

미니프로젝트



<미니프로젝트>

1. 주제

- xxx분야 고객관리 챗봇 개발

2. 기간 : 2/27일~28일까지

3. 아이디어

- "고객"은 다양한 분야의 고객으로
- 조별로 어떤 분야에 대한 챗봇을 개발할 것인지 논의

4. 필수 or 선택 적용사항

- (필수) 미니프로젝트 주제 선정
: 보험분야 고객관리 챗봇 개발
- (필수) 해당 주제 분야에서 어떤부분을 궁극해하는지에 대해,
질의어 사전 분석 실시(인터넷, 블로그 등등..)
- (필수) 가상환경 신규 생성
: pknu_0_miniproj (pknu_조_miniproj)
- (선택) Django 적용 or 미적용
: 작업폴더 이름 : pknu_0_miniproj
- (선택) MySQL Database는 적용 or 미적용
: 조별로 적용할지 말지 논의 후 진행
- (필수) 사용하는 데이터 모두 파일로 관리(*.csv...)
- (필수) 챗봇의 대화 내용을 이용하여, 감성분류 함께 진행
: 네이버 영화 리뷰 감성 분류하기 참고

5. 개인별 제출문서

<최종 제출파일>

- : 개인별 최종 제출 파일명 : miniproj_조_홍길동.zip
- : 아래 모든 내용을 zip으로 묶어서 제출

- 가상환경_라이브러리_설치_목록_전체.txt
(예시 : pip instal pandas)

- 프로젝트_개요.txt
: 선정한 프로젝트 주제에 대해
어떻게 챗봇을 통해 질문과 응답을 할 것인지
시작부터 끝까지의 단계별 모든 시나리오를 작성
: 시나리오는 조별로 자유롭게 작성

- MySQL DB를 사용할 경우
: 사용한 테이블(Table)_생성_스크립트.sql
- 사용하는 데이터 모두 파일로 관리(*.csv)

6. 조별 참고사항

- 어떻게 풀어나갈 것인지에 대한
순서(시나리오)를 조원들과 대략 논의한 후 진행
- 조원들이 진행하면서 발생하는 코드 오류는
조장님 리딩하에 조원들과 해결

프로젝트 개요

1. 어떤 방식으로 진행할까?

- 다같이 진행
2. 주제는 ??
- 에브리타임 새내기게시판에서 정보 수집 후 신입생들을 위한 챗봇
 - 부경대 홈페이지에서 교수 정보 수집 후 학생들을 위한 챗봇
 - 배달의 민족 배달 메뉴 별점 추천 및 각 음식에 관한 장단점 이야기해주는 챗봇.
 - 지그재그 등 쇼핑몰에서 옷의 품질, 특징, 스타일 등 옷에 관련된 정보 뿐만 아니라 배송상태에 대한 상태를 알려주는 챗봇
 - 여행지 추천 및 관련 정보 챗봇
3. 주제 : 부경대 홈페이지에서 교수 정보 수집 후 학생들을 위한 교수 정보 제공 챗봇
- 부경대 홈페이지에서 교수 정보 수집
 - 에브리타임 게시판에서 강의평가 수집
 - 챗봇 : 교수 정보제공, 평가 제공, 교수 시간표 제공,,,,,,

가상환경 활성화 및 라이브러리 설치

- 가상환경 및 기본 라이브러리 설치

```
# 가상환경 생성
conda create -n pknu_1_miniproj python=3.9 # 파이썬 버전에 따라 다르게

# 가상환경 활성화
conda activate pknu_1_miniproj

# 가상공간에 패키지 설치
conda install -c conda-forge jupyter notebook

# 가상환경 연결
python -m ipykernel install --user --name pknu_1_miniproj --display-name pknu_1_miniproj_kernel

# 라이브러리 설치
pip install ipython jupyter matplotlib pandas sklearn xlrd seaborn

pip install openpyxl

pip install tensorflow==2.8.2

conda install -c conda-forge pydot graphviz

# nltk 설치
pip install nltk

# konlpy 설치
pip install konlpy

# Selenium 설치
pip install selenium
```

- 자연어 처리 라이브러리

- 설치 확인
 - python
 - > import nltk
 - > nltk.__version__

```
>>> import nltk
>>> nltk.__version__
'3.8.1'
>>>
```

```
> nltk.download("treebank")
> nltk.download("punkt")
> nltk.download("stopwords")
```

```
>>> nltk.download("treebank")
[nltk_data] Downloading package treebank to
[nltk_data] C:\Users\Administrator\AppData\Roaming\nltk_data...
[nltk_data] Unzipping corpora\treebank.zip.
True
>>> nltk.download("punkt")
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\Administrator\AppData\Roaming\nltk_data...
[nltk_data] Unzipping tokenizers\punkt.zip.
True
>>> nltk.download(stopwords)
File "<stdin>", line 1
  nltk.download(stopwords)
      ^
SyntaxError: EOL while scanning string literal
>>> nltk.download("stopwords")
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\Administrator\AppData\Roaming\nltk_data...
[nltk_data] Unzipping corpora\stopwords.zip.
True
```

- 한글 라이브러리 설치

```
pip install JPyPe1
```

```
• 설치 확인
- python
> import konlpy
> konlpy.__version__
```

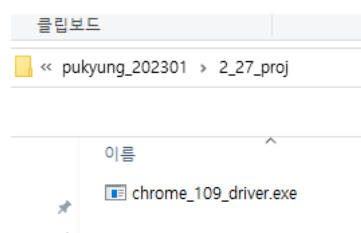
```
Type: help, "copyright",
>>> import konlpy
>>> konlpy.__version__
'0.6.0'
>>>
```

```
> python
> from konlpy.tag import Okt
> word = Okt()
```

데이터 수집 및 전처리

1. 셀레니움 준비

- 폴더 생성 후 크롬 드라이버 생성



- 주피터 실행 후 파일생성 및 코드 실행

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/741340f7-a8f0-4d86-a6b8-1aa44fb75210/Untitled2.ipynb>

셀레니움으로 크롤링 하기 - <https://workingwithpython.com/howtouse selenium-3/>

```
# 패키지
from selenium import webdriver
from selenium.webdriver.common.by import By

# 크롬드라이버 다운 - 버전 주의
# 프로그램에서 실행

driver = webdriver.Chrome('chrome_109_driver.exe')

# url 주소 전달
# 정응대 입장
url = "https://itc.pknu.ac.kr/html/00_main/"
driver.get(url)
driver.maximize_window()

# 교수진 페이지
Button_1 = driver.find_element(By.XPATH,
                                '//*[@id="gnb1m3"]')
Button_1.click()

a = driver.find_element(By.XPATH,
                        '//*[@id="container"]/div[2]/ul')
b = a.find_elements(By.TAG_NAME, "li")

majors = []
names = []
phones = []
labs = []
emails = []

all_professor = []

for i in range(1, len(b)+1):
    # 교수 정보 얻기
    Button = driver.find_element(By.XPATH,
                                '//*[@id="container"]/div[2]/ul/li[{}]/a'.format(i))
    Button.click()

    dd = driver.find_element(By.XPATH,
                            '//*[@id="container"]/div[2]/h3')

    big_table = driver.find_elements(By.XPATH,
                                    '//*[@id="container"]/div[2]/div')

    for human in big_table:
        #print(human.text)
        a = []
        info = human.text.split("\n")
        major = []
        major.append(dd.text)
        name = []
        phone = []
        lab = []
        email = []

        for temp in info :
            if "교수" in temp.split(" ") :
                name.append(temp)

        for num in range(1, len(name)+1):
            locate = "#container > div:nth-child(3) > div > div:nth-child({}) >\n\
                    div.fl.mfn.m_w_100p.pdl_30.mpd_l_0.h_script_thumb1.dpt > div > div:nth-child(4) > div:nth-child({})".format(num, num)
            t = driver.find_element(By.CSS_SELECTOR, locate)
            phone.append(t.text)

            locate = "#container > div:nth-child(3) > div > div:nth-child({}) >\n\
                    div.fl.mfn.m_w_100p.pdl_30.mpd_l_0.h_script_thumb1.dpt > div > div:nth-child(5) > div:nth-child({})".format(num, num)
            t = driver.find_element(By.CSS_SELECTOR, locate)
            email.append(t.text)

            locate = "#container > div:nth-child(3) > div > div:nth-child({}) >\n\
                    div.fl.mfn.m_w_100p.pdl_30.mpd_l_0.h_script_thumb1.dpt > div > div:nth-child(6) > div:nth-child({})".format(num, num)
            t = driver.find_element(By.CSS_SELECTOR, locate)
            lab.append(t.text)
```

al

<오류 해결>

- ```
all_professor
{'빅데이터융합전공': {'김동선 교수': ['051-629-4611', '해알환경 빅데이터 연구실', 'kimsd@pknu.ac.kr'],
'노명석 교수': ['051-629-4612', '보건의료 연구실', 'msnoh@pknu.ac.kr'],
'문형찬 교수': ['051-629-4613', '경영정책 빅데이터 연구실', 'hbmooon@pknu.ac.kr'],
'하지환 교수': ['051-629-4614', '인공지능&생물정보 연구실', 'jhha@pknu.ac.kr']},
'통계·데이터사이언스전공': {'장대을 교수': ['051-629-5537',
'자연과학2관 7215호',
'dhjlang@pknu.ac.kr'],
'박돌준 교수': ['051-629-5538', '자연과학2관 7114호', 'djpark@pknu.ac.kr'],
'이성백 교수': ['051-629-5539', '자연과학2관 7216호', 'sbvii1999@pknu.ac.kr'],
'박인호 교수': ['051-629-5542', '자연과학2관 7214호', 'ipark@pknu.ac.kr'],
'하일도 교수': ['051-629-5536', '자연과학2관 7112호', 'idha1353@pknu.ac.kr'],
'최지은 교수': ['051-629-5543', '자연과학2관 7113호', 'choije@pknu.ac.kr'],
'엄태웅 교수': ['051-629-5541', '자연과학2관 7113호', 'twuhn@pknu.ac.kr']},
'언론정보전공': {'오창호 교수': ['051-629-5481', '인문사회관 341호', 'auho21c@pknu.ac.kr'],
'한혜진 교수': ['051-629-5482', '인문사회관 342호', 'hancon@pknu.ac.kr'],
'남인용 교수': ['051-629-5483', '인문사회관 433호', 'tarzan@pknu.ac.kr'],
'이삼기 교수': ['051-629-5484', '인문사회관 432호', 'lsngk@pknu.ac.kr'],
'길무규 교수': ['051-629-5485', '인문사회관 431호', 'moog@pknu.ac.kr'],
'김정주 교수': ['051-629-5486', '인문사회관 340호', 'adprki@pknu.ac.kr'],
'임수영 교수': ['051-629-5488', '인문사회관 340호', 'imsoyung@pknu.ac.kr']}
```

<완성>

학과별 교수 정보 디렉터리로 구현

## 2. 데이터프레임

## 미니프로젝트

|     | 교수명    | 전공           | 전화번호         | 연구실           | 이메일               |
|-----|--------|--------------|--------------|---------------|-------------------|
| 0   | 김동선 교수 | 빅데이터융합전공     | 051-629-4611 | 해양환경 빅데이터 연구실 | kimds@pknu.ac.kr  |
| 1   | 노명석 교수 | 빅데이터융합전공     | 051-629-4612 | 보건의로 연구실      | msnoh@pknu.ac.kr  |
| 2   | 문형빈 교수 | 빅데이터융합전공     | 051-629-4613 | 경영정책 빅데이터 연구실 | hbmoon@pknu.ac.kr |
| 3   | 하지환 교수 | 빅데이터융합전공     | 051-629-4614 | 인공지능&생물정보 연구실 | jhha@pknu.ac.kr   |
| 4   | 장대홍 교수 | 통계·데이터사이언스전공 | 051-629-5537 | 자연과학2관 7215호  | dhjang@pknu.ac.kr |
| ... | ...    | ...          | ...          | ...           | ...               |
| 91  | 장원두 교수 | 컴퓨터공학전공      | 051-629-6246 | 누리관 2102호     | chang@pknu.ac.kr  |
| 92  | 신인철 교수 | 컴퓨터공학전공      | 051-629-6242 | 누리관 2203호     | icshin@pknu.ac.kr |
| 93  | 김태국 교수 | 컴퓨터공학전공      | 051-629-6241 | 누리관 2209호     | king@pknu.ac.kr   |
| 94  | 김훈희 교수 | 컴퓨터공학전공      | 051-629-6244 | 뇌 기반 인공지능 연구실 | h2kim@pknu.ac.kr  |
| 95  | 유승호 교수 | 컴퓨터공학전공      | 051-629-6243 | 협력 지능형 장치 연구실 | shyoo@pknu.ac.kr  |

96 rows × 5 columns

## 챗봇 만들기

### 한국어 BERT의 마스크드 언어 모델

#### 규칙기반(Rule Based) 챗봇

```
import pandas as pd

df = pd.DataFrame(all_professor)
df.columns = ["교수명", "전공", "전화번호", "연구실", "이메일"]

챗봇 데이터 만들기
response = []

for i in range(len(df)):
 info = df["전공"][i] + " " + df["전화번호"][i] + " " + df["연구실"][i] + " " + df["이메일"][i]
 response.append(info)

chatbot_data = pd.DataFrame({'rule':df["교수명"], 'response':response})
chatbot_data
```

```
chatbot_data["rule"] = chatbot_data["rule"].str.replace(" 교수", "")
chatbot_data
```

|     | rule                         | response                      |
|-----|------------------------------|-------------------------------|
| 0   | 김동선 빅데이터융합전공051-629-4611     | 해양환경 빅데이터 연구실kimds@pknu.ac.kr |
| 1   | 노명석 빅데이터융합전공051-629-4612     | 보건의로 연구실msnoh@pknu.ac.kr      |
| 2   | 문형빈 빅데이터융합전공051-629-4613     | 경영정책 빅데이터 연구실hbmoon@pknu.a... |
| 3   | 하지환 빅데이터융합전공051-629-4614     | 인공지능&생물정보 연구실jhha@pknu.ac.kr  |
| 4   | 장대홍 통계·데이터사이언스전공051-629-5537 | 자연과학2관 7215호dhjang@pkn...     |
| ... | ...                          | ...                           |
| 91  | 장원두 컴퓨터공학전공051-629-6246      | 누리관 2102호chang@pknu.ac.kr     |
| 92  | 신인철 컴퓨터공학전공051-629-6242      | 누리관 2203호icshin@pknu.ac.kr    |
| 93  | 김태국 컴퓨터공학전공051-629-6241      | 누리관 2209호king@pknu.ac.kr      |
| 94  | 김훈희 컴퓨터공학전공051-629-6244      | 뇌 기반 인공지능 연구실h2kim@pknu.ac.kr |
| 95  | 유승호 컴퓨터공학전공051-629-6243      | 협력 지능형 장치 연구실shyoo@pknu.ac.kr |

96 rows × 2 columns

```
[질문]이 데이터프레임의 request에 포함되어 있는지 확인하는 기능 구현
- 있다면 : 데이터프레임에서 해당 index의 response의 값으로 응답 처리
```

```
- 없다면 : "무슨 말인지 잘 모르겠네요~" 응답 처리
** 챗봇은 질문자와 반복해서 주고 받아야 하기 때문에..
- 질문을 처리하는 기능, 즉 함수로 만들어져 있어야 합니다.

def chat(request) :
 # request = "책 추천해줘"
 for k, v in chat_dic.items() :
 # print(k,v)
 ### rule에 포함되어 있는지 확인하기 위한 변수 지정
 chat_flag = False

 ### 질문 내용이 rule에 포함되어 있는지 확인하기
 for word in v :
 if word in request :
 chat_flag = True
 else :
 chat_flag = False
 break

 ### 질문에 rule이 있다면..
 if chat_flag : # chat_flag에 대해 True 일 때.
 return chatbot_data["response"][k]

 ### 질문에 대한 rule이 없다면
 return "무슨 말인지 잘 모르겠네요~" # return은 자동적으로 break됨
```

chat("김성운")

'정보통신공학전공051-629-6235프로토콜공학연구소kimsu@pknu.ac.kr'

chat("김성운")

'정보통신공학전공051-629-6235프로토콜공학연구소kimsu@pknu.ac.kr'

## 진행중인 파일 - 강의평 추가 전

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/89c9063d-bf8d-49ed-b0cb-e31ea5898ac1/%EA%B5%90%EC%88%98%EC%A0%95%EB%B3%B4%EC%B1%97%EB%B4%87.ipynb>

## 사용하는 데이터 파일

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/ac75b828-688c-48bc-a230-09d24bd9de83/%EA%B5%90%EC%88%98%EC%A0%95%EB%B3%B4.csv>

## 교수 정보 챗봇

```
[질문]이 데이터프레임의 request에 포함되어 있는지 확인하는 기능을 구현
- 있으면 : 데이터프레임에서 해당 index의 response값으로 응답처리
- 없으면 : "무슨 말인지 잘 모르겠네요~" 응답처리

** 챗봇은 질문자와 반복해서 주고 받아야하기때문에 질문 처리기능이 함수로 만들어져있어야함

맨처음 챗봇 -> 정보를 넣으면 알려주는 것이므로 request 로 정보 종류에 대해 받아야함
def chat_prof(req, professor_name):
 for i in range(len(df)):
 if df["교수명"][i] == professor_name:
 val = i
 break

 if req == "전공":
 return "{} 교수님의 전공은 {}입니다.".format(professor_name, df["전공"][val])
 elif req == "연구실":
 return "{} 교수님의 연구실은 {}입니다.".format(professor_name, df["연구실"][val])
 elif req == "이메일":
 return "{} 교수님의 이메일은 {}입니다.".format(professor_name, df["이메일"][val])
 elif req == "전화번호":
```

```

 return "{} 교수님의 전화번호는 {}입니다.".format(professor_name, df["전화번호"][val])
 elif req in chat_dic.values():
 return req
 else :
 return "전공/연구실/이메일/전화번호 만 입력해주세요"

```

```

챗봇 인터페이스 만들기
- 질문자와 응답챗봇간의 인터페이스
- 질문자의 질문을 함수에 전달
- 함수가 리턴한 응답을 질문자에게 보여주기
- 인터페이스 종료까지

professor_name = ""
#무한반복시키기
while True :
 ### 외부장치 오류 있을 수 있으므로 try, except로
 try :
 if not professor_name: # 전역 변수가 비어있으면 교수님 이름 입력받기
 print("ChatBot : 궁금한 교수님 성함을 입력해주세요")
 req = input("나 : ")
 print("-----")
 if req in chat_dic.values(): # 입력받은 교수님 이름이 chat_dic.values() 경우에만 전역 변수에 저장합니다.
 professor_name = req
 elif req == "exit":
 print("ChatBot : 감사합니다")
 print("-----")
 break
 else :
 req = input("ChatBot : 다시 입력해주세요: ")
 print("-----")
 if req == "exit":
 print("ChatBot : 감사합니다")
 break

 else: # 전역 변수가 비어있지 않은 경우에는 해당 교수님 정보를 출력합니다.
 print("ChatBot : {} 교수님에 대해 더 알고싶은 정보를 입력해 주세요. (전공/연구실/이메일/전화번호)\n(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가
 req = input("나 : ")
 print("-----")
 if req == "없어":
 print("감사합니다")
 print("-----")
 break

 ### 질문을 함수에 보내서 응답메세지 받아오기
 elif req in chat_dic.values():
 professor_name = req
 else :
 print("ChatBot : ", chat_prof(req, professor_name))
 print("-----")

 except :
 print("알 수 없는 오류가 발생했습니다. 종료합니다.")
 print("-----")
 break

```



ChatBot : 궁금한 교수님 성함을 입력해주세요  
나 : 정연호

ChatBot : 정연호 교수님에 대해 더 알고싶은 정보를 입력해 주세요.(전공/연구실/이메일/전화번호)  
(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가 없으면 '없어'를 입력해주세요):  
나 : 전공

ChatBot : 정연호 교수님의 전공은 정보통신공학전공입니다.

ChatBot : 정연호 교수님에 대해 더 알고싶은 정보를 입력해 주세요.(전공/연구실/이메일/전화번호)  
(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가 없으면 '없어'를 입력해주세요):  
나 : 이메일

ChatBot : 정연호 교수님의 이메일은 yhchung@pknu.ac.kr입니다.

ChatBot : 정연호 교수님에 대해 더 알고싶은 정보를 입력해 주세요.(전공/연구실/이메일/전화번호)  
(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가 없으면 '없어'를 입력해주세요):  
나 : 전화번호

ChatBot : 정연호 교수님의 전화번호는 051-629-6236입니다.

ChatBot : 정연호 교수님에 대해 더 알고싶은 정보를 입력해 주세요.(전공/연구실/이메일/전화번호)  
(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가 없으면 '없어'를 입력해주세요):  
나 : 연구실

ChatBot : 정연호 교수님의 연구실은 이동전송시스템연구실입니다.

ChatBot : 정연호 교수님에 대해 더 알고싶은 정보를 입력해 주세요.(전공/연구실/이메일/전화번호)  
(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가 없으면 '없어'를 입력해주세요):  
나 : 김성운

ChatBot : 김성운 교수님에 대해 더 알고싶은 정보를 입력해 주세요.(전공/연구실/이메일/전화번호)  
(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가 없으면 '없어'를 입력해주세요):  
나 : 연구실

ChatBot : 김성운 교수님의 연구실은 프로토콜공학연구실입니다.

ChatBot : 김성운 교수님에 대해 더 알고싶은 정보를 입력해 주세요.(전공/연구실/이메일/전화번호)  
(다른 교수님을 원하시면 성함을, 더 알고싶은 정보가 없으면 '없어'를 입력해주세요):  
나 : 없어

감사합니다

## 진행중인 파일 - 강의평 추가 후 감성분류 함수 추가 전

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/503c2f32-54c2-4962-9e08-f62beeb60561/%EA%B5%90%EC%88%98%EC%A0%95%EB%B3%B4%EC%B1%97%EB%B4%87.ipynb>

## 감성분류 함수 추가 후 최종

[https://s3-us-west-2.amazonaws.com/secure.notion-static.com/98b11b82-3140-421e-9ecc-062d9e7bb4d6/%EA%B5%90%EC%88%98%EC%A0%95%EB%B3%B4\\_%EC%9B%B9%ED%81%AC%EB%A1%A4%EB%A7%81.ipynb](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/98b11b82-3140-421e-9ecc-062d9e7bb4d6/%EA%B5%90%EC%88%98%EC%A0%95%EB%B3%B4_%EC%9B%B9%ED%81%AC%EB%A1%A4%EB%A7%81.ipynb)

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/24fe74d8-bc1c-4a3f-a34c-7017a7d15e51/%EA%B5%90%EC%88%98%EC%A0%95%EB%B3%B4%EC%B1%97%EB%B4%87.ipynb>

## 감성 분류

### 네이버 영화리뷰 감성 분류하기

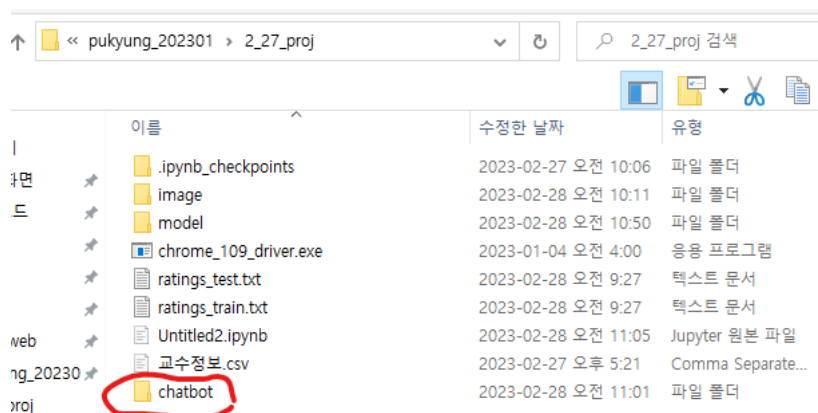
- 설치목록
  - `pip install chardet`
  - `pip install charset-normalizer`

### 모델

[https://s3-us-west-2.amazonaws.com/secure.notion-static.com/d402bd0e-a0d1-4736-98e2-d3eec84502de/model\\_best.h5](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/d402bd0e-a0d1-4736-98e2-d3eec84502de/model_best.h5)

## Django 적용

- 라이브러리 설치
  - # django 설치  
`conda install -c conda-forge django==4.0.1`
  - 위치 이동 후 서버 설치 / 우리 주피터 작업 폴더 위치로 이동 후 chatbot 폴더 생성 후 폴더 안에서 코드 입력

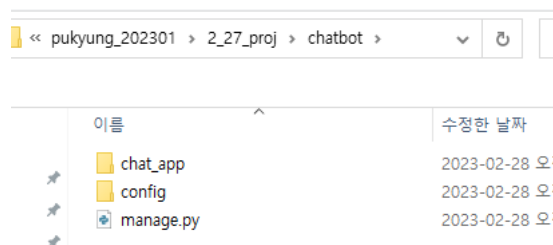


|                   | 이름                    | 수정한 날짜              | 유형                |
|-------------------|-----------------------|---------------------|-------------------|
| 폴더                | .ipynb_checkpoints    | 2023-02-27 오전 10:06 | 파일 폴더             |
|                   | image                 | 2023-02-28 오전 10:11 | 파일 폴더             |
|                   | model                 | 2023-02-28 오전 10:50 | 파일 폴더             |
| 실행 파일             | chrome_109_driver.exe | 2023-01-04 오전 4:00  | 응용 프로그램           |
| 텍스트 문서            | ratings_test.txt      | 2023-02-28 오전 9:27  | 텍스트 문서            |
| 텍스트 문서            | ratings_train.txt     | 2023-02-28 오전 9:27  | 텍스트 문서            |
| Jupyter 원본 파일     | Untitled2.ipynb       | 2023-02-28 오전 11:05 | Jupyter 원본 파일     |
| Comma Separate... | 교수정보.csv              | 2023-02-27 오후 5:21  | Comma Separate... |
| 파일 폴더             | chatbot               | 2023-02-28 오전 11:01 | 파일 폴더             |

```
django-admin startproject config .
```

- 앱 생성

```
python manage.py startapp chat_app
```



|             | 이름        | 수정한 날짜              |
|-------------|-----------|---------------------|
| 파일 폴더       | chat_app  | 2023-02-28 오전 11:01 |
| 파일 폴더       | config    | 2023-02-28 오전 11:01 |
| Python 스크립트 | manage.py | 2023-02-28 오전 11:01 |

- config/settings 설정하기
  - 호스트 설정

```
config > settings.py > ...
27
28 ALLOWED_HOSTS = ['127.0.0.1']
29
```

- 앱등록

```
config > settings.py > ...
33 INSTALLED_APPS = [
34 'chat_app'
```

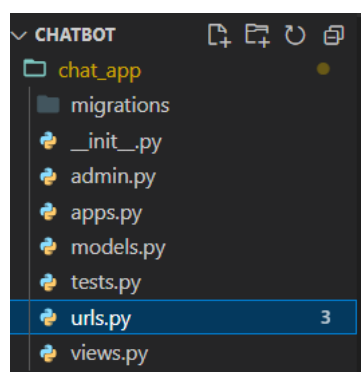
- html 관리 폴더 지정

```
config > settings.py > ...
55 TEMPLATES = [
56 {
57 'BACKEND': 'django.template.backends.django.DjangoTemplates',
58 'DIRS': [BASE_DIR / 'templates']
59 },
60 {
61 'APP_DIRS': True,
62 'OPTIONS': {
63 'context_processors': [
64 'django.template.context_processors.debug',
65 'django.template.context_processors.request',
66 'django.contrib.auth.context_processors.auth',
67 'django.contrib.messages.context_processors.messages',
68],
69 },
70]
71]
```

- 지역, 시간 설정

```
config > settings.py > ...
109 LANGUAGE_CODE = 'ko-kr'
110
111 TIME_ZONE = 'Asia/Seoul'
112
```

- config/urls.py 복사 후 chat\_app에 붙여넣기



- config/urls.py에 라우팅 설정

```

config > urls.py > ...
14 2. Add a URL to urlpatterns: path('blog/', i
15 """
16 from django.contrib import admin
17 from django.urls import path, include
18
19 urlpatterns = [
20 path('chat/', include('chat_app.urls')),
21 path('admin/', admin.site.urls),
22]

```

```

<!DOCTYPE html>
{% load static %}
<html>
<head>
 <script tpye="text/javascript">
 function input() {
 fm = document.getElementById("fm");
 fm.action = '/chat/loading/'
 fm.submit();
 }
 $("#fm").keypress(function(e) {
 if (e.keyCode == 13) {
 input();
 }
 });
 </script>
 <title>:: 교수 정보 챗봇 ::</title>
</head>
<body>
 <table border=1 width=1000 align="center">
 <tr>
 <th>교수 정보 챗봇</th>
 </tr>
 {% for sentences in lists %}
 <tr>
 <th>{{sentences}}</th>
 </tr>
 {% endfor %}
 <tr>
 <form name="fm" id="fm" method="post" action="#">
 {% csrf_token %}
 <td align="right"><input type="text" id="sentence" name="sentence" value="" size=60 ></td>
 </form>
 </tr>
 </table>
 <table border=0 align="center" width=1000>
 <td align="right">
 <input type="button" onclick="input()" value="입력">
 </td>
 </table>
</body>
</html>

```

[https://s3-us-west-2.amazonaws.com/secure.notion-static.com/de3ce050-3f09-4899-b3f4-6fc0ed2b3b44/%ED%94%84%EB%A1%9C%EC%A0%9D%ED%8A%B8\\_%EA%B0%9C%EC%9A%94.txt](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/de3ce050-3f09-4899-b3f4-6fc0ed2b3b44/%ED%94%84%EB%A1%9C%EC%A0%9D%ED%8A%B8_%EA%B0%9C%EC%9A%94.txt)