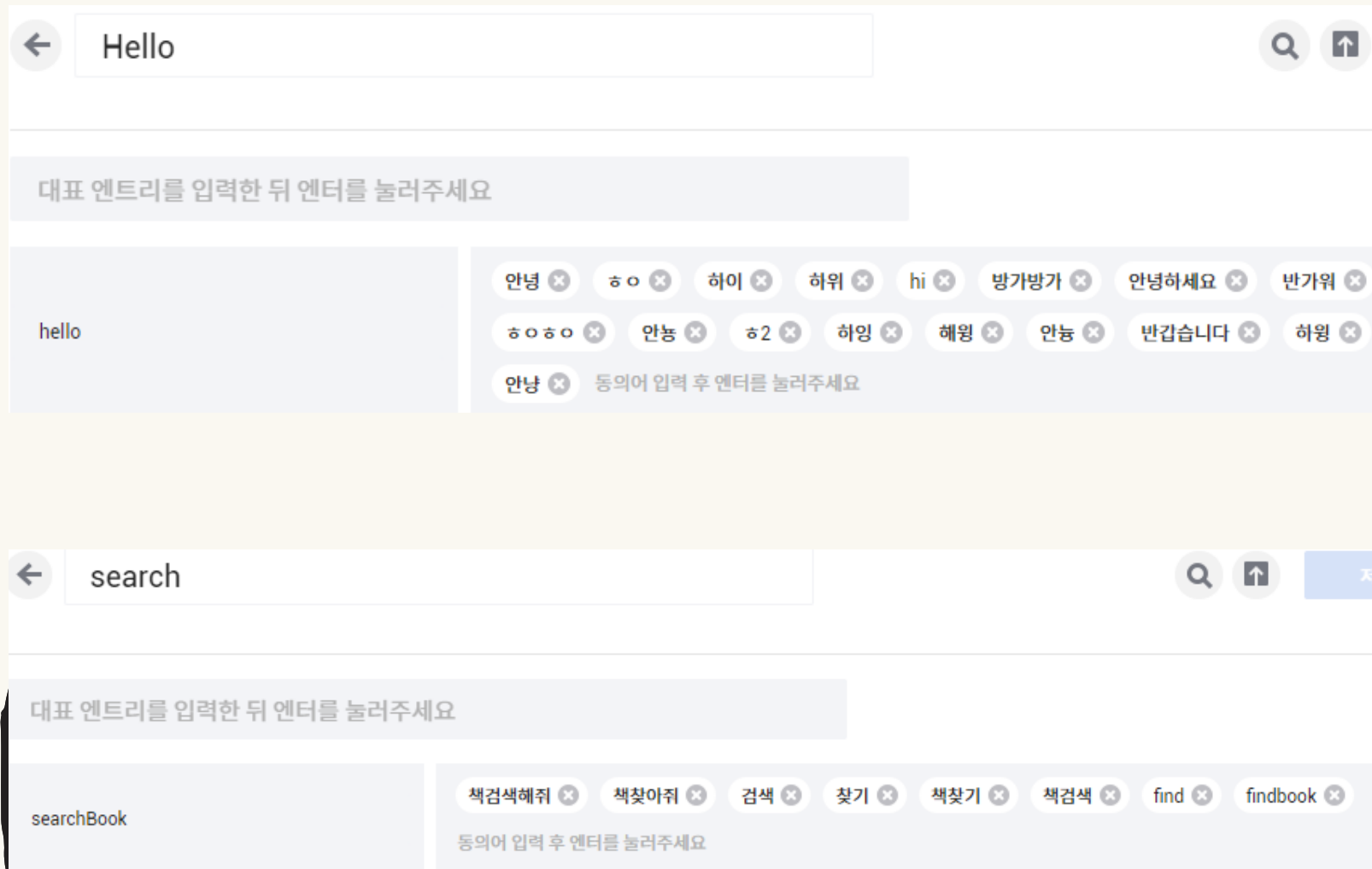
Application.py



프로젝트 절차

챗봇 엔티티

**사용자들이 충분히 말할 수 있는
키워드들을 직접 엔티티에 저장하여
키워드가 발화될 때 다음 동작으로
실행된다.**



← Hello 🔍 ↗

대표 엔트리를 입력한 뒤 엔터를 눌러주세요

hello

안녕 × ㅇㅇ × 하이 × 허위 × hi × 방가방가 × 안녕하세요 × 반가워 ×
ㅇㅇㅇㅇ × 안녕 × ㅇ2 × 허잉 × 해킹 × 안녕 × 반갑습니다 × 허잉 ×
안녕 × 동의어 입력 후 엔터를 눌러주세요

← search 🔍 ↗ 저장

대표 엔트리를 입력한 뒤 엔터를 눌러주세요

searchBook

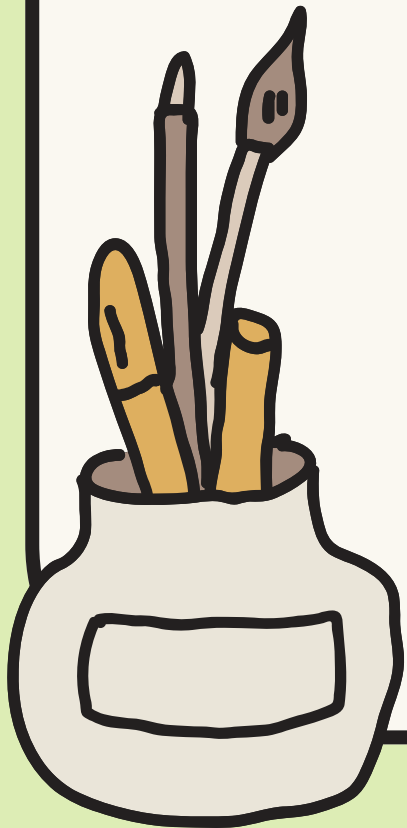
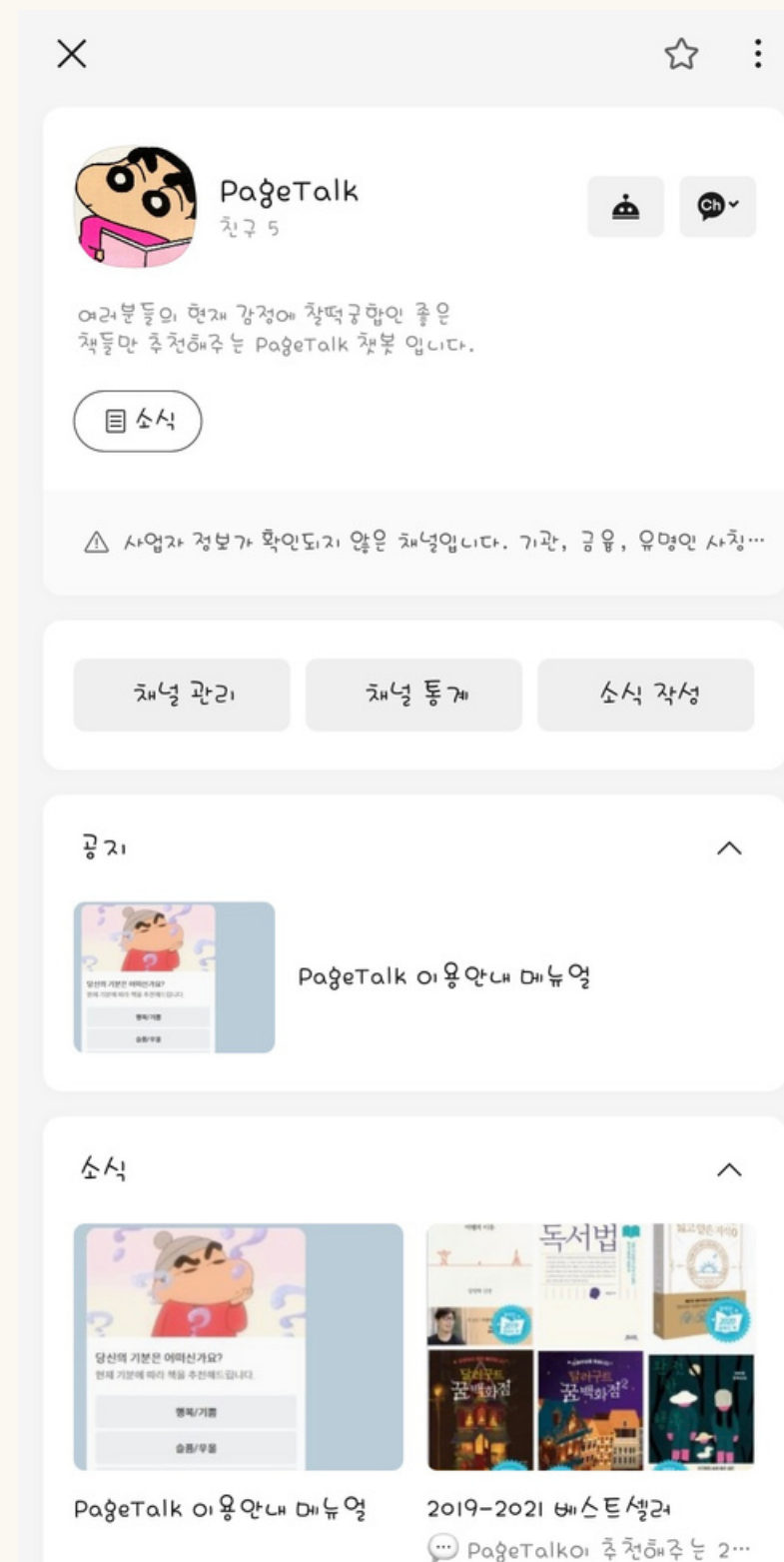
책검색해줘 × 책찾아줘 × 검색 × 찾기 × 책찾기 × 책검색 × find × findbook ×
동의어 입력 후 엔터를 눌러주세요



프로젝트 절차

챗봇 메인화면

**카카오톡 실행 후 PageTalk
검색 및 추가하면
다음과 같은 챗봇 메인화면이 출력된다.**



프로젝트 절차

챗봇 웰컴 메시지

**챗봇을 친구 추가 후 채팅창
화면으로 넘어가면
챗봇이 웰컴 메시지를 사용자에게
띄어준다.**



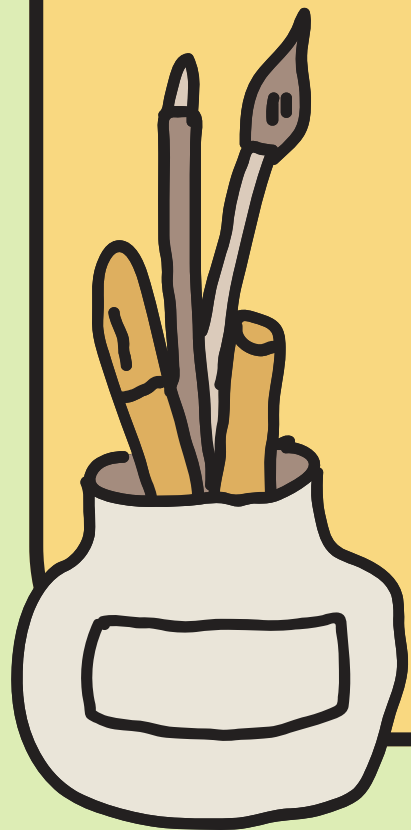
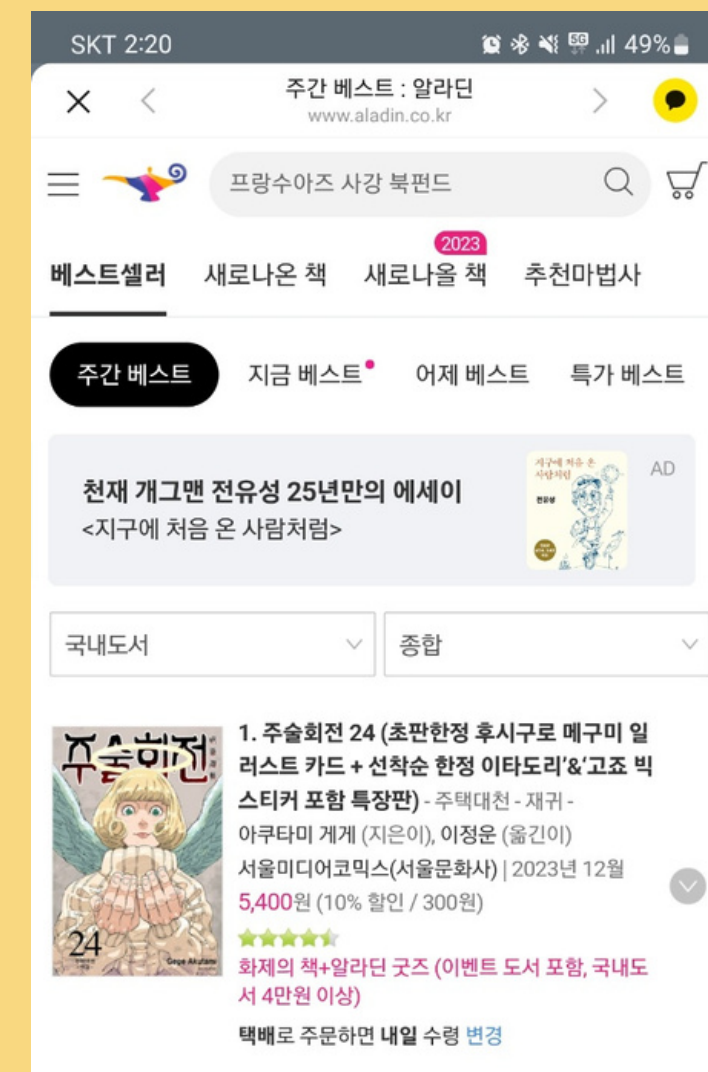
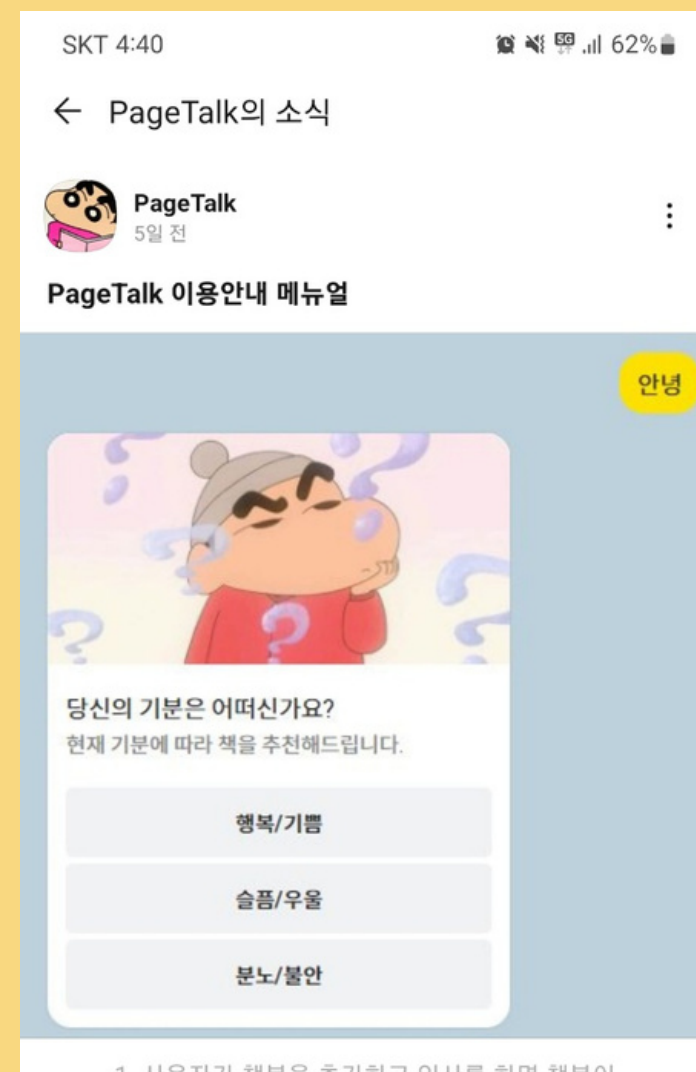
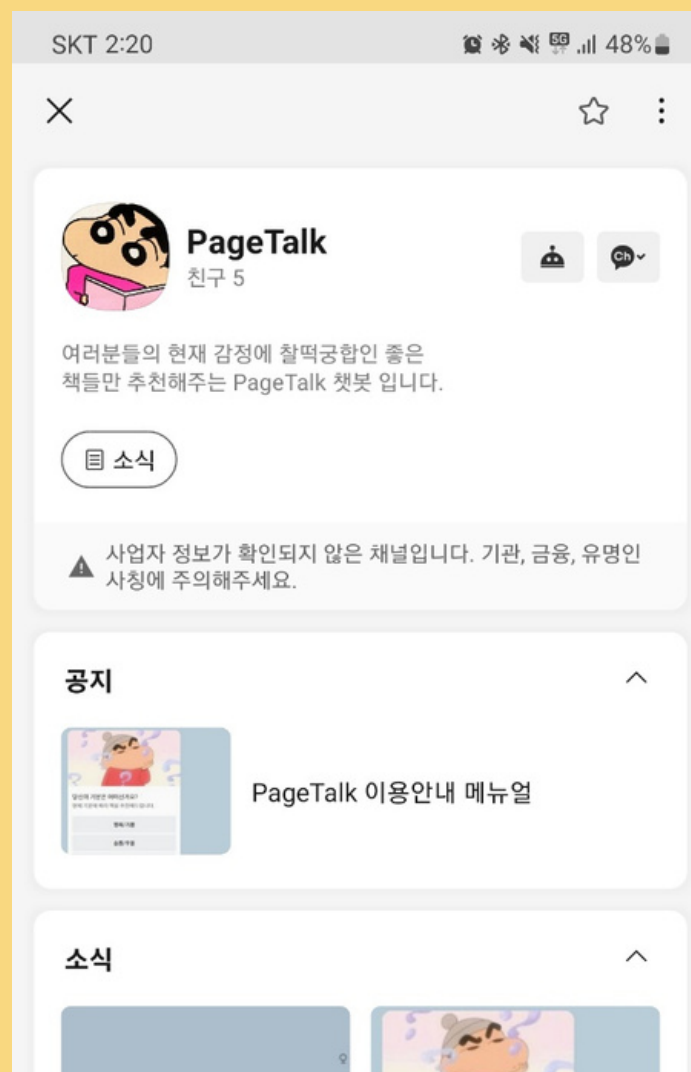
프로젝트 절차

채널홈

이용안내

베스트셀러

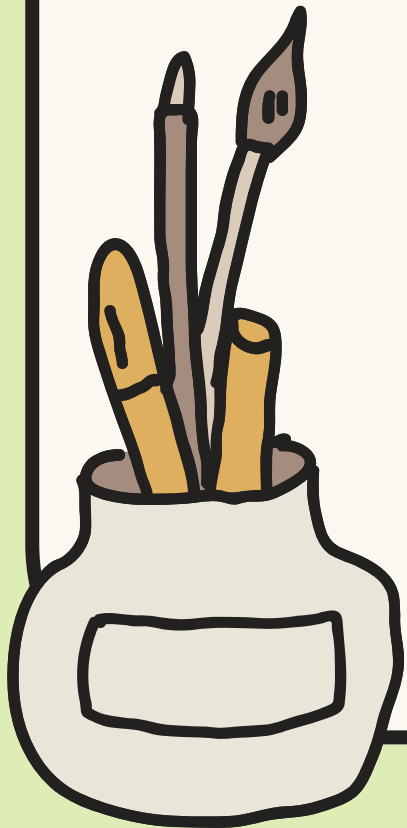
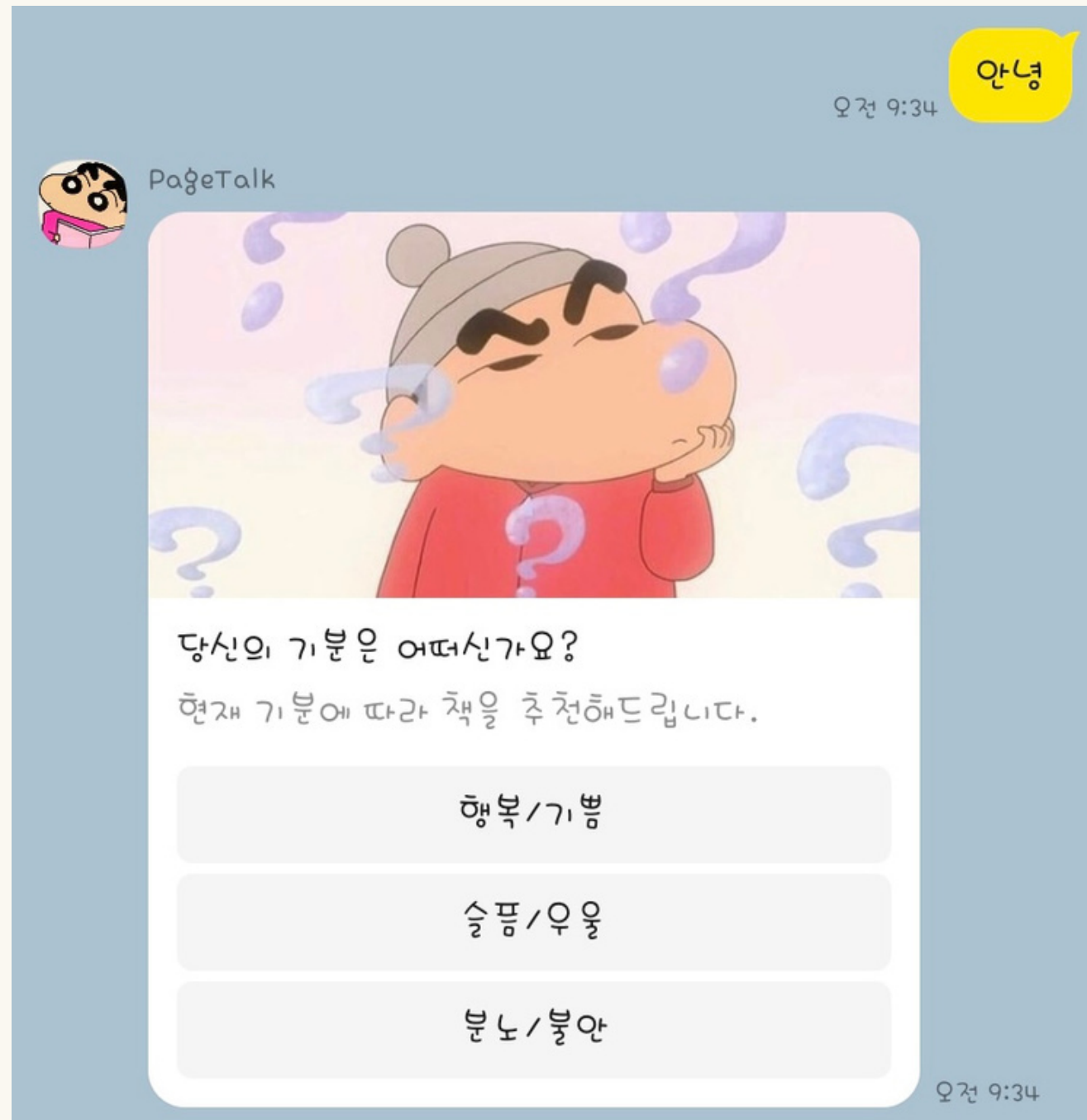
챗봇 안내 메시지



프로젝트 절차

챗봇 감정 질의

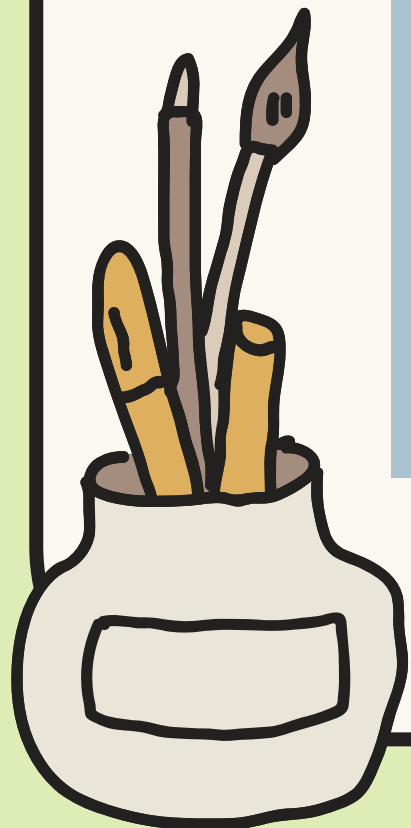
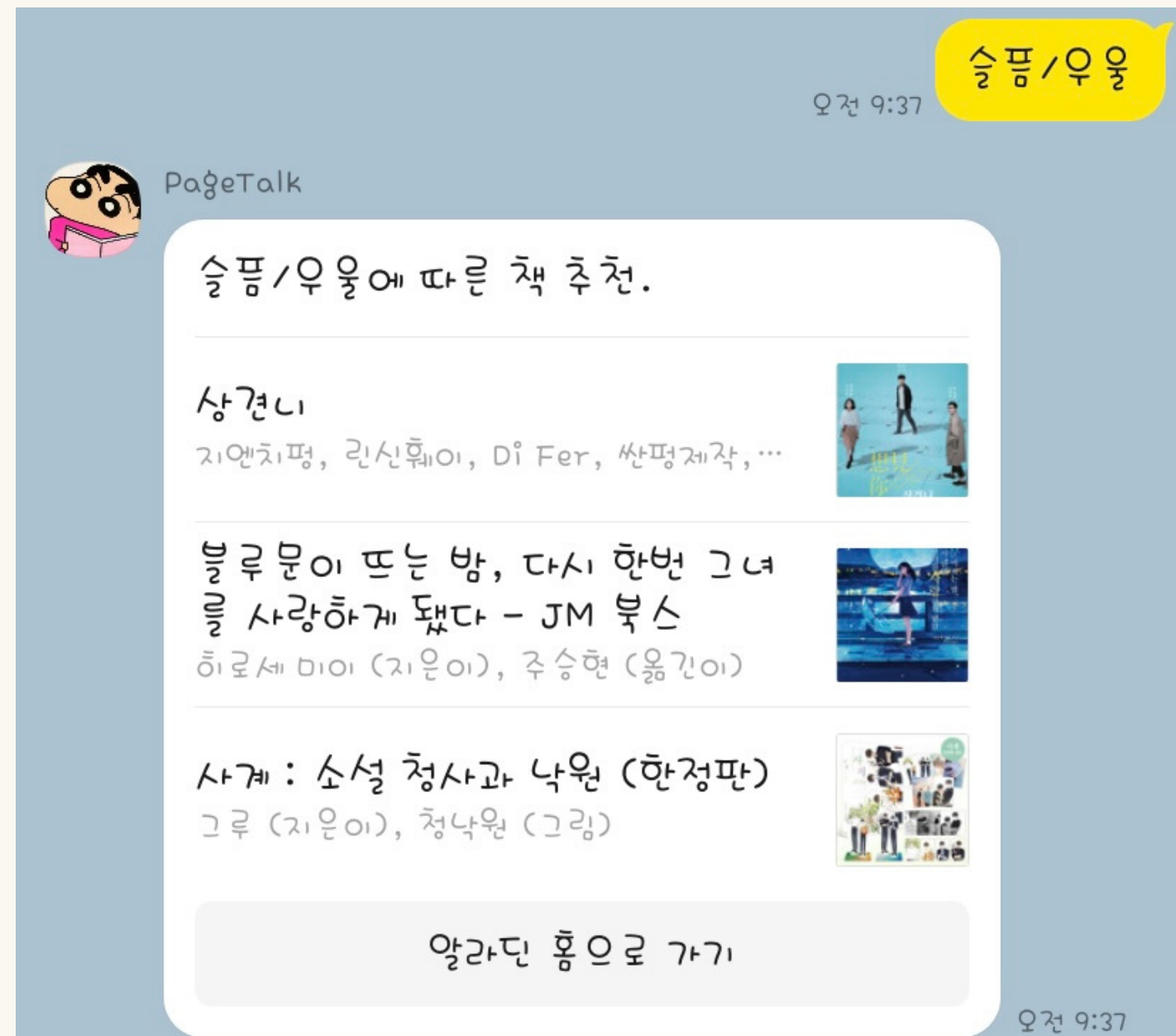
**사용자가 챗봇에게 인사를 건네면
챗봇은 사용자의 발화를 인식하여
사용자에게 감정질의 메시지를
출력한다.**



프로젝트 절차

챗봇 도서추천

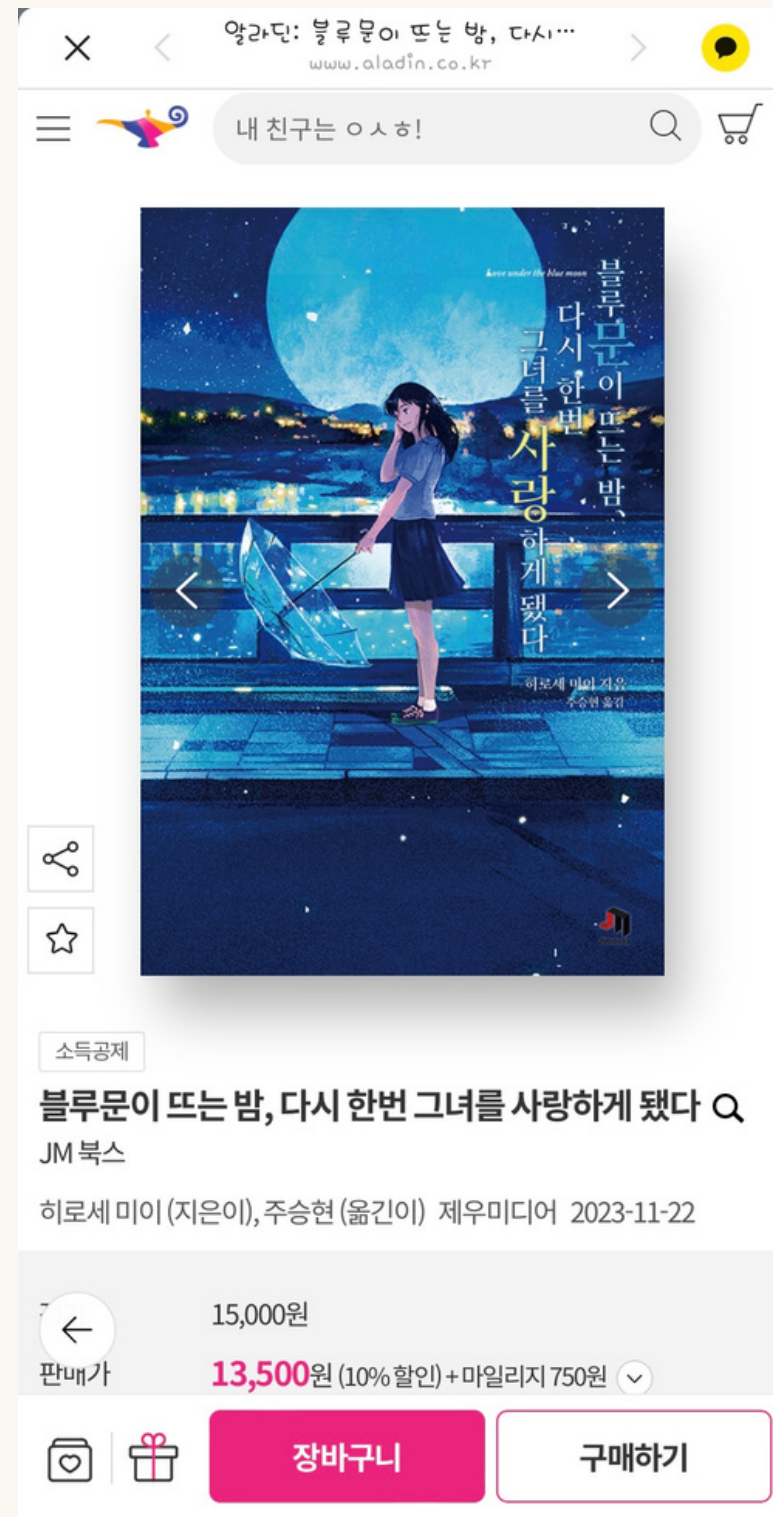
**사용자가 챗봇의 감정 질의에 응답을
하게 되면 챗봇이 랜덤으로
사용자의 감정에 맞는 도서를
추천 해준다.**



프로젝트 절차

챗봇 도서 구매창

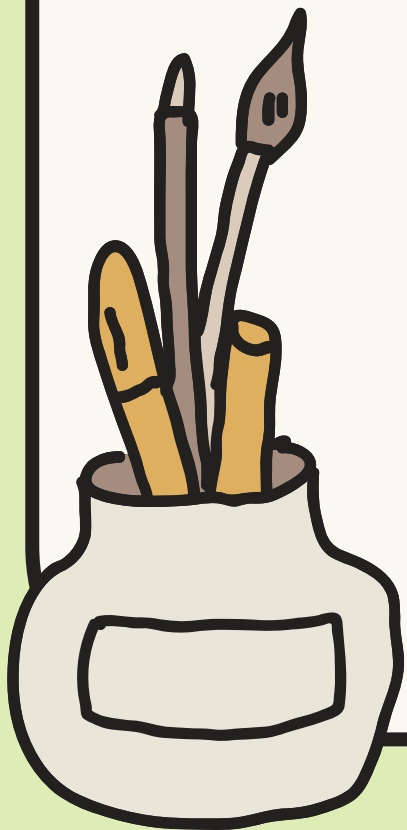
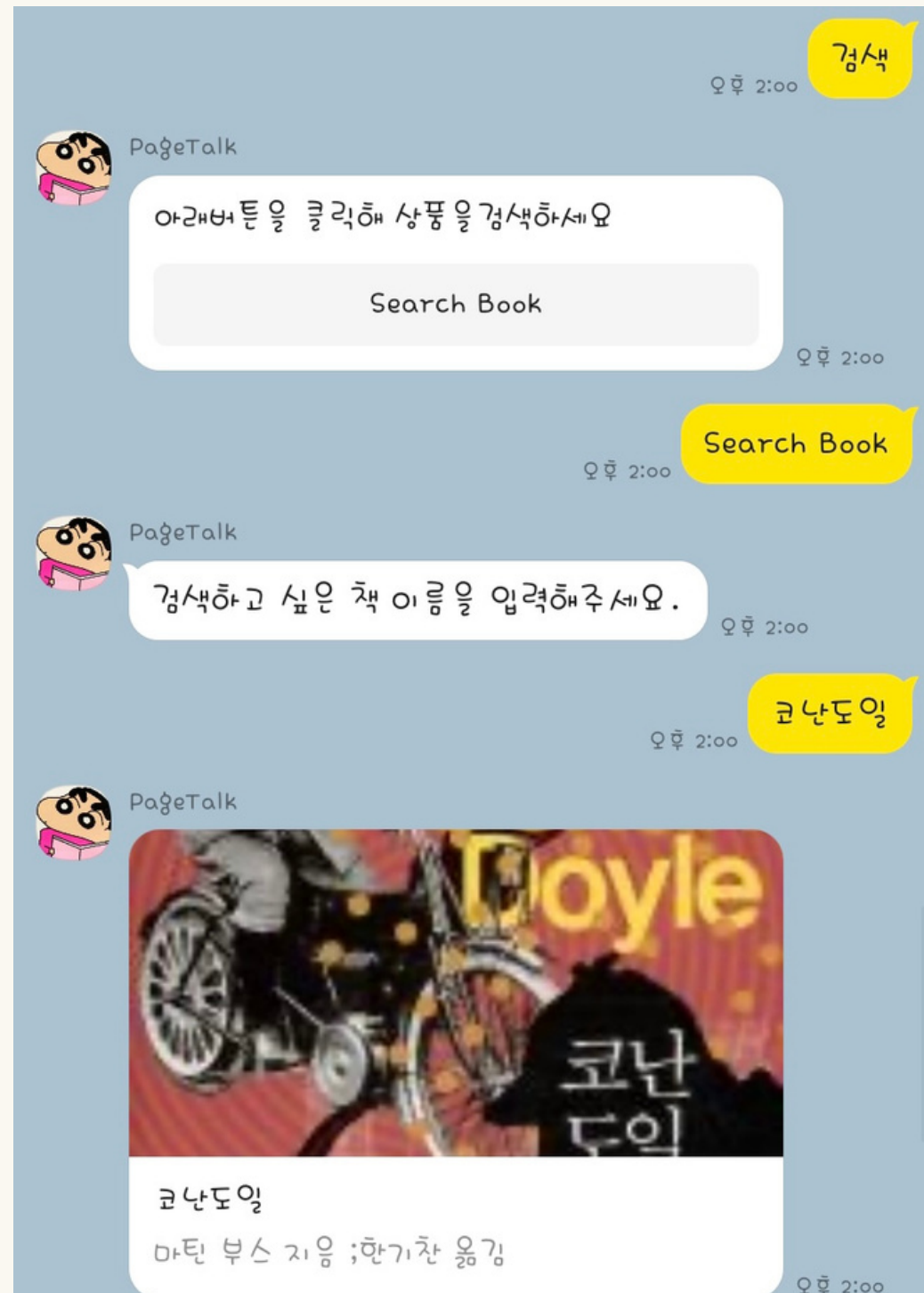
챗봇이 추천해 준 도서를 사용자가
버튼을 클릭하면 알라딘의
구매 링크로 이동하여 도서를
구매할 수 있다.



프로젝트 절차

챗봇 도서 검색

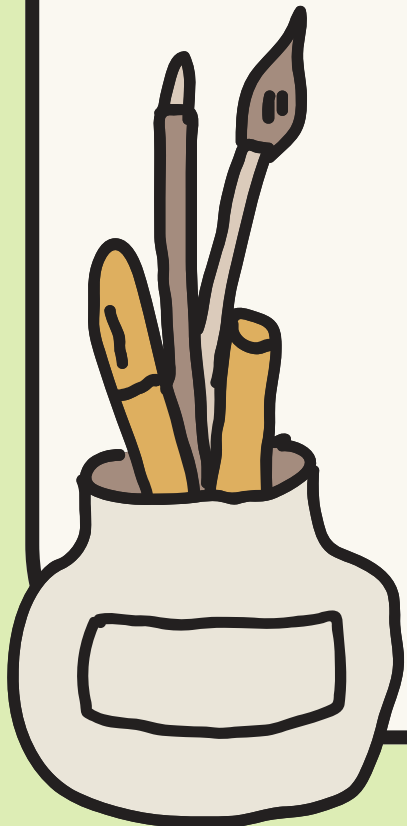
**사용자가 챗봇에게 검색이라는 키워드를
입력하게 되면 챗봇이 상품검색이라는
버튼을 보여주는데 여기서 사용자는
버튼을 누르고 찾고싶은 책 제목을
입력하면 챗봇이 검색한 책의 정보를
사용자에게 보여줍니다.**



프로젝트 절차

챗봇 소식화면

금주 및 연도별 베스트 셀러와
장르별 베스트 셀러, 사용자 메뉴얼을
소식을 통해 알려준다.



프로젝트 결과

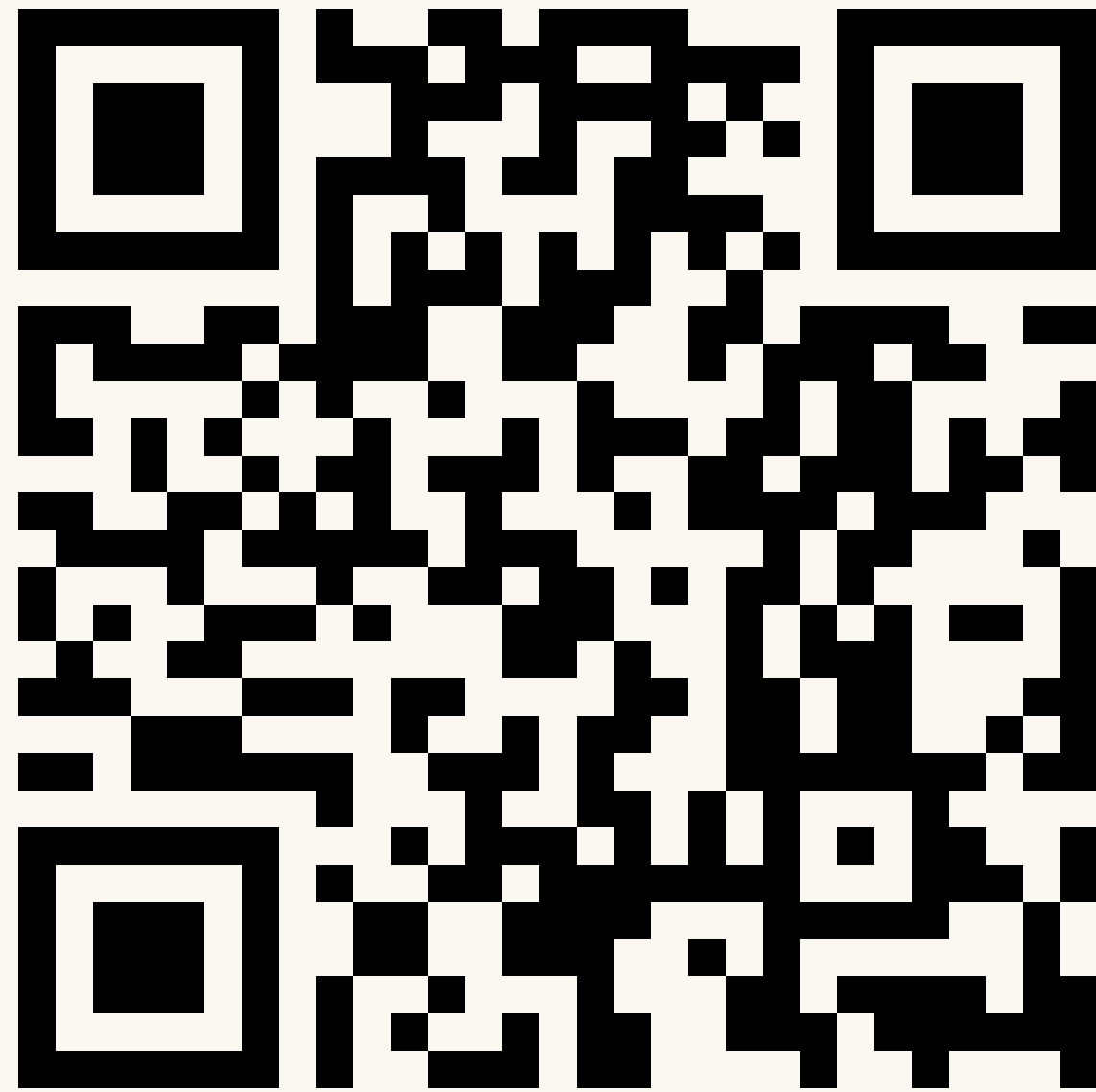


프로젝트 결과

시연영상

<https://github.com/JeongYoonBaek/PythonProject>

챗봇 QR코드



PageTalk 채널 QR코드

팀원 자체평가



팀원 자체평가

이현오

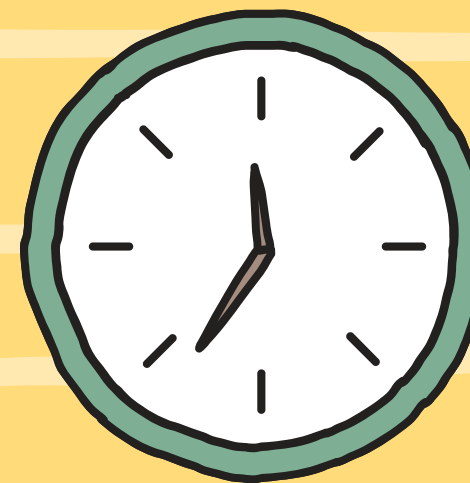
이번 프로젝트를 하면서 머신러닝을 구현 하지 못 하여서 아쉬움이 크지만 조금 더 공부해서 플라스 크 파이썬으로 챗봇의 기능을 더 다양하게 만들어 보고 싶습니다

백정운

프로젝트 기간동안 챗봇이 동작하는 다양한 방식들을 접해서 신기했고, 덕분에 챗봇에 대한 흥미도가 증가한 것 같아서 보람찬 시간이었습니다.

유지웅

프로젝트 하면서 항상 이용하던 챗봇 기능을 직접 구현한 계기가 되어 흥미로웠지만 한편으론 아직 부족한 내가 아쉬웠다.



파이썬 프로젝트

감사합니다~!

PageTalk

