츄러스 미 시스템 계획서

1. 개요

- **프로젝트명**: 츄러스미
- **개발 배경 및 목적**: 심리 상담 지원, 감정 분석, 맞춤형 미디어/병원 추천
- **주요 목표**: 사용자 감정 파악 → 요약·리프레이밍 → 맞춤형 콘텐츠 추천 → 결과 시각화

2. 요구사항 정의

2-1. 기능 요구사항 (Functional Requirements)

1. **감정 분석**
   ● 사용자가 입력한 텍스트를 기반으로 6가지 감정(기쁨(0), 슬픔(1), 분노(2), 상처(3), 당황(4), 불안(5)) 으로 분류
   ● 각 감정은 softmax 확률(%)로 출력하여 레이더 차트로 시각화 가능

2. **감정 기반 콘텐츠 추천**
   (1) 콘텐츠 추천
   ● 감정 키워드를 콘텐츠 메타데이터 태그와 매칭 (예: "불안" → ‘Calm’, ‘Healing’ 태그 곡)
   ● 사용자-콘텐츠 선호 행렬로 학습, 감정 태그를 추가 feature로 반영
   ● 멀티모달 추천 모델: 텍스트 감정과 미디어 설명(줄거리/가사) 임베딩을 결합해 유사도 기반 추천 가능
   (2) 데이터 수집(크롤링/오픈 API)
   ● 영화: TMDb API, KMDb(한국영화데이터베이스)
   ● 음악: Spotify API, Melon/Genie (크롤링)
   ● 병원: 공공데이터포털 → 보건복지부 의료기관 정보 API
   (3) 데이터 분류 및 목적 태깅
   ● 수집 데이터: 제목, 장르, 설명(줄거리/가사)
   ● 텍스트 전처리: 키워드 추출 (TF-IDF, 키워드 사전)
   ● 목적 태깅 예시
     - ‘위로’ : Healing, Calm, Hope, Family
     - 공감 : Youth, Failure, Growth, Reality
     - 회복 : Motivation, Challenge, Success
   ● 적용 모델
     - Rule-based 태깅(키워드 매핑) : 초기 빠른 적용

- KoBERT 기반 텍스트 분류 모델 : 감정과 콘텐츠 설명 매칭 자동화

3. 병원 및 상담소 추천

● 사용자가 주소 입력 : Kakao Map API / Google Geocoding API로 위도·경도 변환
● DB 내 병원 좌표와 거리 계산 (Haversine formula)
● 가까운 병원 Top-N 리스트 반환

4. 챗봇 인터페이스

(1) 긍정/부정 : 6가지 감정 분류 확장
● 단순 긍·부정(Sentiment)만으로는 부족 : 다중분류(Multi-class) 필요
(2) 접근 방법
● 이중 단계 분류
  - 1단계: 긍/부정(Positive vs Negative)
  - 2단계: 부정언어 → (분노, 슬픔, 불안, 상처, 당황) / 긍정언어 → (기쁨)
● 직접 다중분류 모델
  - KoBERT 기반 6클래스 분류 모델 학습
  - 레이블: {기쁨, 분노, 슬픔, 상처, 당황, 불안}
(3) 문맥 단위 감정 추출 기반
●Transformer (예: BERT, GPT)
  - 요즘 제일 많이 쓰는 모델
  - 대화 전체를 한꺼번에 보고, 문맥 관계를 잘 이해함
  - 예: “괜찮아”라는 말이 앞에 “너무 힘들다” 다음에 나오면, 진짜 괜찮은 게 아니라 “억지”라는 걸 파악
(4) 연속된 대화에서 감정 변화 추이를 시계열(time-series)로 시각화

5. 사용자 대시보드

● 개인 감정 변화 추이 그래프 제공
● 추천 콘텐츠 히스토리 확인
● 자가진단 결과 리포트 확인 가능

6. 관리자 대시보드

● 전체 사용자 감정 분포 집계
● 사용자 활동 로그, 사용자 게시글 모니터링
● 병원 추천 로그 확인

2-2. 비기능 요구사항 (Non-Functional Requirements)

1. **성능 요구사항** [ERD 참조부분 → 사용자 채팅 / 사용자 감정 로그 / 상담 리포트 요약 테이블 데이터량]

- 감정 분석 요청 시 **평균 20초 이내** 응답 제공 (CPU 환경, KoBERT 모델 고려)
- 동시 사용자 **3명 이내** 처리 가능 (시연용 환경 기준)
- 일일 **사용자 감정 로그 테이블 데이터 처리량**은 **1,000건 이하**로 가정

2. **보안 요구사항**

- 사용자 계정 비밀번호는 **bcrypt 해시 암호화** 후 저장
- bcrypt: 비밀번호 저장 전용 해시 함수 (같은 비밀번호라도 사용자마다 해시값이 달라짐)
- 개인정보는 최소한의 항목(이름, 성별, 나이, 주소, 이메일)만 수집
- API Key는 .env 파일에 보관하고, GitHub 등 외부 저장소에 노출 금지

3. **신뢰성 및 가용성**

- 발표/시연 시간(평일 주간) 동안 안정적으로 구동되는 것을 목표
- DB 및 로그 데이터는 **하루 1회 수동·자동 백업**
- 오류 발생 시 **사용자 감정 로그 테이블** 및 별도 로그 파일에 기록 후, 관리자가 확인 가능

4. **확장성**

- 초기에는 MySQL 단일 DB 사용하지만 추후 클라우드 RDS로 이관 목표
- 감정 카테고리(6종 → 기쁨, 분노, 슬픔, 상처, 당황, 불안)는 별도 컬럼으로 된 테이블로 설계
- 서버 구조는 단일 인스턴스 기준

5. **사용성 (Usability)**

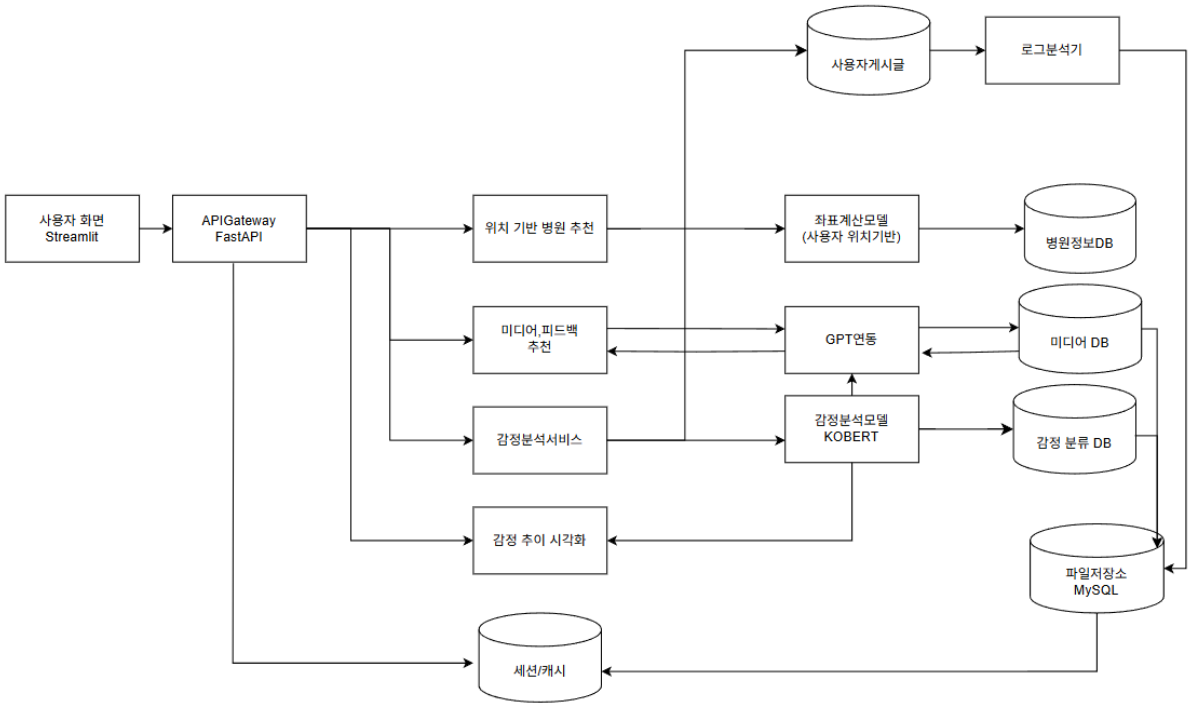
- PC 브라우저에서 간단히 접속 가능 (Streamlit 기반 UI)
- 회원가입 → 대화 입력 → 감정 결과 확인까지 4~5번 클릭 이내로 진행되도록 설계

6. **운영 및 유지보수**

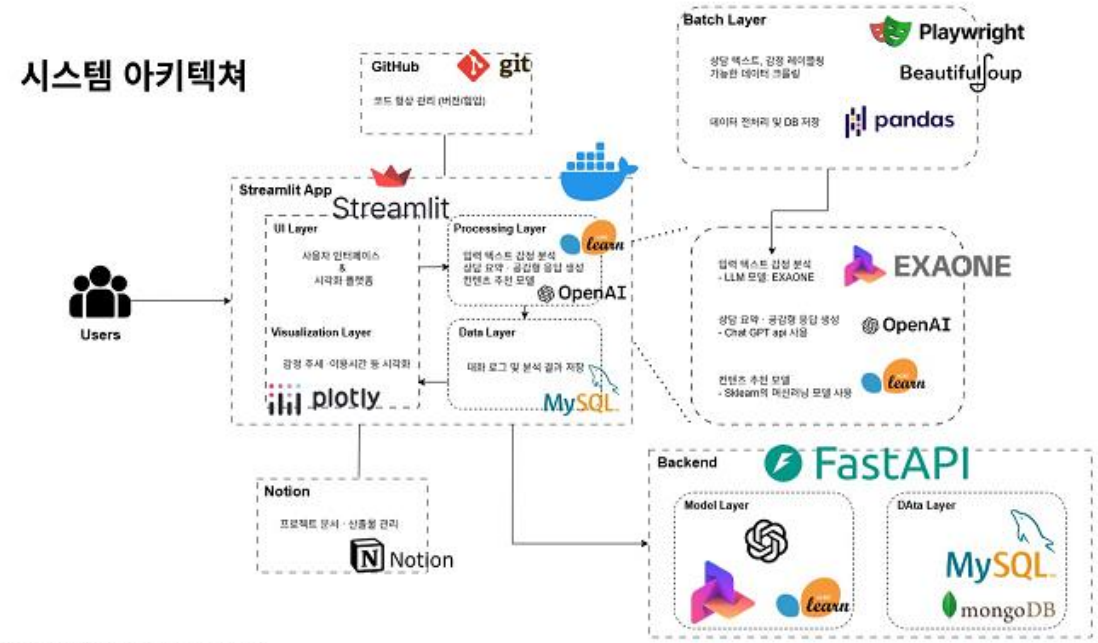
- 로그 관리**:** 사용자 요청 및 오류를 사용자 감정 로그 + CSV로 기록/저장
- 관리자 기능: 관리자는 대시보드를 통해 사용자 활동과 감정 기록을 모니터링하고, 필요 시 데이터 관리(삭제·백업) 수행 가능

3. 시스템 아키텍처

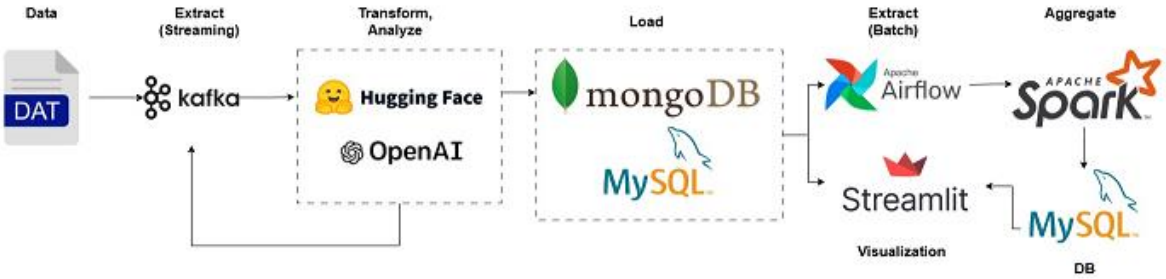
3-1. 시스템 흐름도


사용자게시글
로그분석기
사용자 화면
Streamlit
APIGateway
FastAPI
위치 기반 병원 추천
좌표계산모델
(사용자 위치기반)
병원정보DB
미디어,피드백
추천
GPT연동
미디어 DB
감정분석서비스
감정분석모델
KOBERT
감정 분류 DB
감정 추이 시각화
파일저장소
MySQL
세션/캐시

3-2. 시스템 아키텍처 / 데이터 파이프라인



데이터 파이프라인



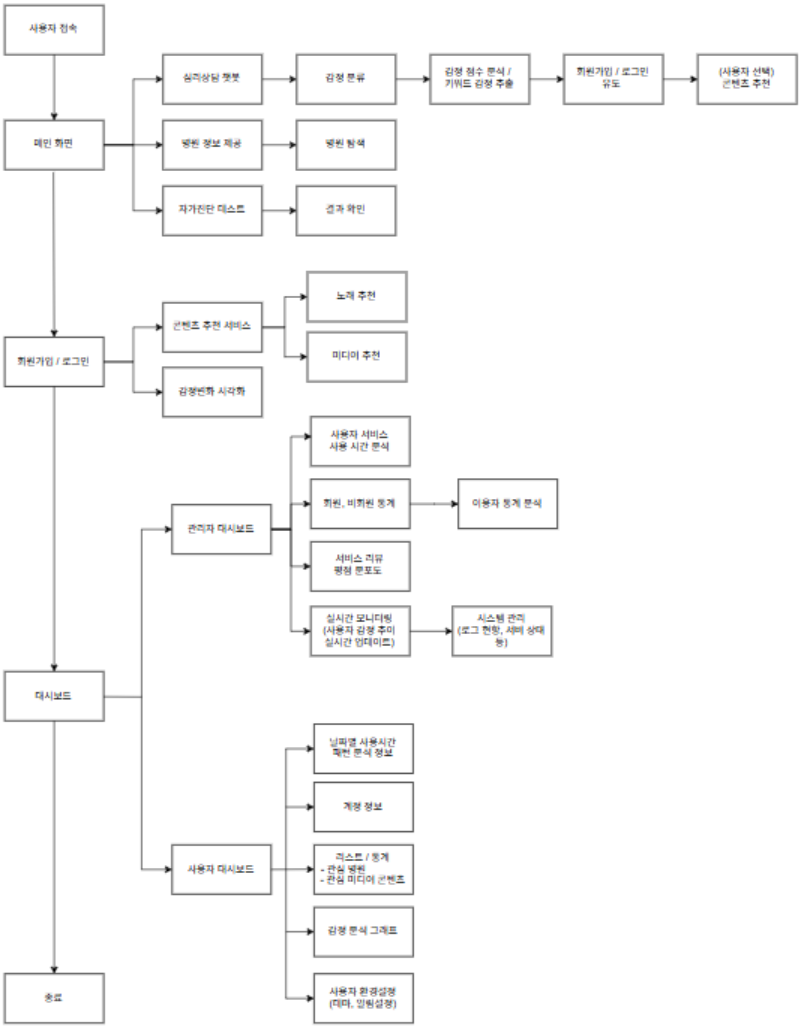
3-3. 구성 요소 설명

1. **Users(사용자) 서비스 이용자**
   - 서비스 최종 이용자
   - 텍스트 입력을 통해 감정 분석 및 추천 결과를 확인
   - 웹 브라우저(PC/모바일) 환경에서 접근
2. **Streamlit App (UI, Processing, Visualization, Data Layer)**
   - UI: 사용자 입력 인터페이스 제공 (대화 입력창, 그래프 출력 등)
   - Processing: 입력 텍스트를 감정 분석 API로 전달하고, 결과를 가공

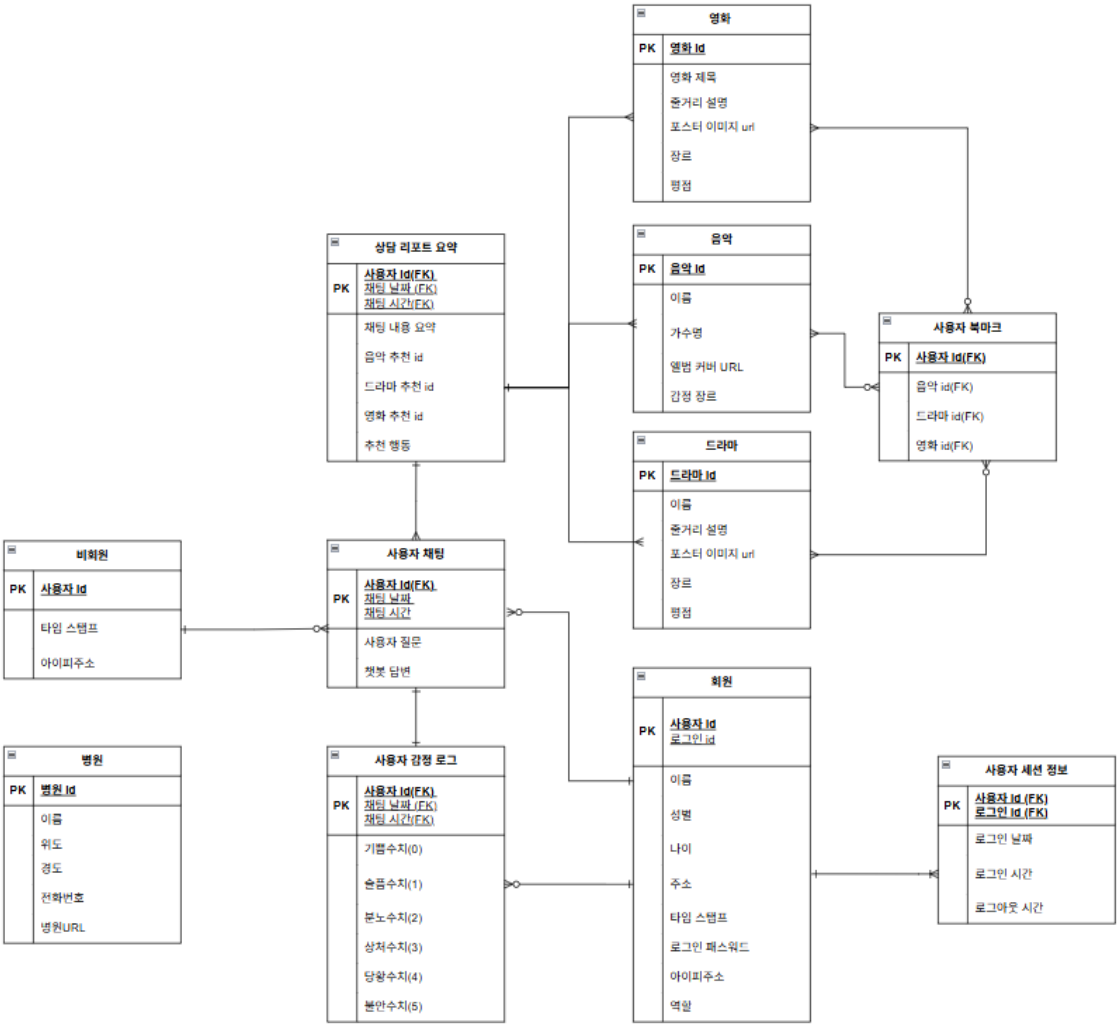
- Visualization: 감정 변화 추이, 분석 결과를 그래프로표시
- Data Layer: 분석 결과, 사용자 로그를 MySQL DB에 저장/조회

3. **외부 API (GPT / OpenAI)**
   - 텍스트 감정 분석 및 추천 문장 생성을 위한 API 활용
   - 감정 분류, 상담 요약, 콘텐츠 추천에 활용
4. **저장소 (MySQL, Mongo DB)**
   - 사용자 입력 로그, 감정 분석 결과, 추천 내역을 저장
   - ERD에 따라 User / EmotionResult / Log / Content / Hospital 등 테이블로 구성
5. **관리 도구**
   - GitHub: 코드 버전 관리 및 협업
   - Notion: 프로젝트 산출물, 문서, 일정 관리
   - 웹 브라우저(PC/모바일) 환경에서 접근
6. **배포 (Docker)**
   - Streamlit App을 Docker 컨테이너로 패키징
   - 환경 의존성을 줄이고, 발표/시연 시 동일 환경에서 안정적 실행 보장

3-4. 서비스 구성도


사용자 접속
메인 화면
심리상담 챗봇
감정 분류
감정 점수 분석 / 키워드 감정 추출
회원가입 / 로그인 유도
(사용자 선택) 콘텐츠 추천
병원 정보 제공
병원 탐색
자가진단 테스트
결과 확인
회원가입 / 로그인
콘텐츠 추천 서비스
노래 추천
미디어 추천
감정변화 시각화
대시보드
관리자 대시보드
사용자 서비스 사용 시간 분석
회원, 비회원 통계
이용자 통계 분석
서비스 리뷰 평점 분포도
실시간 모니터링 (사용자 감정 추이 실시간 업데이트)
시스템 관리 (로그 현황, 서버 상태 등)
사용자 대시보드
닉네임 사용시간 챗봇 분석 정보
계정 정보
리스트 / 통계
- 관심 병원
- 관심 미디어 콘텐츠
감정 분석 그래프
사용자 환경설정 (테마, 알림설정)
종료

4. 데이터 설계



4-1. 데이터 흐름도 (Data Flow Diagram)

1. **비회원**: 비회원 전용의 임의 사용자 계정 생성
2. **회원(사용자)**: 텍스트 입력을 기반으로 감정 분석 요청 → 감정 결과 로그 저장
3. **사용자 채팅**: 사용자의 입력 문장 → 감정 분석 모듈 처리 → 상담 리포트 요약 및 감정 결과 생성
   콘텐츠 추천: 감정 결과를 기반으로 영화, 드라마, 음악 등 추천 제공
4. **병원 추천**: 사용자 위치 정보 기반으로 병원 정보 제공
5. **사용자 세션 정보**: 로그인 세션과 연결되어 개인화된 감정 기록 및 추천 관리
6. **사용자 북마크**: 추천받은 콘텐츠(영화, 드라마, 음악 등)를 북마크 저장 가능

4-2. 저장 구조 (Data Storage Structure)

1. 사용자 계정(User)
    - 회원가입 정보(이메일, 닉네임, 비밀번호 해시 등) 저장
    - 로그인 세션과 연결
2. 사용자 감정 로그(Emotion Log)
    - 사용자 입력 문장, 분석된 감정 결과, 감정 강도 기록
    - 상담 리포트 요약 포함
3. 추천 콘텐츠(Content)
    - 감정별 추천 미디어 정보 저장 (영화, 드라마, 음악 등)
    - 사용자 북마크와 연결
4. 병원 정보(Hospital)
    - 사용자 위치 기반으로 제공할 수 있는 병원 정보 저장 (병원명, 주소, 위도/경도)
5. 세션(Session)
    - 사용자 로그인 상태, 최근 활동, 감정 히스토리 연결

4-3. ERD (Entity Relationship Diagram)

1. Member (회원)

- 핵심 속성: user_id, login_id, name, password, ip_address, latitude, longitude
- 역할: 서비스에 가입한 사용자의 기본 정보 관리
- 관계:
    - UserChat, EmotionLog, CounselingSummary, UserSession, UserBookmark와 1:N 관계

2. Guest (비회원)

- 핵심 속성: user_id, ip_address, created_at
  역할: 회원 가입하지 않은 사용자의 최소한의 접속 정보 관리
- 관계: 현재 다른 테이블과 직접 FK는 없음 (단순 기록용)

3. UserSession (사용자 세션)

- 핵심 속성: session_id, user_id, login_time, logout_time
- 역할: 회원의 로그인/로그아웃 기록 관리

- 관계: Member와 N:1 관계

4. UserChat (사용자 채팅 로그)

- 핵심 속성: chat_id, user_id, question, answer, chat_date, chat_time
- 역할: 사용자가 입력한 문장과 챗봇 응답 기록 관리
- 관계:
    - Member와 N:1 관계
    - EmotionLog, CounselingSummary와 1:1 또는 1:N 관계 (채팅 1건 → 감정분석/리포트 결과 여러 개 가능)

5. EmotionLog (사용자 감정 로그)

- 핵심 속성: emotion_id, chat_id, joy_score, sadness_score, anxiety_score, dominant_emotion
- 역할: 사용자의 채팅을 감정 분석한 결과 저장 (점수 및 대표 감정)
- 관계:
    - UserChat과 N:1 관계
    - Member와도 N:1 관계

6. CounselingSummary (상담 요약/추천)

- 핵심 속성: summary_id, user_id, chat_id, summary_text, action_plan, music_rec_id, drama_rec_id
- 역할: 한 세션/채팅에 대한 요약 및 추천 미디어 결과 저장
- 관계:
    - UserChat과 N:1 관계
    - Member와 N:1 관계
    - Music, Drama와 N:1 관계 (추천 결과 연결)

7. Hospital (병원)

- 핵심 속성: hospital_id, name, latitude, longitude, phone

- 역할: 위치 기반으로 추천할 수 있는 병원 정보 저장
- 관계: 현재 다른 테이블과 직접 FK는 없지만, Member의 위도/경도와 거리 계산을 통해 연계

8. Movie / Music / Drama (추천 컨텐츠)

- 핵심 속성: title, poster_url, emotion_genre (불안, 우울, 기쁨 등)
- 역할: 감정 기반 추천이 가능한 미디어 컨텐츠 저장
- 관계:
    - CounselingSummary와 연결되어 추천 결과로 활용
    - UserBookmark와 N:1 관계

9. UserBookmark (사용자 북마크)

- 핵심 속성: bookmark_id, user_id, movie_id, music_id, drama_id
- 역할: 사용자가 마음에 들어 한 컨텐츠(영화/음악/드라마)를 저장
- 관계:
    - Member와 N:1 관계
    - Movie, Music, Drama와 각각 N:1 관계

5. 모듈 설계

시스템의 기능적 구조 (Layer 기반)

1. **UI Layer**
    - 사용자 입력창 (텍스트 입력)
    - 결과창 (감정 분석 결과/추천 콘텐츠 표시)
    - 차트 영역 (시각화 결과)
2. **Processing Layer**
    - KoBERT: 감정 분류 모델 (다중 클래스 분류)
    - GPT API (OpenAI): 상담 요약 문장, 자연스러운 대화 응답 생성
    - 추천 모델: 감정 기반 콘텐츠 추천 (Scikit-learn 활용)
3. **Data Layer**
    - SQL 저장 (MySQL DB)
    - 사용자 로그, 감정 결과, 추천 기록 저장

4. **Visualization Layer**
    - 감정 추세 그래프 (시계열 기반 변화 시각화, Plotly)
    - 병원 지도 (Folium 기반 위치 표시)

6. 기술 스택

실제 사용하는 기술명/라이브러리

1. **Frontend / Visualization**
    - Streamlit (웹 UI/UX)
    - Plotly (차트, 시각화)
    - Folium (지도 시각화)
2. **ML / NLP**
    - KoBERT (감정 분류) *[모델 변경 가능성 있음]*
    - GPT API (OpenAI, 상담 요약 및 응답)
    - Scikit-learn (간단한 추천 모델, 예: 콘텐츠 기반 추천)
3. **Storage**
    - MySQL (사용자/로그/분석결과 저장소)
4. DevOps / 협업 도구
    - Docker (배포 환경 통합)
    - GitHub (코드 관리)
    - Notion (산출물 관리, 일정 관리)

7. 배포 설계

1. 기본 설계
    - Docker 기반 단일 컨테이너 실행
    - Streamlit + 모델(KoBERT/GPT API) + MySQL DB 포함

2. 확장 설계
    - Docker Compose를 이용해 다중 컨테이너 구조로 분리
        - Frontend (Streamlit)
        - Backend (FastAPI, 모델 서버)
        - DB (MySQL)
    - 확장 시 서비스 안정성 및 관리 용이성 확보 가능

8. 확장/발전 계획

- FastAPI 백엔드 분리
- 실시간 상담 기록 분석/추천 고도화
- 모바일 앱 서비스 확장?