MOODY
MOVIE

MOODY
MOVIE

목차

MOODY
MOVIE

프로젝트 개요

MOODY
MOVIE

프로젝트의 목적 및 목표

무디무비의 목적

사용자의 감정을 기반으로
맞춤형 영화를 추천하는
감정형 큐레이션 플랫폼 구축

무디무비의 목표

키워드 검색 기반 추천이 아닌,
사용자의 감정 상태를 분석하고
영화 장르와 작품을 제안하는
감정 공감형 챗봇

프로젝트의 중요성

최근 OTT 시장이
포화 상태에 이르면서,
단순한 '인기 영화' 추천이 아닌
개인 감정 맞춤형
콘텐츠 추천의 중요성이 커짐

MOODY
MOVIE

팀 구성원 소개

MOODY
MOVIE

강윤혁
Backend

라서윤
Backend

서민형
Frontend

윤동주
Frontend

MOODY
MOVIE

강윤헌

강윤헌

2001

GPT API 연동 및 챗봇 응답 처리
OpenAI GPT API 키 발급 및 환경 변수 설정

감정 분석 외 일반 대화 응답(질문/답변) 기능 확장

GPT 응답을 통해 사용자 맞춤형 영화 추천 메시지 생성

사용자의 입력 문장을 받아 GPT API로 전송하고 결과를 반환.

▶ 재생

＋ 내가 찜한 콘텐츠

MOODY
MOVIE

라서
윤

라서윤

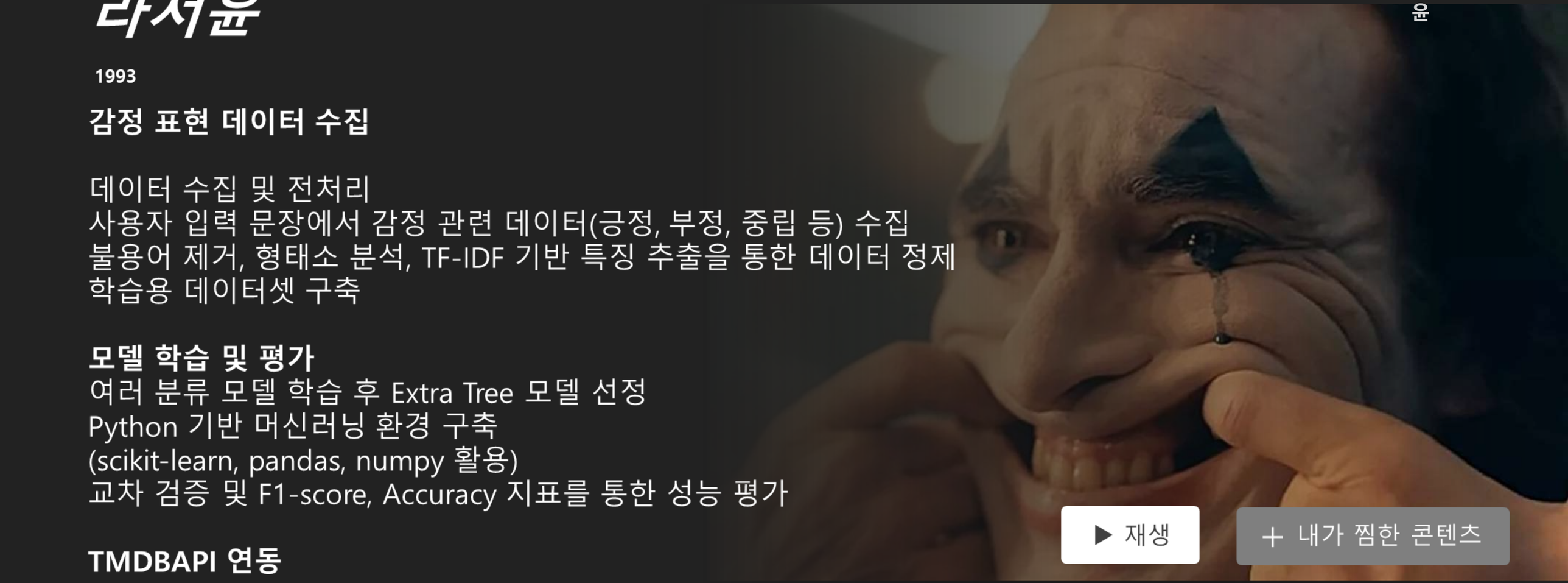
1993

감정 표현 데이터 수집

데이터 수집 및 전처리
사용자 입력 문장에서 감정 관련 데이터(긍정, 부정, 중립 등) 수집
불용어 제거, 형태소 분석, TF-IDF 기반 특징 추출을 통한 데이터 정제
학습용 데이터셋 구축

모델 학습 및 평가
여러 분류 모델 학습 후 Extra Tree 모델 선정
Python 기반 머신러닝 환경 구축
(scikit-learn, pandas, numpy 활용)
교차 검증 및 F1-score, Accuracy 지표를 통한 성능 평가

TMDBAPI 연동



▶ 재생

＋ 내가 찜한 콘텐츠

MOODY
MOVIE

서민형

서민형

2001

HTML, CSS, JavaScript를 기반으로 웹페이지 기본 구조제작

Flask 통신 연결

그룹 협업 환경 구성
Git 레포지토리 초기화 및 원격 레포지토리 연동
브랜치 전략, Pull / Push를 통한 협업 기반 마련
충돌 관리 및 코드 버전 관리를 통한 안정적인 개발 환경 유지

▶ 재생

＋ 내가 찜한 콘텐츠

MOODY
MOVIE

윤동주

윤동주

2001

HTML, CSS, JavaScript 기반 메인 화면 구성
PC·모바일 환경 대응을 위한 반응형 레이아웃 설계
영화 포스터 슬라이드, 검색 창, 챗봇 버튼 등 홈 화면 주요 인터랙션 요소 구현
Flask–MySQL 연동을 통한 감정 및 영화 추천 데이터 실시간 표시 기능 개발
기존 datalist 제거 및 로컬 스토리지 기반 검색어 기록 리스트 기능 구현

데이터 검증용 더미 데이터 삽입
감정 통계 시각화 기능 구현
“많이 추천된 영화 Top 10” 슬라이더 구성 및 UI 반영

Figma를 활용한 프로젝트 기획 및 시각적 콘셉트 수립
와이어프레임 및 프로토타입 제작을 통한 UI 구조 설계

MOODY
MOVIE

프로젝트 진행 일정

MOODY
MOVIE

1차

아이디어 회의 및 주제 선정

2차

웹 프론트, 챗봇 프론드&백

HTML,CSS, JS를 사용해 웹 제작

3차

데이터 추가 학습 및 배포

감정 외 사용자 니즈에 맞춘 모델 학습

배포

4차

PPT 작성 및 발표

프로젝트 요약

평가

MOODY
MOVIE

간트 차트

분야	기능	1주차					2주차				
		10월 13일	10월 14일	10월 15일	10월 16일	10월 17일	10월 20일	10월 21일	10월 22일	10월 23일	10월 24일
백엔드	프로젝트 기획										
	API 설계 및 구현										
	DB 설계										
프론트 엔드	프로젝트 기획										
	UI/UX설계										
	프로토타입 제작										
	사용자 인터페이스 개발										
	API 연동										
AI	프로젝트 기획										
	데이터 수집 및 전처리										
	모델 학습										
	모델 검증										

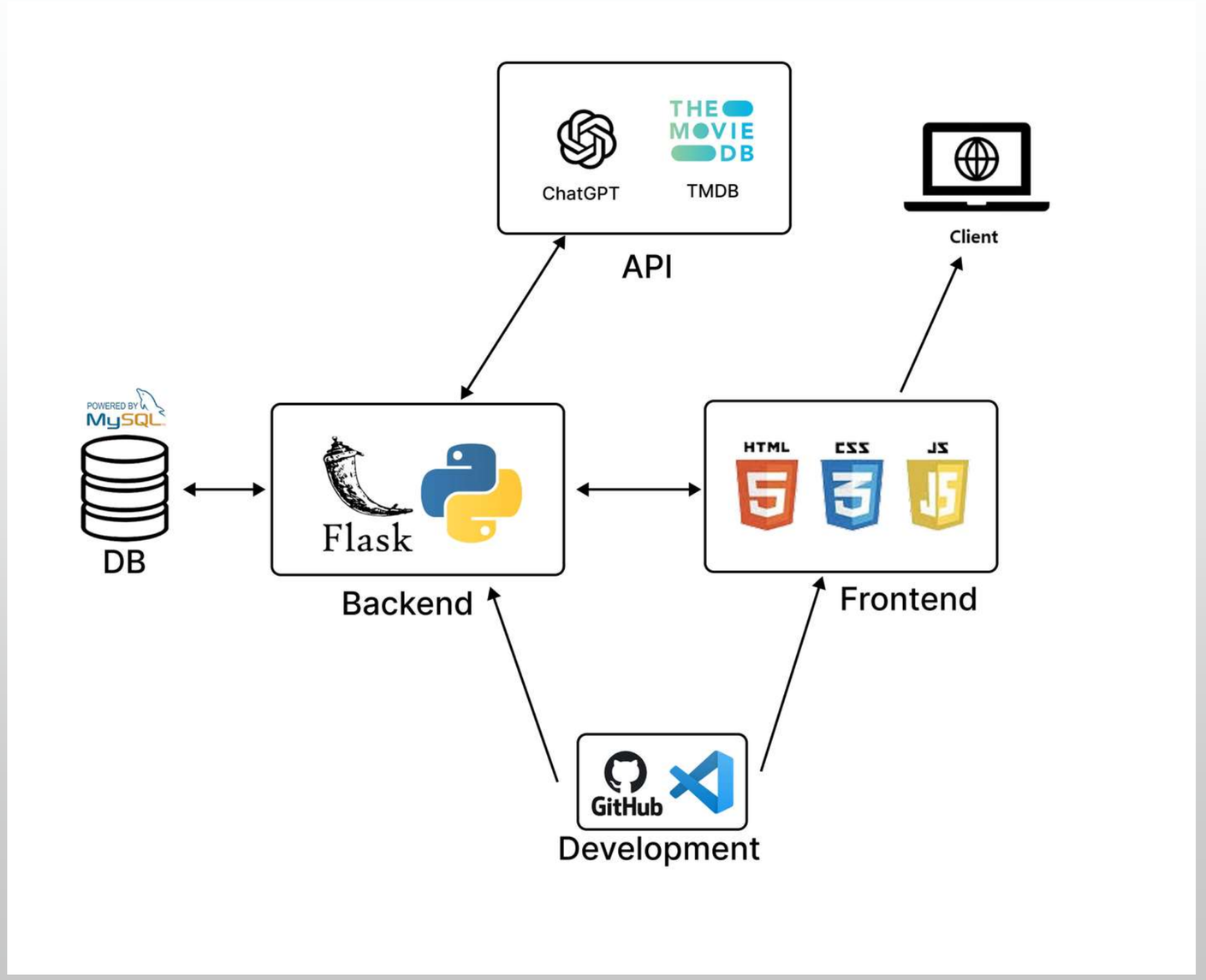
MOODY
MOVIE

프로젝트 구조

MOODY
MOVIE

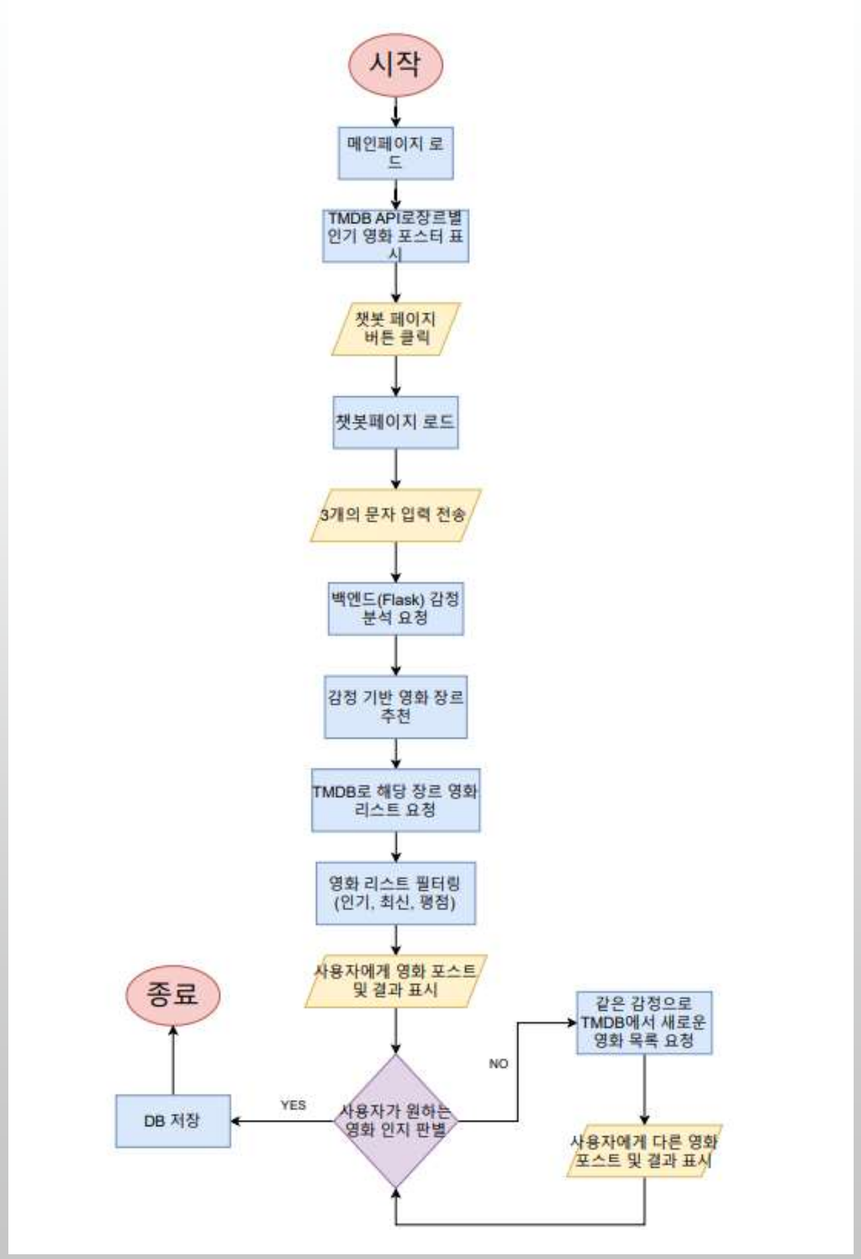
프로젝트 구성도

Flask 서버를 중심으로
Frontend, Database, API가
유기적으로 연동된
감정 기반 영화 추천 시스템 구조



MOODY
MOVIE

플로우 차트



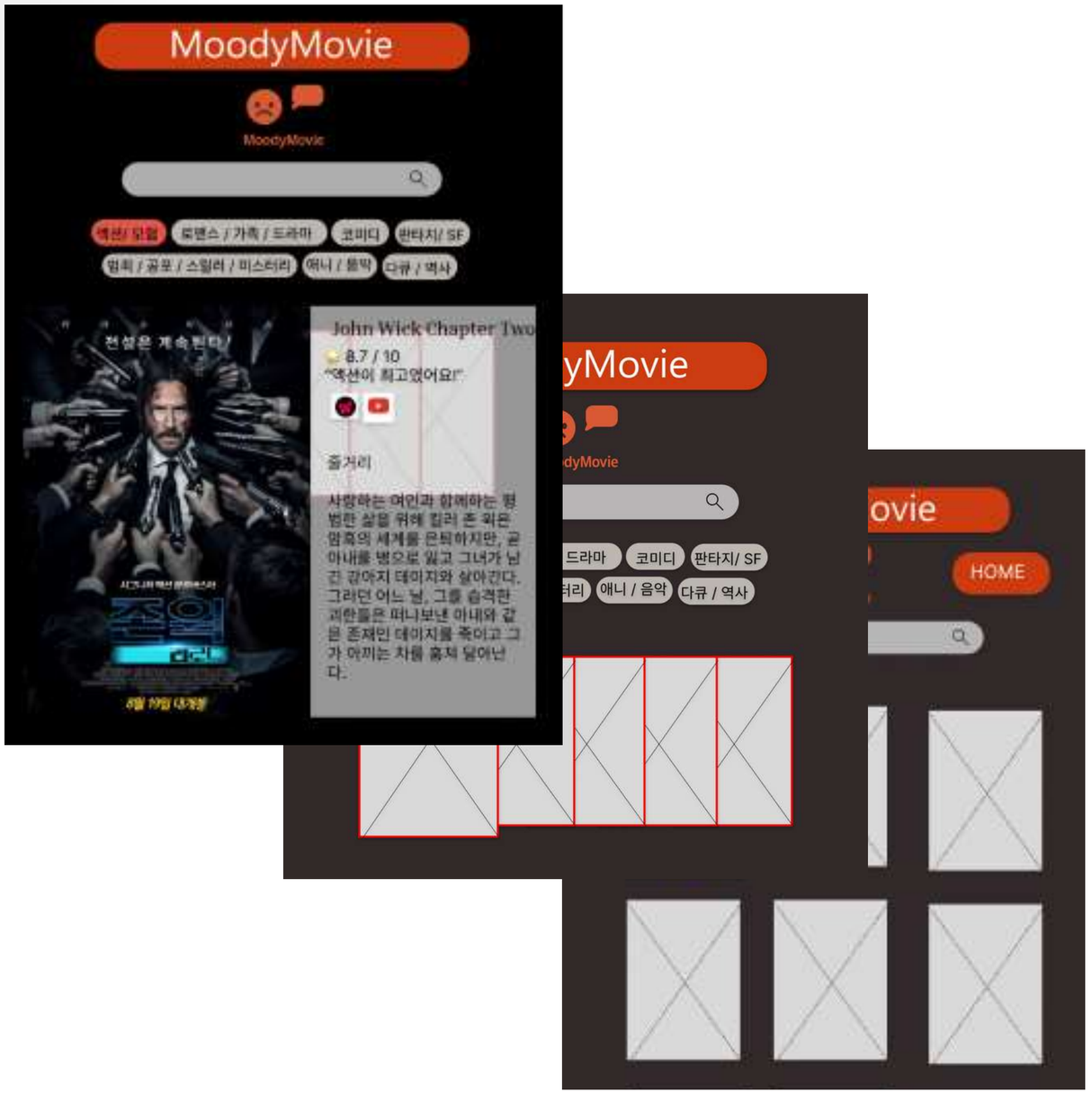
사용자가 입력한 문장을
Flask 서버가
감정 분석 모델을 통해 분류

그 결과를 기반으로
ChatGPT와 TMDB API를 연동해
감정에 어울리는 영화를 추천

해당 추천 결과를
MySQL 데이터베이스에
저장하는 전체 자동화 프로세스

MOODY
MOVIE

Figma 프로토타입



MOODY
MOVIE

프로젝트
주요 기능

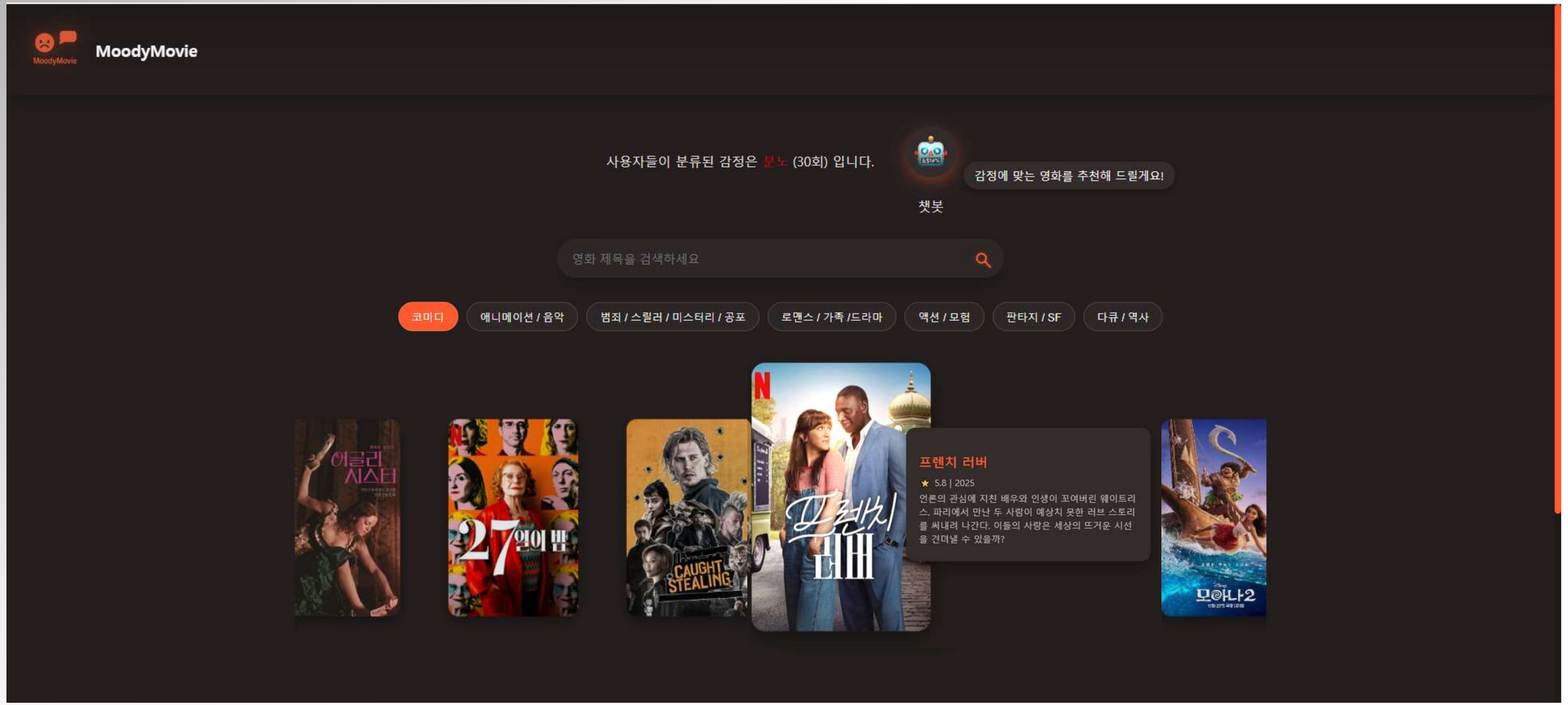
MOODY
MOVIE

프로젝트 주요 기능

- 영화 검색 기능
- 실시간 영화 추천 TOP 10
- 감정 기반 영화 추천 챗봇
- TMDB API 최신 영화 포스터 및 상세 정보 UI

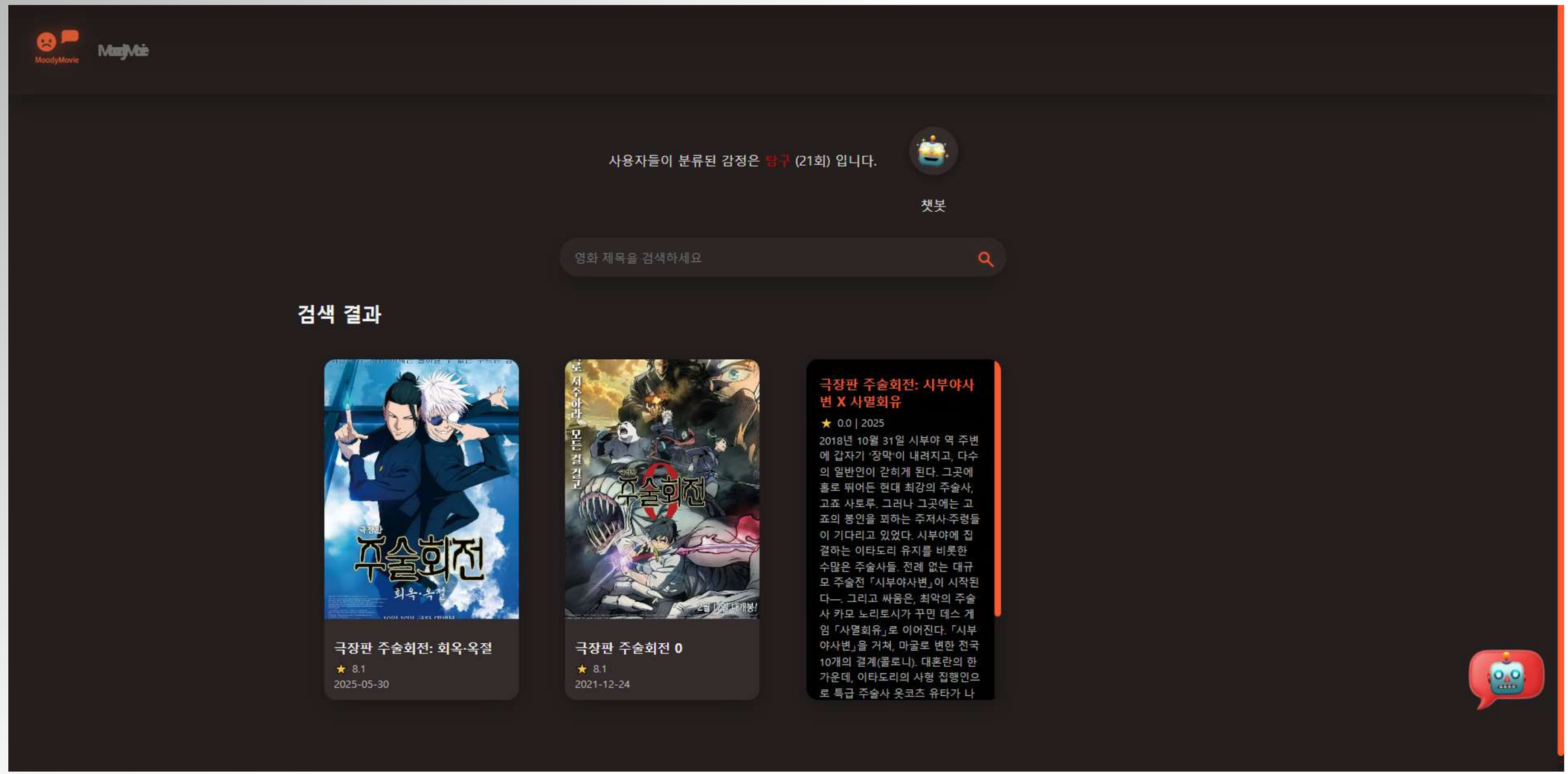
MOODY
MOVIE

메인 홈페이지



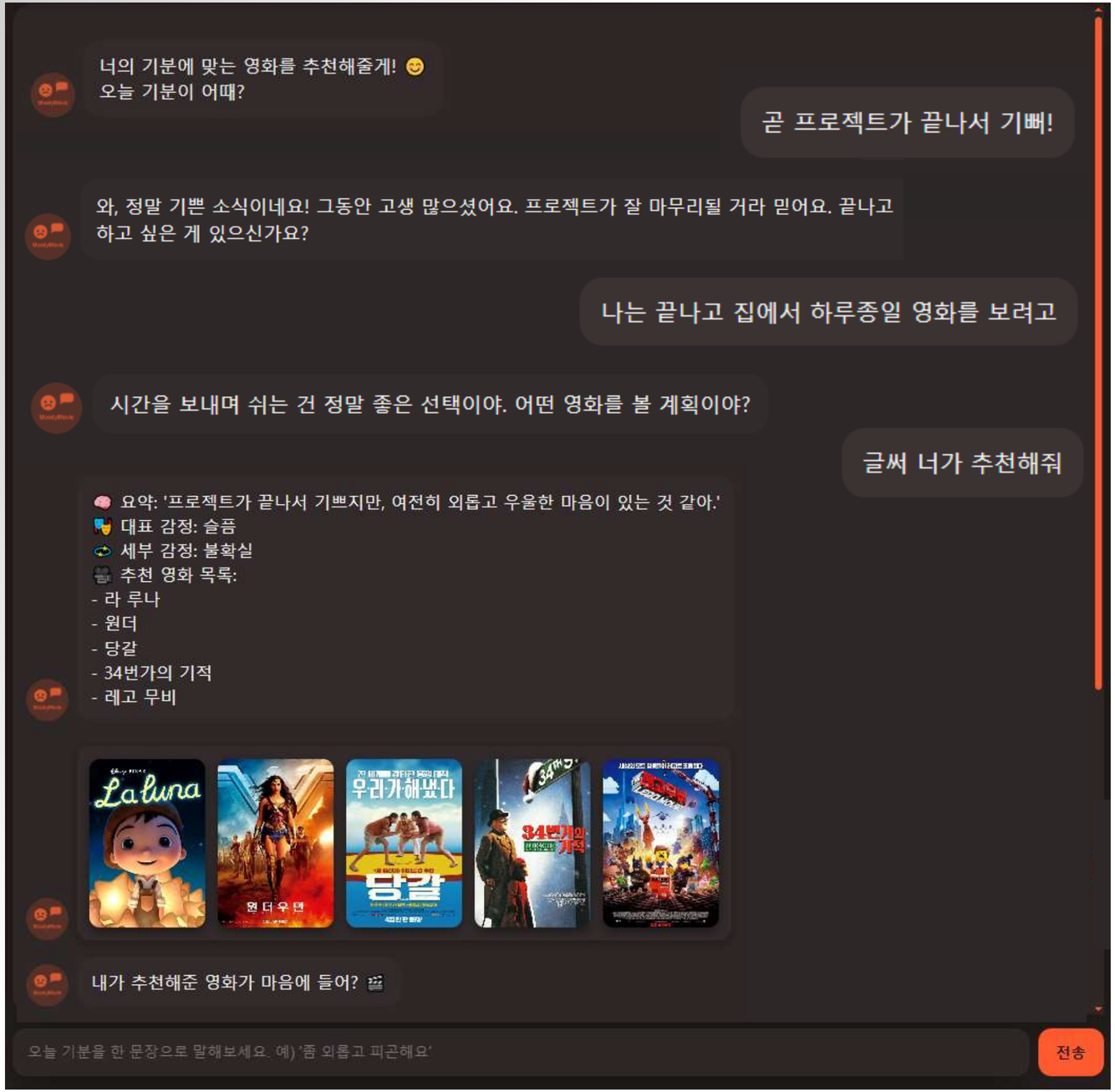
MOODY
MOVIE

검색 결과 UI



MOODY
MOVIE

무디 무비
챗봇 UI



MOODY
MOVIE

데이터 수집

MoodyMovie 감정 데이터 학습 준비 요
약 10754개　　약

```
탐구,호기심,혹시 저들이 뭔가를 숨기려는 것일까요?
탐구,호기심,왜 이 장소는 유독 기묘한 분위기를 풍길까요?
탐구,호기심,이 모든 질문에 대한 답은 과연 어디에 있을까요?
탐구,호기심,과연 우주 끝에는 무엇이 존재할까요?
탐구,호기심,저 어린 아이는 무엇 때문에 저렇게 집중하는 걸까요?
탐구,호기심,왜 어떤 사람은 유독 특별한 재능을 가지고 태어날까요?
탐구,호기심,이 고통의 의미는 대체 무엇일까요?
탐구,학습,저 인공지능은 수많은 기보를 학습해서 세계 챔피언을 이겼어요
탐구,학습,이 악기는 꾸준한 반복 연습을 통해 손과 악기의 일체감을 학습해야만 해요
탐구,학습,다른 부서와의 협업을 통해 프로젝트 관리의 복잡성을 학습했어요
탐구,학습,저는 저의 잠재력을 믿고 새로운 도전을 통해 스스로를 학습하는 중이에요
```

감정 사전: 약 384

감정,단어개	불안,겁나
불안,걱정	불안,무서움
불안,긴장	불안,두렵다
불안,두렵	불안,긴장감
불안,겁	불안,불안정
불안,조마조마	불안,무서
불안,혼란	불안,걱정스러
불안,불확실	불안,두근거
불안,위험	불안,무서워
불안,무섭	불안,불안해
불안,공포	불안,두려워
불안,두려	불안,초조하다
불안,떨려	불안,겁났다
불안,불안감	불안,긴장되
불안,초조해	불안,불안했
	불안,위축

MOODY
MOVIE

데이터 수집

Chat GPT 95% (gpt 데이터를 확인하고 수정함)
대표 감정: 7개
대표 감정 별 세부 감정: 15개

TMDB 기준 영화 장르를 확인한 후
각 감정에 장르를 매칭함.

사람마다 각 감정에 따라
보고 싶은 장르가
다르기 때문에 감정 분석 후에
사용자가 장르를 선택할 수도 있도록 함.

대표 감정	매핑된 주
😢 슬픔	드라마, 로맨 가족, 다큐
😊 행복	코미디, 애니 가족, 음악
😡 분노	액션, 범죄, 미스터리, 공
😤 스트레스	코미디, 음악 애니, 가족

MOODY
MOVIE

데이터 전처리

- 불용어 리스트 작성
  - 한글, 공백 제외 모든 문자 제거

```
# ------------------------------------------------------------
# 🧹 2. 불용어 리스트 (감정 분석에 불필요한 단어들)
# ------------------------------------------------------------
stopwords = [
    "은","는","이","가","을","를","에","에서","에게","께","한","하다",
    "그리고","그래서","그러나","하지만","또한","때문에","너무","정말",
    "그냥","조금","좀","또","또는","이나","거나","저는","나는","내가",
    "우리","너","니","그","그녀","이것","저것","거","게","걸","것"
]

# ------------------------------------------------------------
# ✨ 3. 텍스트 전처리 함수 정의
# ------------------------------------------------------------
def clean_text(text):
    # ① 문장에서 한글과 공백을 제외한 모든 문자 제거
    text = re.sub(r"[^ㄱ-ㅎㅏ-ㅣ가-힣\s]", " ", str(text))
    # ② 불용어 리스트의 단어 제거
    for sw in stopwords:
        text = text.replace(sw, " ")
    # ③ 불필요한 공백 정리 (연속된 공백을 하나로)
    text = re.sub(r"\s+", " ", text).strip()
    return text
```

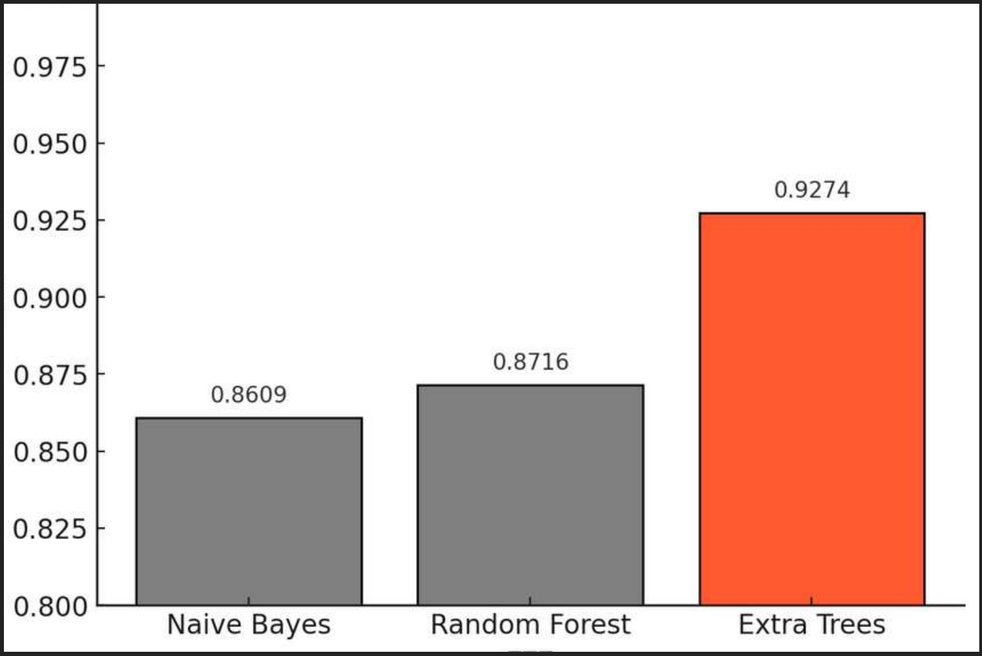
MOODY
MOVIE

데이터 분석 및 학습

Extra tree

Train Accuracy: 0.9988

Test Accuracy: 0.9274




```python
# 학습/테스트 데이터 분리 (stratify는 클래스 비율 유지용)
X_train, X_test, y_train, y_test = train_test_split(
    texts, labels, test_size=0.2, random_state=42, stratify=labels
)

# TF-IDF 벡터화 (문자 단위로 1~3글자 조합)
vectorizer = TfidfVectorizer(
    analyzer='char',
    ngram_range=(1, 3),
    max_features=8000,  # 단어(문자) 특징 수 제한
    min_df=2            # 최소 2회 이상 등장해야 사용
)

# 벡터화 진행
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# ExtraTreesClassifier 모델 생성
# GridSearchCV 결과로 얻은 최적 하이퍼파라미터 조합
et_model = ExtraTreesClassifier(
    n_estimators=674,         # 트리 개수 (클수록 안정적)
    max_depth=None,           # 트리 깊이 제한 없음
    min_samples_split=2,      # 분할 최소 샘플 수
    min_samples_leaf=1,       # 리프 노드 최소 샘플 수
    max_features='log2',      # 사용할 특징 수 기준
    class_weight='balanced', # 감정별 데이터 불균형 보정
    n_jobs=-1,                # 모든 CPU 사용
    random_state=42,          # 재현 가능한 결과
    verbose=1                 # 학습 로그 출력
)
```

MOODY
MOVIE

세부 감정 학습

감정을 더 **세분화(세부 감정)** 해서
학습 문맥 속에서
감정의 강도와 방향성 구체적으로 파악

‘슬픔’이라는 대표 감정 안에서도
‘후회’, ‘외로움’, ‘상실감’은
전혀 다른 분위기의 영화를 추천해야 함.

```
# ──────────────────────────────────────────────
# 😈 5. 대표감정별로 세부감정 모델 학습
# ──────────────────────────────────────────────
models = {}       # 대표감정별 학습된 모델을 저장할 딕셔너리
vectorizers = {}  # 대표감정별 TF-IDF 변환기를 저장할 딕셔너리

# “대표감정” 열에 있는 모든 감정 종류를 하나씩 반복
for main_emotion in df["대표감정"].unique():
    # 대표감정이 같은 데이터만 따로 모음
    sub_df = df[df["대표감정"] == main_emotion]

    # ⚠ 세부감정이 1개뿐이면 학습할 수 없으므로 건너뜀
    if len(sub_df["세부감정"].unique()) < 2:
        print(f"[{main_emotion}] 세부감정이 1개뿐이라 학습 생략")
        continue

    print(f"[{main_emotion}] 세부감정 모델 학습 중...")

    # 입력(문장)과 출력(세부감정) 나누기
    X = sub_df["대화"]
    y = sub_df["세부감정"]
```

MOODY
MOVIE

GPT 영화 추천 응답 생성 함수

GPT가 감정 요약

ML 모델이 감정 분류

TMDB/DB에서 영화 추천

결과를 챗봇 대화 형식으로 전달

```python
# GPT로 감정 요약 생성
summary_response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {"role": "system", "content": "너는 감정을 따뜻하게 요약하는 친구야."},
        {"role": "user", "content": summary_prompt},
    ],
)
summary_text = summary_response.choices[0].message.content.strip()
print("🧠 대화 요약문:", summary_text.encode("utf-8", "ignore").decode("utf-8"))

# 대표감정 예측
X = vectorizer.transform([summary_text])
predicted_emotion = model.predict(X)[0]

# 대표감정별 세부감정 예측
try:
    vec = sub_vectorizer.get(predicted_emotion)
    model_for_emotion = sub_model.get(predicted_emotion)
    if vec is not None and model_for_emotion is not None:
        X_sub = vec.transform([summary_text])
        # 🔽 여기서 확률 기반으로 최고 세부감정 선택
        probs = model_for_emotion.predict_proba(X_sub)[0]
        classes = model_for_emotion.classes_
        predicted_sub = classes[probs.argmax()]
    else:
        predicted_sub = "세부감정 없음"
except Exception as e:
    print("세부감정 분석 오류:", e)
    predicted_sub = "세부감정 없음"

# 감정에 맞는 영화 추천
genre_id = get_genre_by_emotion(predicted_emotion)
movies = get_movies_by_genre(genre_id)
movie_titles = [m["title"] for m in movies if isinstance(m, dict)]

# 추천 영화 목록을 대화 히스토리에 저장
conversation_history.append({
    "role": "assistant",
    "content": f"추천 영화 목록은 {', '.join(movie_titles)}야."
})
conversation_history.append({
    "role": "assistant",
    "content": "내가 추천해준 영화가 마음에 들어? 🎬"
})
```

MOODY
MOVIE

추천 이후 대화 처리
영화 메모리 저장 로직

사용자 입력을 반복적으로 받아
GPT 응답을 출력함.
감정 기반 추천이 불가능한 경우,
장르나 배우 등 다른 기준으로
추천하도록 안내함.

```
    # ============================================================
    # 🎬 3턴 이후 대화 GPT 자율모드 + 추천 영화 메모리기억
    # ============================================================
    elif turn_type == "after_recommend":
        followup_prompt = (
            "너는 감정 기반 영화 추천 친구야. "
            "사용자와 나눈 모든 대화를 기억하고 자연스럽게 이어가. "
            "사용자가 영화에 대해 이야기하면 감정적으로 공감하면서 답해. "
            "필요할 때만 평점, 줄거리, 배우, 장르, 분위기 등을 말해줘. "
            "추천해준 영화는 사용자가 아직 보지 않았다는 전제로 이야기해. "
            "사용자가 '새로운 영화', '다른 영화', '바꿔' 등을 말하면 "
            "이전 추천 목록과 사용자의 감정을 참고해서 비슷하거나 반대 분위기의 영화를 추천해줘. "
            "평점이나 상세정보를 묻는다면 대화 맥락에서 어떤 영화를 의미하는지 스스로 추론해. "
            "만약 영화와 직접 관련 없는 이야기를 하면, 공감은 짧게 표현하고 "
            "그 감정과 연결될 수 있는 영화 주제로 자연스럽게 대화를 이어가. "
            "예를 들어 사용자가 일상 이야기를 하면, 그 기분과 비슷한 영화나 장면을 떠올려서 언급해. "
            "절대 기계적으로 묻지 말고, 사람처럼 부드럽게 대화의 흐름을 유지해."
        )
        response = client.chat.completions.create(
            model="gpt-4o-mini",
            messages=[
                {"role": "system", "content": followup_prompt},
                *conversation_history[-12:],  # 최근 12턴만 기억 유지
                {"role": "user", "content": user_msg},
            ],
        )

        gpt_reply = response.choices[0].message.content.strip()
        conversation_history.append({"role": "assistant", "content": gpt_reply})

        # 새 영화 제목만 메모리에 추가
        import re
        new_titles = re.findall(r'"([^"]+)"', gpt_reply)
        for t in new_titles:
            if t not in recommended_movies_memory:
                recommended_movies_memory.append(t)

        return jsonify({"reply": gpt_reply})

except Exception as e:
    print("❌ /chat 오류:", e)
    return jsonify({"reply": "서버 오류 발생"}), 500
```

MOODY
MOVIE

시연

MOODY
MOVIE

소감 및 평가

MOODY
MOVIE

강윤
헌

소감 : 생각보다 챗봇이 이끄는 대로 따라주지 않는 모습에서 종사자의 고충을 느낄 수 있었던 것 같다.
평가 : 완벽하다, 라고 말하기엔 아쉬운 부분도 많지만 그래도 뜻하는 바를 답으로 내뱉을 때는 기분이 좋았다.
보완점 : 배운 것 이상으로 코드를 짜는 것엔 다양한 방법과 방식이 있다는 것을 느꼈고 앞으로도 꾸준한 공부가 필요하다고 느꼈다.
마치 무수한 구멍이 뚫린 둑마냥 손을 가져가도 자꾸만 튀어나오는 아쉬움에 더 많은 배움의 필요성을 느꼈다.

라서
윤

소감 : 훈련이라는게 수업과는 다르게 정말 끝이 나지 않는다.
평가 : 최대한 똑똑하게 만들어보고 싶었으나 아직 부족한 부분이 많아 아쉽다.
보완점 : 예쁜 데이터가 많지 않아 훈련을 열심히 시켜도 과적합만 나오고 마음에 쏙드는 결과값이 나오지 않아 좀 더 정제된 품질 좋은 데이터가 있다면 좋겠다. 그렇다면 좀 더 성능좋은 감정평가가 나오지 않았을까 추후 딥러닝을 배운다면 딥러닝으로도 성능을 다시 평가해보고 싶다.

MOODY
MOVIE

서민
형

소감: 부족한 조장이지만 팀원들이 열심히 따라주고 스스로 할려고 노력하는 점에서 만족스러웠다.
평가: 챗봇과 사용자와의 대화 능력은 탁월하나 영화의 추천에 대해서의 완성도를 올려야 한다고 생각한다.
보완점: 챗봇의 데이터 수집을 조금 더 품질을 올려야 한다고 생각한다.

윤동
주

소감: 너무 힘들었지만, 함께 일할 줄 아는 사람으로 업그레이드 됐길, 자소서나 면접에 쓸 만한 스토리가 생겼기를 바란다.
평가: 디자인이 아쉽다. 챗봇 버튼 아래에 챗봇이라고 명시 하지 않았다면 챗봇으로 생각하기 쉽지 않았을 것이다. 아이콘 만으로 챗봇임을 확실히 알 수 있게 만들었다면 화면이 더 완성도 있었을 거 같다. 기획 단계에 시간을 더 들여서 계획을 구체화해서 프로젝트를 진행했다면 디자인할 때 더 잘할 수 있지 않았을까?
보완점: 과연 챗봇이 사용자에게 적합한 영화를 추천할 수 있을까? 감정 분석 전에 장르나 배우 등 사용자의 선호를 분석하고 감정에 맞춰 추천을 해주면 더 사용자에게 더 적합한 영화를 추천할 수 있지 않을까?

MOODY
MOVIE

QnA