



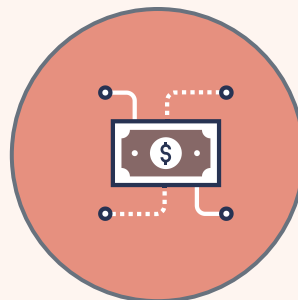
Diary

오늘 하루를 적어주세요!

11 : 57 PM

INTRO

- 팀원 소개
 - 주제 선정 배경
 - 감정 선정
-



데이터 및 모델

- 명언, 음악 데이터 수집 방식
 - 모델 훈련용 데이터 수집 및 전처리
 - 모델 세부내용
-

웹 구현 기능

- 서비스 개요
 - 주요 코드
-



한계 및 방향성

- 팀내 협업사항
 - 한계
 - 방향성
-

We Can Do Team Profile

Data



오 도운

AI



윤 이현

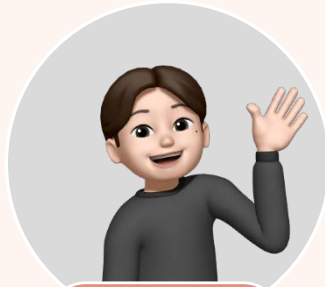
Back-end



오 형동



고 병욱

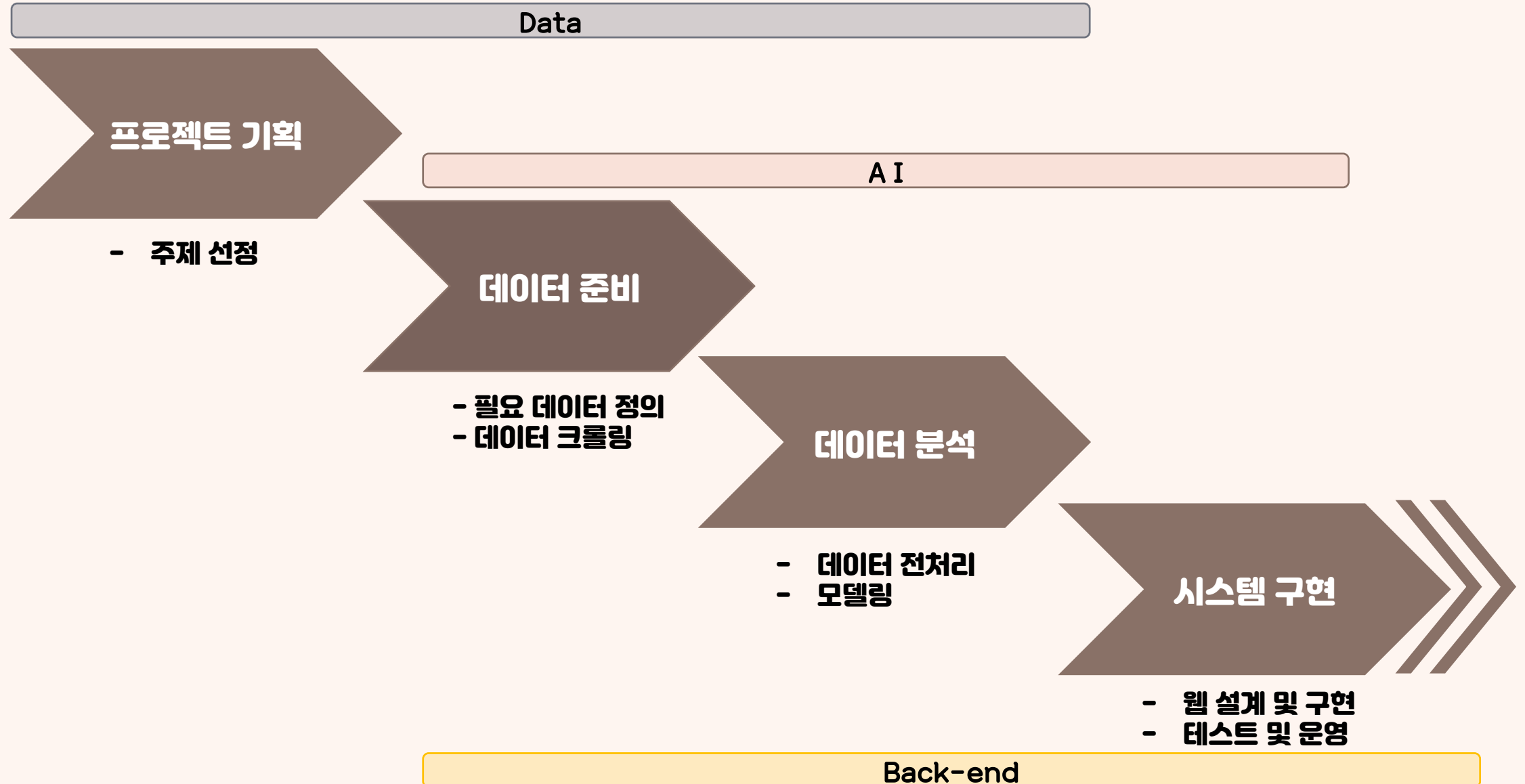


정 범준



강 희연

Team Project Schedule



우울증 1000만명 시대...치료율 OECD 꼴찌 원인은?



최선 기자

발행날짜: 2021-09-14 15:36:16

신경과학회, SSRI 항우울제 처방 제한 불합리 주장
"우울증 치료 없이 자살률 저하 불가능...처방 완화 해답"

[메디칼타임즈=최선 기자] 지난 6년간 우울증 치료율이 OECD 최하위에 머무르면서 신경과학회가 SSRI 항우울제의 처방 제한 완화를 요구하고 나섰다.

국내의 SSRI 항우울제 사용량이 전세계 최저라는 점, 우울증이 자살 등 사회적 비용 발생에 미치는 영향을 때 안전한 약물인 SSRI를 규제할 명분이 부족하다는 것이다.



KNA 대한신경과학회

14일 신경과학회는 성명을 통해 "우울증으로 국민의 정 나라가 무너지고 있다"며 "2013년 OECD 나라들 중 가 한국의 우울증 치료율(인구 1000명당 항우울제 사용량 전혀 변하지 않았다"고 지적했다.

전체 항우울제 사용량은 라트비아 다음으로 두 번째로 낮지만 안전한 SSRI 항우울제의 사용량은 세계 최저다. 한국 은 6년 동안 자살률도 계속 OECD 1위를 기록했다. 우울증 치료율이 증가하지 않으면 자살률이 감소하지 않는다는

일기장으로 우울증 관리 한다고? 9개월만에 3.5만명이 쓰는 앱'

머니투데이 | 김유경 기자

VIEW 9,962 | 2021.12.12 13:00

| [스타트업스토리] 윤정현 블루시그널 대표



더 알아보기

• 일기쓰기가 우울증 및 자아 존중감에 긍정적인 영향을 미침

• 일기쓰기를 이용한 긍정적 자기암시가 학생의 자아 존중감 향상과 우울증 감소에 미치는 효과, 정 도교(2008)

- **기존의 일기 앱 비교 분석**

- 단순히 일상을 기록
 - 일기와 그 날의 소비 습관을 접목하는 일기 + 가계부 형식의 앱도 존재
 - 일기에 현재 자신의 감정을 추가하여 통계 결과 제시
 - 앱에 따라 사진 저장 , A P I 를 활용하여 영화 저장 등 다양한 기능을 추가
- **일기와 감정을 접목** , A I 가 감정 상태를 분류하여 그 날의 감정상태에

적절한 명언과 음악을 추천해주는 서비스 제공

Paul Ekman	문장에 내재한 감정 분석을 위한 감정 분류 방법 (국내 특허)	딥러닝 모델(BERT)과 감정어휘 사전을 결합한 음원가사 감정 분석
심리학자로 감정연구의 대가	감정사전을 제작하여 감정을 긍정, 부정에서 나아가 SVM을 활용 디테일하게 분류	가사분석을 딥러닝 모델을 활용하여 감정 분석
기쁨, 사랑, 놀람, 괴로움, 두려움, 경멸, 분노 7가지로 분류	기쁨, 슬픔, 분노, 공포, 중립 5가지로 분류	사랑, 즐거움, 열정, 행복, 슬픔, 분노, 외로움, 그리움, 두려움 9가지로 분류

→ 이를 토대로 프로젝트에서는 감정 라벨링을 기쁨, 사랑, 슬픔, 분노, 걱정, 중립으로 분류

<메인 페이지 - 로그인 전>

		로그인
설명글	로그	→ 시작하기
Footer		

<상세 페이지>

로그	로그인 상태 표시
일기 내용	추천 음악 추천 내용 →
Footer	

<일기 작성 페이지>

로그	로그인 상태 표시
제목 작성 내용 작성 <div> <div>사진 삽입</div> <div>그림 혹은 스티커 삽입</div> </div>	
Footer	

- 웹 페이지 크롤링을 통해 데이터 수집
- 명언의 경우 영어와 한글로 구분하여 사이트에 게재된 명언 내용을 크롤링
 - 1차 전처리로 상황에 맞지 않는 명언, 속담 등은 제거
 - 2차 전처리로 명언의 경우 긍정적 감정과 부정적 감정 상황에서

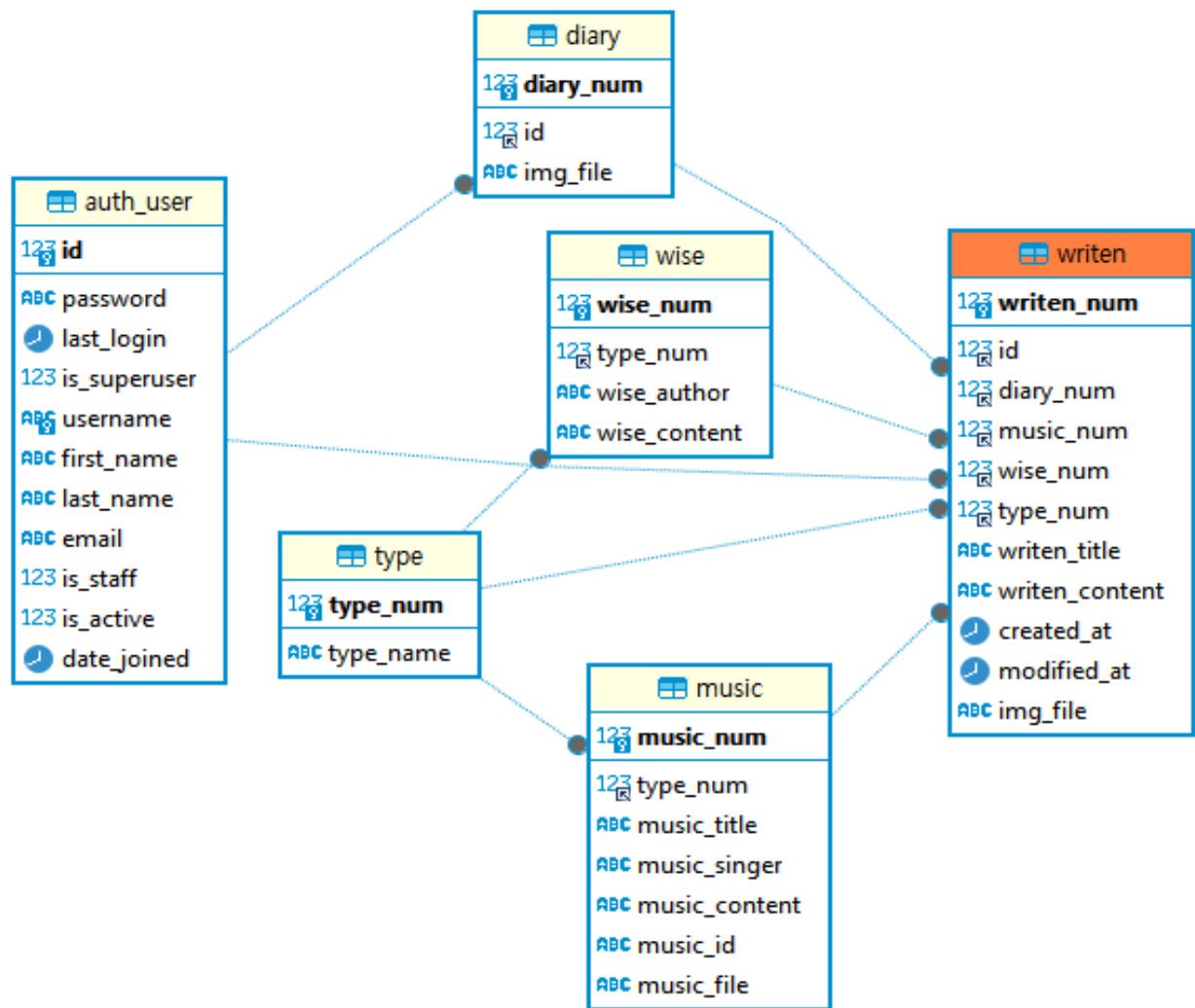
결과가 나올 수 있게끔 라벨링



- 음악의 경우 멜론에서 제공하는 DJ가 작성한 플레이 리스트를 제목, 가수, 노래가사를 크롤링
 - 국내 곡과 Pop-song을 분류하여 각 650곡씩 크롤링
- 노래의 경우 노래가사를 각각 한글, 영어 AI 모델을 통해 1차 라벨링을 진행
- 노래를 크롤링할 때 분류된 라벨링과 AI가 분류한 라벨링을 비교하여 데이터 전처리 진행
 - 라벨링이 일치하지 않는 곡 삭제
 - 해당 라벨링을 비교하여 전처리 할 때 라벨링의 일치율이 50%정도 밖에 되지 않았음
 - 최초 크롤링 당시 분류된 분류도 DJ의 주관적인 판단에 의한 분류



<데이터 베이스 ERD>



- 서버 : Docker를 활용하여 가상서버 MYSQL 구축
- Auth_user : 가입 사용자 테이블
- Type : 감정라벨링(긍정 2개, 부정 3개, 중립 1개)
- Wise : 명언 데이터
- Music : 음악 데이터
- Written : 작성된 일기 데이터
- Written 테이블과 Type_Num이 일치하는 범주에서 명언과 음악 무작위 추천

- **기본 베이스 데이터 : GPT를 활용하여 감정별 예시문장 1000개씩 생성하여 훈련**
- **베이스 데이터로 훈련시킨 모델을 바탕으로 라벨링**
 - **트위터 / Reddit / 영화감상평 데이터(IMDB) 등 감정 라벨링별로 균일하게 영어 문장 데이터 수집**
 - **최종적으로 활용된 데이터 : 약 56,000여 문장**
- **라벨링한 문장들을 대상으로 전처리를 따로 진행**
 - **기본 베이스 데이터로 1차 라벨링 진행**
 - **추가로 수집된 데이터를 통해 모델 추가 훈련**

- 사전학습 모델인 BERT를 활용(bert-base-uncased)
- 영어의 경우 불용어 사전의 유무가 모델의 성능에 차이를 보이지 않아 불용어 처리 없이

모델의 학습 진행

- 영어 문장을 토큰화와 패딩을 통해 문장의 길이를 일치
- 최적화 optimizer로 Adam W 활용
 - BERT와 같은 대규모 신경망에 적합하게 설계되어 있음
 - 모델의 과적합을 예방하는데 매우 효과적

- **AI HUB 에서 빅데이터를 수집하여 전처리와 라벨링 진행**
- **사용된 데이터 (총 90,826 개)**
 - 감성 대화 주제별 텍스트 일상 대화 데이터 – 51,631 개
 - 한국어 감성 정보가 포함된 단발성 대화 데이터 – 38,595 개
 - OPEN.AI의 CHAT GPT를 사용하여 각 감정에 대한 문장을 100개 씩 출력 – 600 개
- **데이터 전처리**
 - 기존 라벨링 된 데이터를 한글 모델을 활용하여 재라벨링
 - 교차하지 않은 데이터는 삭제 처리

- 한글 특성상 은, 는, 이, 가 등 조사와 보조사가 문장에 있기 때문에 불용어 사전을 만들어 처리.

```
stopwords = set([
    '은', '는', '이', '가', '을', '를', '에', '의', '로', '으로', '와', '과', '도', '에서', '에게', '한', '하다',
    '그', '그녀', '저', '이것', '저것', '그것', '무엇', '어느', '우리', '너희', '당신', '여기', '저기', '거기',
    '하다', '되다', '있다', '없다', '같다',
    '사실', '경우', '문제', '사람', '이유', '시간', '일', '년', '월', '일',
    '음', '아', '어', '그러나', '하지만', '그래서', '그리고', '또한', '요', '까', '곧', '질', '야', '너무'
])
```

- 한국어를 분석할 수 있게끔 bert-base-multilingual-case 활용

- Kobert와 실제문장 평가시 더 높은 성능을 보였음

- 다중 분류에 사용하는 SOFTMAX 함수 사용

- 불용어 사전의 단어를 제외한 후 토큰화를 진행,

패딩을 통해 문장의 길이 일치

<모델 구조>

Model: "custom_bert_model_8"

Layer (type)	Output Shape	Param #
tf_bert_model_8 (TFBertModel)	multiple	177853440
dropout_341 (Dropout)	multiple	0
dense_8 (Dense)	multiple	4614

=====

Total params: 177858054 (678.47 MB)
 Trainable params: 177858054 (678.47 MB)
 Non-trainable params: 0 (0.00 Byte)

- NLP(자연어 처리)에서 BERT와 GPT가 가장 대중적으로 쓰이는 모델

	BERT	GPT
문맥의 이해	양방향 이해	단방향 이해(왼쪽 → 오른쪽)
전이학습	범용 모델 다양한 NLP 작업에 적용 가능	활용은 가능 일반적으로 텍스트 생성에 더 강점
작업의 특성	감정 분석, 텍스트 분류, 질문 답변	텍스트 생성, 스토리텔링, 챗봇 개발

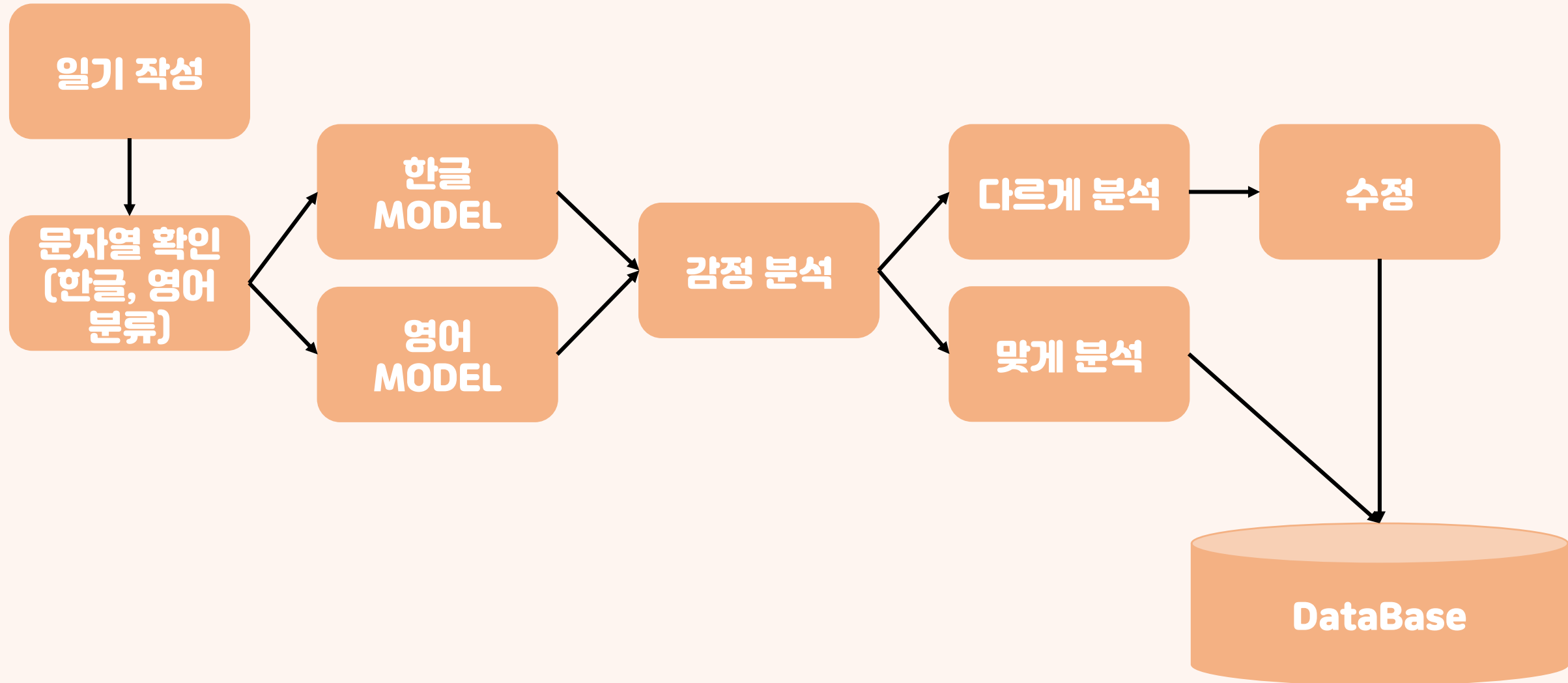
※ 전이 학습 : 방대한 양의 데이터로 이미 학습된 인공지능을 유사하지만 다른 분야에 적용하는 방법

- 실제로 모델 평가에 있어 프로젝트 기간 중 GPT 모델 활용
 - 한글과 영어 모델 모두 다양하게 모델링을 시도
 - 영어의 경우 본 프로젝트에서 평가지표로 삼은 정확도가 BERT 모델에 비해 낮은 수치를 보임

- 본 프로젝트에서는 모델의 정확도를 평가지표로 활용
- 기존의 연구들에서는 감성 분류(이진 분류)가 주를 이루었음
 - 기존 연구들과 정확도를 비교하기엔 분류의 수가 상이하다고 판단
- 라벨링의 종류는 다르지만 다중 분류의 수가 똑같은 연구를 찾아 해당 연구의 정확도와 비교 분석

	기존 연구	영어 모델	한글 모델
데이터셋	37,000 여개	56,000 여개	91,000 여개
특이사항	라벨링한 데이터의 재분류 기존 데이터의 전처리	감정별 데이터의 비율	다양한 데이터의 활용
정확도	0.91	0.82	0.78

※ [참고 연구 링크](#)



3. 웹 기능 구현

모델 웹에 구현(코드)

We Can Do

<Apps.py>

```
# 한국어 모델 구성에 필요한 BERT 토큰라이저와 모델 로드
tokenizer_ko = BertTokenizer.from_pretrained('bert-base-multilingual-cased')
bert_model = TFBertModel.from_pretrained('bert-base-multilingual-cased')
MAX_LEN = 50

# 영어 모델 PyTorch 모델 로드
model_path = 'C:/Users/ITSC/Desktop/Project/WECANDO/final/wecando/static/wecando/model/영어_최종_Bert.pth'
model_eng = BertForSequenceClassification.from_pretrained('bert-base-uncased', num_labels=6)
# 영어 모델을 CPU에서 불러오기
model_eng.load_state_dict(torch.load(model_path, map_location=torch.device('cpu')))
model_eng.eval()

# 영어 모델 데이터 토큰화
tokenizer_eng = BertTokenizer.from_pretrained('bert-base-uncased')

# 한국어 모델 사용자 정의 모델 정의
class CustomBERTModel(tf.keras.Model):
    def __init__(self, num_classes):
        super().__init__()
        self.bert = bert_model
        # 추가적인 레이어는 필요에 따라 정의
        self.dropout = tf.keras.layers.Dropout(0.1) # 예시로 dropout 레이어 추가
        self.classifier = tf.keras.layers.Dense(num_classes, activation='softmax')

    def call(self, inputs, attention_mask=None, token_type_ids=None, training=False):
        # BERT 모델의 출력을 가져옵니다.
        outputs = self.bert(inputs, attention_mask=attention_mask, token_type_ids=token_type_ids)
        pooled_output = outputs.pooler_output
        pooled_output = self.dropout(pooled_output, training=training)
        return self.classifier(pooled_output)
```

```
# 한국어 모델 예측 함수
def lyrics_evaluation_predict(sentence):
    data_x = WecandoConfig.sentence_convert_data(sentence) # 문장을 모델 입력 형식으로 변환
    predict = WecandoConfig.model_ko.predict(data_x)
    predict_value = np.ravel(predict[0])
    predict_emotion: {0: '기쁨', 1: '사랑', 2: '슬픔', 3: '분노', 4: '걱정', 5: '중립'}
    # 예측된 클래스의 인덱스를 찾기.
    predicted_class = np.argmax(predict_value)

    return predicted_class

# 영어 모델 예측 함수
def predict_emotion(text):

    text1 = text.replace('[^A-Za-z0-9가-힣]', '')
    text2 = text1.lower()

    inputs = tokenizer_eng(text2, return_tensors="pt", add_special_tokens=True, padding=True,
                           truncation=True, max_length=512)
    emotion_labels = [0,1,2,3,4,5]
    with torch.no_grad():
        outputs = model_eng(**inputs)
        prediction = torch.argmax(outputs.logits, dim=1).item()
    return emotion_labels[prediction]
```

- 작성된 일기를 감정 평가를 위해 모델 탑재
- Views.py에 작성할 경우 모델이 매번 로딩되어, 감정 평가시 시간이 매우 오래 걸림

<Views.py>

일기내용중에 한글과 영어 중 더 많이 사용된 언어의 모델을 사용하기 위해 각 문자의 사용량을 확인

```
content = self.request.POST.get("writen_content")
```

```
ko = 0
```

```
eng = 0
```

```
for i in content:
```

```
    if (ord(i) >= 65 and ord(i) <= 90) or (ord(i) >= 97 and ord(i) <= 122):
```

```
        eng = eng + 1
```

```
    else:
```

```
        ko = ko + 1
```

```
print(ko, eng)
```

한국어가 더 많으면 한국어 모델을 사용

```
if ko >= eng:
```

```
    a = WecandoConfig.lyrics_evaluation_predict(self.request.POST.get("writen_content"))
```

```
    print("한국어 모델 사용 " + str(a))
```

영어가 더 많으면 영어 모델을 사용

```
else:
```

```
    a = WecandoConfig.predict_emotion(self.request.POST.get("writen_content"))
```

```
    print("영어 모델 사용 " + str(a))
```

Writen의 type_num에 분석된 감정을 저장

```
type_key = Type.objects.get(type_num=(a))
```

```
form.instance.type_num = type_key
```

- Apps.py에서 구성한 한글, 영어 모델 중 문자열 내 철자 수를 바탕으로 적용할 모델 선정

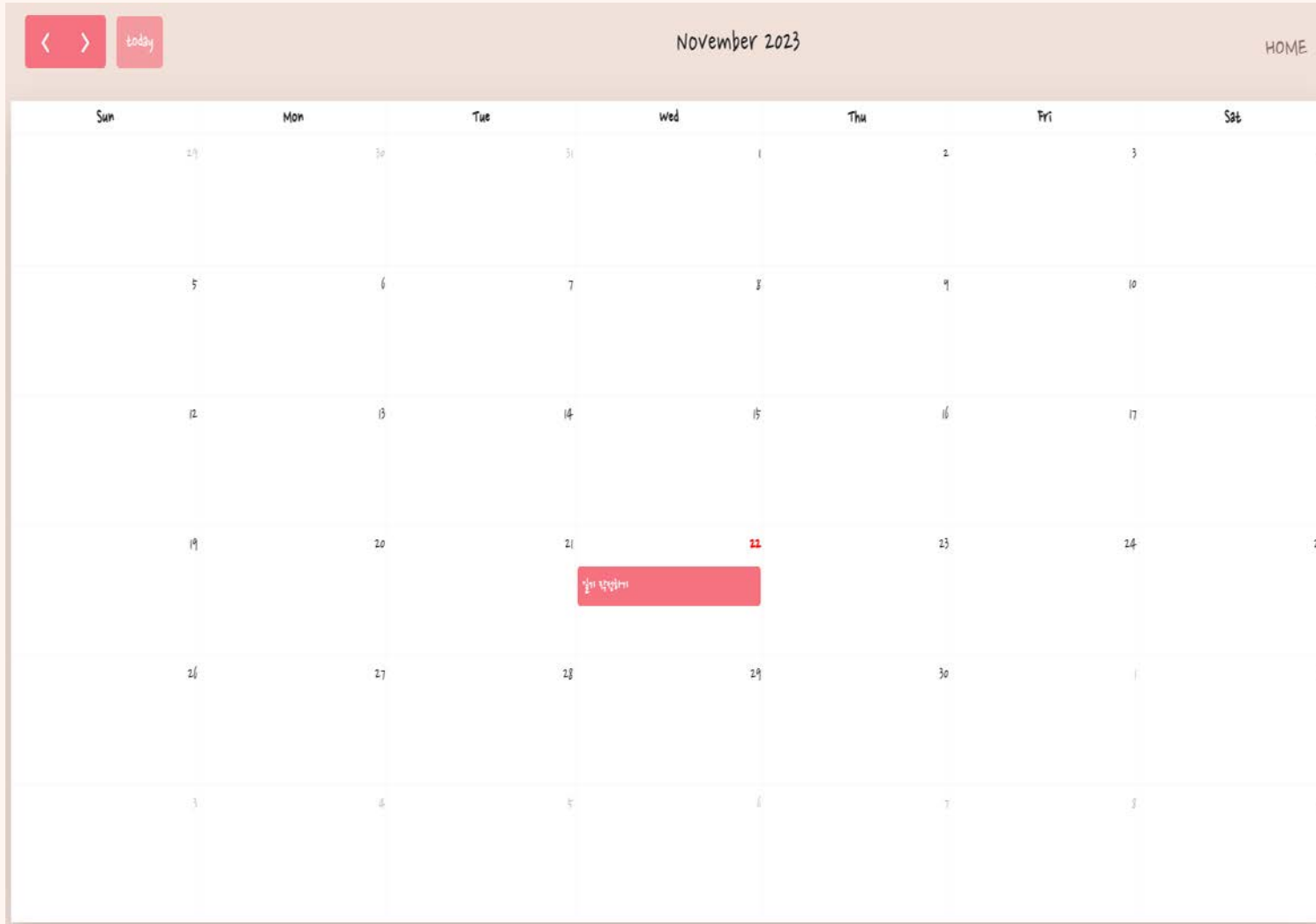


- 회원가입한 ID, E-mail을 활용하여
비밀번호 찾기 기능 구현
- ID와 E-mail이 일치하지 않을 경우
오류 페이지로 넘어가도록 구성
- ID와 E-mail이 일치할 경우
해당 E-mail로 비밀번호 재설정
링크를 회신

```
# python 표준 라이브러리인 base64 : urlsafe_base64_encode 함수 제공
# 유저의 primary key를 base64로 인코딩하여 토큰 생성
# 사용자의 고유 식별자(user.pk)를 안전하게 url에 넣기 위한 함수
uid = urlsafe_base64_encode(force_bytes(user.pk))
# Django에서 제공하는 유저에 대한 기본 토큰 생성(django.contrib.auth.tokens 내장 모듈)
# 비밀번호 재설정 토큰 생성
token = default_token_generator.make_token(user)
# 이메일에 보내질 제목 및 템플릿(이메일 내용) 설정
subject = '비밀번호 재설정'
email_template_name = "wecando/password_reset_email.html"
# 이메일 템플릿에 전달할 컨텍스트 설정
c = {
    "email": user.email,
    'domain': current_site.domain,
    'site_name': '11:57',
    'uid': uid,
    "user": user,
    'token': token,
}
# 이메일 내용 렌더링
# 템플릿 렌더링 -> 문자열 생성 (이메일 본문 생성)
email = render_to_string(email_template_name, c)
```

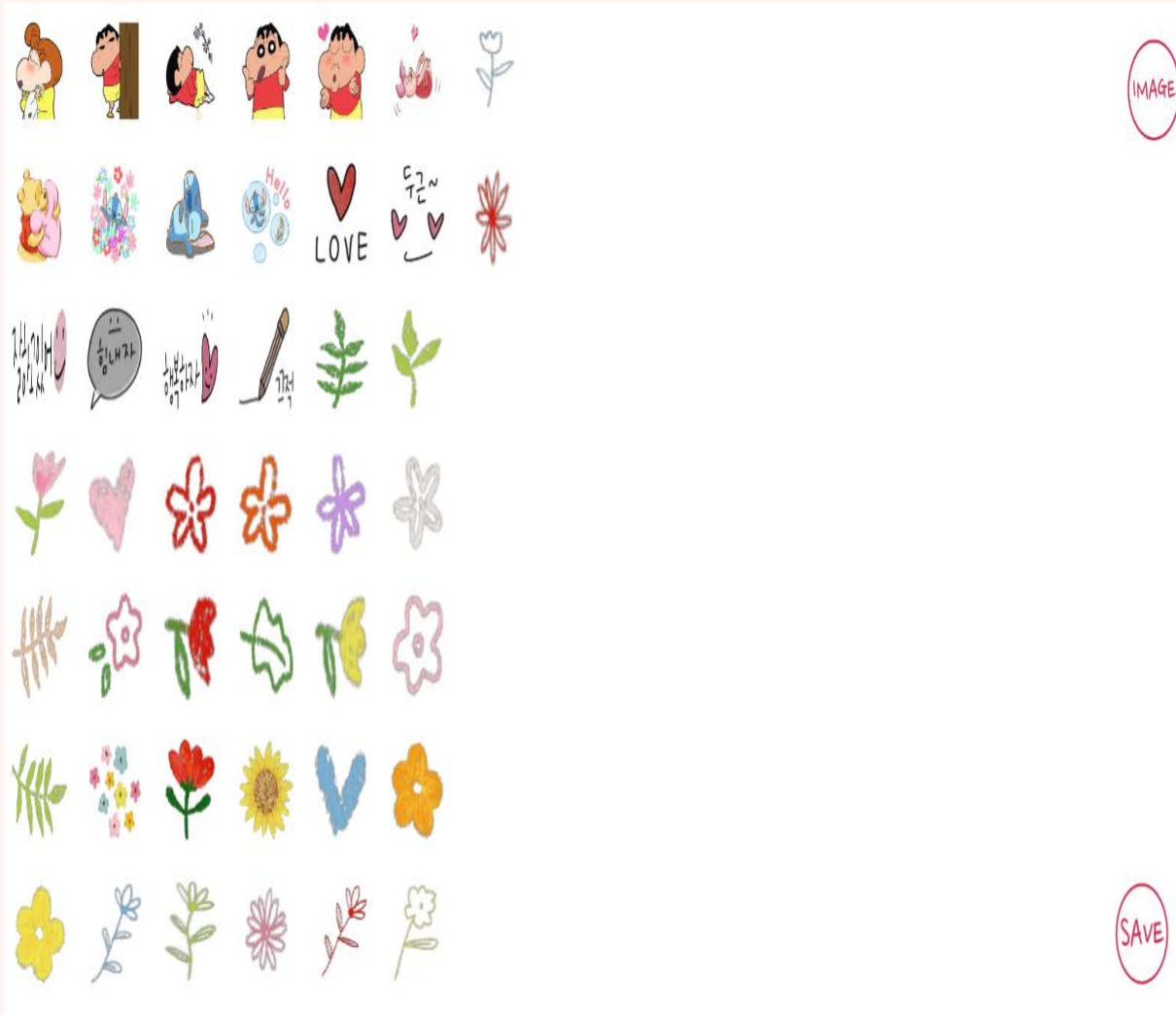
- Django에서 제공되는 내장 모듈을 활용
 - SMTP 설정을 통해 관리자 이메일 자동 로그인
 - 비밀번호를 재설정하기 위한 토큰 생성
- 이메일 템플릿에 컨텍스트를 설정하고, 사용자 계정 이메일로 전달

※ [참고 페이지 링크](#)



- **Full Calendar 라이브러리 활용**
 - 오픈소스 라이브러리
 - 달력, 일정, 스케줄러 구현시
많이 활용
- **패키지를 다운로드 하여**
웹의 전체적인 디자인에 맞게
구성을 변경하여 활용

※ [참고 페이지 링크](#)



- **Konva 라이브러리 활용(Java Script)**
 - 스티커 영역
 - Drag & Drop, Resize 기능 구현
- **HTML2Canvas 라이브러리 활용**
 - 해당 영역 내 데이터와 url을 생성
 - 이미지를 다운로드

3. 웹 기능 구현

감정분석

We Can Do

delete edit

신난다

2023년 11월 1일 3:15 오후 🧡

뭐, 일상은 평범하지만 나에게는 즐거운 순간들이 많아.

오늘도 한없이 흘러가는 시간에 즐거움을 찾았어.

작은 행복이 모여 특별한 하루를 만들어.

감정 분류 결과

안녕하세요

안녕하세요 마지막 프로젝트가 끝났어요 신이 나네요.

저희가 분석한 감정은 기쁨입니다.

잘못 분석했다면 아래에서 올바른 감정을 선택해주세요.

분류

잘 분류했어요



SAVE



환상동화 (Secret Story of the Swan) - IZ*ONE (아이즈원)



- Gabriel García Márquez: One Hundred Years of Solitude

"It's enough for me to be sure that you and I exist at this moment."

3. 웹 기능 구현


감정분석

We Can Do


November 2023							HOME
Sun	Mon	Tue	wed	Thu	Fri	Sat	
			1	2	3	4	
아니! 😞	기분 전바 😞	서러함 😞	서러함 😞	기분전 😞	????????????? 😞	힘들! 😞	
5	6	7	8	9	10	11	
내일은 일요일 😞	일요일인데! 😞	서러 😞	힘들 😞	동아들 이혼 😞	사직서 😞	ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ ^^ 😞	
12	13	14	15	16	17	18	
방학! 😞	The chaser 😞	disappointment 😞	angry 😞	rollercoaster of emotions 😞	tough day 😞	frustration lingered 😞	
19	20	21	22	23	24	25	
determined 😞	disagreement 😞	delight 😞	magic 😞	worry 😞	satisfaction 😞	communication 😞	
26	27	28	29	30			
teamwork 😞	social gathering 😞	레스토랑 식사 😞	일이 많음! 😞				

11시 57분


Team

 우리는 능히 해낼 수 있다 (우해)

⋮ RULE

 rule


회의

 일별 회의록


웹 구성도

 웹 페이지 구성도

머신러닝 모델

 모델 코드

To-Do List

 전체

- 노션을 활용하여 팀내 전달사항 공유
- 각자의 페이지를 구성하여,

분담하고 있는 업무를 명확히 하고

일을 반복적으로 처리하는 것을

미연에 방지
- 오전과 오후 회의를 통해 할 일을

구체화

※ [Notion Team Page](#)



○ DY	origin/main	origin/HEAD	main	Merge branch 'main' of https://github.com/itsc-ai/wecando	22 11 2023 15:30
origin/hy_09	다이어리 표지 꾸미는 기능 구현				22 11 2023 15:25
origin/ody	영어 모델 추가 및 한글 모델 수정 자잘한 디자인 수정				22 11 2023 15:23
	다이어리 표지 꾸미는 페이지 기능 구현				22 11 2023 15:23
	전체 디자인 수정 각종 기능 정상 동작하게 수정 및 이그노어에 경로 추가 일기장 모델 이미지 파일 null입력 불가하게 수정 언어 및 시간 한국어로 수정				17 11 2023 16:56
	Merge remote-tracking branch 'origin/OHD' into hy_07				16 11 2023 9:38
	모델 올려서 동작하게 수정 및 db에 데이터 올려서 해당 감정에 맞는 데이터 뜨게 수정				15 11 2023 21:15
	원래 있던 음원 파일 삭제 일기 작성 페이지 수정 비밀번호 찾는 기능 구현 및 페이지 디자인(views.py, urls.py, settings.py에 필요 코드 추가) 필요한 버튼 이미지 추가 다이어리				15 11 2023 17:22
	일기 쓰는 페이지 수정 비밀번호 찾는 기능 구현 페이지 디테일 디자인 수정				15 11 2023 16:49
	Merge branch 'main' of github.com:itsc-ai/wecando				10 11 2023 9:20
	간단한 링크 수정				10 11 2023 9:14
origin/hy_06	일기 수정할 forms.py models.py 수정 일기 수정, 삭제 views.py 수정 일기 수정할 페이지 생성				10 11 2023 9:11
	Merge branch 'main' of https://github.com/itsc-ai/wecando				8 11 2023 9:47
origin/hy_05	일기 작성 페이지 필드 수정				8 11 2023 9:30
	캘린더 설정, db에서 데이터 불러들여서 해당 날짜에 일기 링크 출력 일기 링크 클릭시 해당 일기 상세 페이지로 이동 오늘날짜에 일기작성된 것이 없으면 일기 작성 링크 생				8 11 2023 9:28

- Back-end에서 깃을 활용하여 웹 구현 및 DB를 직접적으로 관리
- 이후 웹의 구성이 어느정도 이루어지고 나서 다른 팀원들과 공유하여 디자인 및 세부사항 수정

4. 한계 및 방향성

팀내 협업사항

We Can Do

<다이어리 표지 생성 페이지>

로고

로그인 상태 표시

스티커 영역

업로드 이미지 및 꾸미기 영역

IMAGE

SAVE

Footer

<감정 분석 결과 확인 페이지>

로고

로그인 상태 표시

일기 내용

감정 분류 결과
일치하지 않을 시 선택 가능

SAVE

Footer

<상세 페이지>

로고

로그인 상태 표시

Calendar

Delete edit

작성 날짜, 감정

일기 내용

추천 음악

추천 명언

업로드 이미지

Footer

- **한계**

- 감정 분류의 한계 : 이전의 연구나 머신러닝, 딥러닝을 살펴보면 감정의 분류 이진으로 진행
- 웹 페이지 Drag & Resize 구현에서 스티커 수가 제한될 수 밖에 없었다는 점
- Tableau와 연동하여 감정의 빈도를 보여주는 히스토그램을 작성할 수 있었으나,
실시간 데이터베이스 연동에 어려움, ID 별로 통계를 보여줘야 했으나 적용이 어려웠음

- **추후 개선방향**

- 감정 분류에서는 객관적인 기준의 확립
- 데이터를 수집하고 전처리 방식 개선
- 웹의 경우 기획 단계에서 세부적인 사항 결정이 필요
- 시각화 프로그램 활용 개선

※ 참고문헌

- 일기쓰기를 이용한 긍정적 자기암시가 학생의 자아존중감 향상과 우울증 감소에 미치는 효과
- 정도교(2008)
- 문장에 내재한 감정 분석을 위한 감정 분류 방법(특허 정보)
- 가톨릭 대학교 산학 협력단
- 딥러닝 모델BERT과 감정어휘사전을 결합한 음원가사 감정분석
- 윤경섭, 오종민(2022)
- 욕구충족 영상콘텐츠 브이로그 ASMR 먹방 이용동기 수용자 특성시청만족도에 관한 연구
- 강미정, 조창환(2020)

※ 참고사이트

- Ekman 관련 내용

- https://en.wikipedia.org/wiki/Paul_Ekman

- Full Calendar

- <https://m.blog.naver.com/lifetripper/221930938974>

- 비밀번호 재설정 관련 내용

- <https://velog.io/@oen/Django>

- 모델의 정확도

- <https://hoit1302.tistory.com/159>



Diary

오늘 하루를 적어주세요!

Q & A

감사합니다.