

2025 캡스톤 디자인

‘AnOn’ 프로젝트 보고서

: AI 기반 심리 케어 서비스

한양여자대학교 빅데이터과
Team. WAVIEW

제출일: 2025.11.19.

팀원: 원희, 김빈나, 이예린, 윤서연, 정수연

목차

1. 프로젝트 계획서	
가. 과제 개요	4
나. 과제의 필요성 및 목적	4
다. 주요 기능 및 구현 요소	5
라. 개발 및 추진 방법	6
마. 기대효과	9
바. 사용 기술 소개	9
2. 요구사항 명세서	
가. 감정 분석 및 심리지원 정보 제공	22
나. 감정 변화 추적 및 시각화	22
다. 상황 맞춤 심리 안정 콘텐츠 제공	23
라. 회원 관리	23
3. 분석 활동	
가. 회원 관리	24
나. 감정 분석 및 심리지원 정보 제공	26
다. 감정 변화 추적 및 시각화	26
라. 상황 맞춤 심리 안정 콘텐츠 제공	27
마. 정서적 챗봇 대화 가능	27
4. 기능 명세서	
가. 회원 관리	28
나. 감정 분석 및 심리지원 정보 제공	28
다. 감정 변화 추적 및 시각화	29
라. 상황 맞춤 심리 안정 콘텐츠 제공	29

5. 설계	
가. Data 설계	30
나. 화면 설계	36
다. Logic 설계	38
라. Architecture	52
6. 학습	
가. 감정분석 모델 학습을 위한 데이터셋	60
나. 감정분석 모델 학습 과정	65
7. 코딩	
가. GitHub	70
나. 실행 화면	70
8. 테스트	
가. Test case	71
9. 회의록	74

1. 프로젝트 계획서

가. 과제 개요

본 과제는 창의자유과제로 수행되는 「인공지능을 이용한 정서 안정 지원 앱 개발」이며, 프로젝트명은 AnOn이다. 과제 수행 기간은 2024년 12월 22일부터 2025년 10월 30일까지로 설정하였다. 본 프로젝트에서는 KcELECTRA-small-v2022, LLaMA, Neon DB, FastAPI, MediaPipe, TensorFlow Lite, Hugging Face Spaces, Google Colab 등의 기술을 주요 기반으로 활용한다.

과제명: 인공지능을 이용한 정서 안정 지원 앱 개발

과제 구분: 창의자유과제

프로젝트명: AnOn

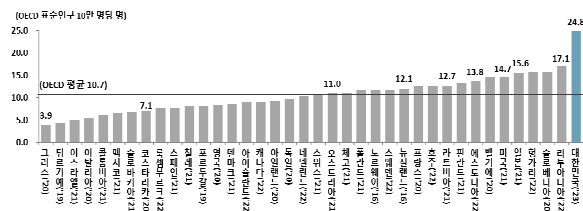
수행기간: 2024.12.22 ~ 2025.10.30

주요 기술: KcELECTRA-small-v2022, LLaMA, Neon DB, FastAPI, MediaPipe, TensorFlow Lite, Hugging Face Spaces, Google Colab

나. 과제의 필요성 및 목적

현대 사회의 빠른 변화와 경쟁 구조로 인해 우울, 불안, 스트레스 등 정신 건강 문제가 꾸준히 증가하고 있으며 한국은 여전히 ‘자살률 1위(OECD 기준)’를 기록하고 있다.

□ OECD 국가 연령표준화 자살률 비교



* 자료: OECD.STAT, Health Status Data(2023. 9. 추출), 우리나라 최근 자료는 OECD 표준인구로 계산한 수치임

* OECD 평균은 자료 이용이 가능한 38개 국가의 가장 최근 자료를 이용하여 계산 (출처 - 보건복지부, 「2023년 자살사망통계 발표」 보도자료, 2024.10.4. p.5)

그러나 기존의 대면 심리상담 서비스는 시간, 비용, 사회적인 시선 등의 이유로 많은 이들이 가볍게 이용하기 어렵다는 단점이 있다. 본 프로젝트는 사용자 입력 텍스트와 생체 정

보를 분석하여 정서 상태를 파악하고 이를 바탕으로 상황에 맞는 정서 안정 콘텐츠를 추천하는 심리 지원 앱이다.

언제 어디서든 모바일 앱 형태의 익명 서비스로 사용 가능하며, 부담 없이 심리적 안정을 취할 수 있도록 돕는 것이 본 프로젝트의 목표이다.

다. 주요 기능 및 구현 요소

(1) 감정 분석 및 콘텐츠 제공

사용자의 입력 문장을 KcELECTRA-small-v2022 모델로 분석하여 감정 상태를 분류함.

분석 결과는 FastAPI 서버를 통해 실시간으로 처리되며, 감정 결과에 따라 명상, 호흡법, 음악, 걷기 등 심리 안정 콘텐츠를 추천함.

분석 이력은 NeonDB에 저장되어 추후 시각화 및 개인화 추천 기능에 활용됨.

사용자가 자살 충동·극심한 우울·자기 비하 등의 표현을 입력할 경우 긴급 상담 기관 정보(1393, 1577-0199 등)를 제공하도록 구성함.

*(추후 전문 상담사 연계 시스템 도입 예정임.)

(2) 대화형 챗봇 모듈

감정 분석 기능은 대화형 챗봇 모듈을 통해 사용자와의 상호작용 속에서 구현됨.

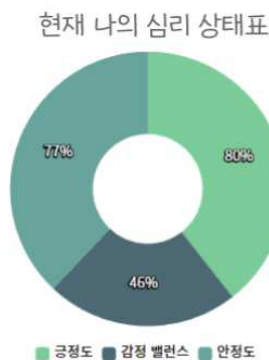
자연어 처리(NLP) 기반 분석 엔진을 적용하여 입력 문장을 문맥 단위로 분류하고,

감정 라벨(기쁨, 분노, 불안, 우울 등)을 식별함.

KcELECTRA-small-v2022 모델을 통해 감정을 판별하며, LLaMA 모델의 문맥 이해 구조를 참고하여 대화 응답 구조를 설계함.

감정 분석 결과를 기반으로 공감·위로·조언형 피드백을 제공하며, FastAPI 서버를 통해 연산 결과를 앱으로 전달함.

(3) 감정 변화 시각화



TIP 1

가까운 공원이나 집앞을 산책하면 감정을 다스리고 편안한 상태로 마인드 컨트롤을 할 수 있습니다.

TIP 2

기분 전환을 위해 좋아하는 음식을 섭취하거나 평소 즐겨하는 여가생활을 함께 해주시면 더욱 좋습니다.

감정 분석 및 챗봇 대화 결과를 NeonDB에 저장함.

저장된 데이터를 MPAndroidChart 라이브러리를 이용해 시각화함.

주간·월간 단위의 감정 변화를 그래프로 표현하여 사용자가 자신의 감정 패턴을 직관적으로 확인할 수 있도록 구성함.

시각화 결과는 앱 내 대시보드 형태로 제공됨.

(4) 하루일기 및 심리테스트

사용자가 하루의 감정과 생각을 기록할 수 있는 감정일기 기능을 제공함.

작성된 일기는 날짜별로 NeonDB에 저장되며, 리스트 형태로 조회 가능함.

간단한 심리테스트 기능을 통해 사용자가 자신의 정서 상태를 점검할 수 있도록 구성함.

테스트 결과는 감정 분석 데이터와 함께 통합 관리됨.

라. 개발 및 추진 방법

1) 팀 구성 및 역할 분담

성명	학번	주요 역할	담당 모듈	참여도(%)	대표 여부
원희	2302561	- 백엔드 개발 로그인, 사용자 DB 설계 감정 분석 모델 구축	- Firebase 기반 로그인, 사용자 정보 DB 설계 - 감정 분석 모델 개발	100	O
김빈나	2302535	- 프론트엔드 개발 안드로이드 UI/UX 구현 API 연동	- 안드로이드 개발 - 감정 분석 모델 개발 및 API 연동	100	
이예린	2302577	- 백엔드 개발 챗봇 기능 구현 및 연동 감정 추적 기능 구현	- 셀프 심리치료 기능 구현(챗봇 연계 및 감정 추적)	100	
정수연	2302590	- 프론트엔드 개발 안드로이드 UI/UX 개발 감정 분석 API 연동	- 안드로이드 개발 - 감정 분석 모델 API 연동	100	

윤서연	2302565	- 백엔드 개발 상담 내역 및 시각화 정보 DB 설계 챗봇 기능 구현	- Neon DB 기반 감정 분석, 시각화 데이터 설계 - 셀프 심리치료 기능 구현(챗봇 연계 및 감정 추적)	100	
-----	---------	--	---	-----	--

2) Man-Month

본 프로젝트에서는 1 Man-Month를 60시간(주 15시간 × 4주) 기준으로 산정하였으며, 이에 따라 2,700시간은 45 Man-Month로 환산된다.

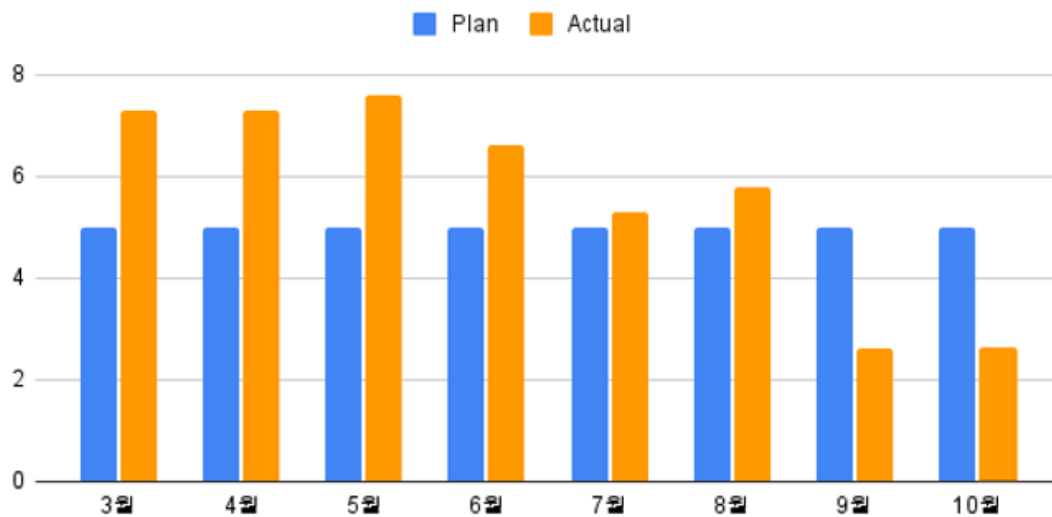
항목	내용
총 인원	5명
참여 기간	2025년 3월 ~ 11월 첫째 주 (총 36주)
주간 투입 시간 (1인 기준)	15시간
총 개발 시간	5명 × 15시간/주 × 36주 = 2,700시간
Man-Month 환산	2,700시간 ÷ 60시간 = 약 45 Man-Month

2) - 1. 월별 개발 분포

Month	실제 작업일 수	주요 활동
3월	22일	아이디어 회의 및 구상
4월	22일	기획
5월	23일	설계
6월	20일	개발 초기 단계
7월	23일	프론트엔드 위주 개발
8월	23일	백엔드 위주 개발
9월	10일(9/16 기준)	백엔드 위주 개발
10월	(기입 예정)	테스팅

2) - 2. Plan vs Actual 비교표

Month	계획 시간	계획한 일량 (Plan Man-Month)	실제 작업 시간	실제로 한 일량 (Actual Man-Month)
3월	300	5	440	7.3
4월	300	5	440	7.3
5월	300	5	460	7.6
6월	300	5	400	6.6
7월	300	5	320	5.3
8월	300	5	350	5.8
9월	300	5	160	2.6
10월	300	5	160	2.6



마. 기대효과

1) 대외적 효과

- 개인의 감정 데이터를 기반으로 한 맞춤형 정서 지원 실현
- 감정 변화에 따른 실시간 대응으로 심리적 부담 완화 및 자가 관리 유도
- 시간·장소 제약 없이 접근 가능한 비대면·익명형 심리 케어 서비스 제공
- 감정 시각화 기능을 통한 사용자 자가 인식 및 심리 회복력 향상
- 정신건강 관리 접근성 향상 및 사회적 심리 복지 기여
- 인공지능 기반 감정 분석 기술의 공공·교육·헬스케어 분야 확장 가능성 제시

2) 대내적 효과

- AI 감정 분석, 데이터베이스, 서버 통신, UI 설계 등 기술 실무 역량 강화
- 졸업작품을 실제 모바일 앱 형태로 구현하여 개발 실무 능력 향상
- 팀 단위 협업을 통한 개발 프로세스 이해 및 프로젝트 관리 능력 향상
- 공모전 출품 및 기술 발표를 통한 기술 경쟁력 검증 및 외부 피드백 확보
- 기술 실용화 경험을 바탕으로 향후 고도화·상용화 추진 가능성 확보
- AI 기반 감정 분석 시스템 구축 경험을 통해 심리 케어 분야 기술 응용 역량 축적

바. 사용 기술 소개

1) Fast API

가) 개요

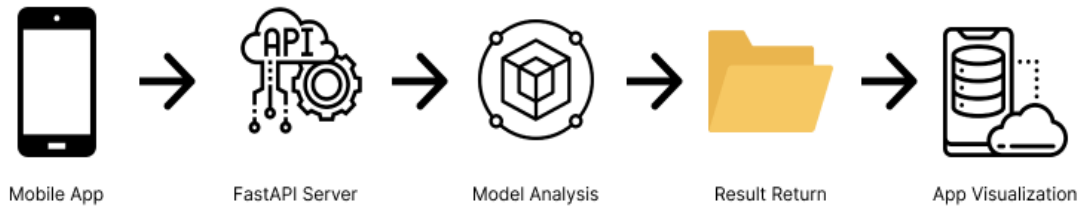
FastAPI는 Python 기반의 고성능 웹 프레임워크로, 서버통신, 데이터 처리, 보안 등의 복잡한 기능이 기본 제공되어 개발자는 핵심 로직만 작성하면 된다. 자동 API 문서 생성과 높은 처리량을 지원하도록 설계되었으며, 사용자로부터 전송된 데이터를 받아 처리한 뒤 분석 결과를 JSON형태로 반환하는 과정을 간단한 Python함수로 구현한다. 최근 AI 백엔드 개발 분야에서 활용도가 급격히 증가하고 있는 프레임워크이다.

본 프로젝트에서는 빠른 응답 처리와 안정적인 서버-DB 연동을 지원하기 위해 FastAPI를 도입하였다.

나) 사용 목적 및 역할

본 프로젝트에서는 FastAPI를 활용하여 감정 분석 모델(KoBERT, KcELECTRA-small-v2022)과 앱 간의 연동 API 서버를 구축하였다. 사용자가 모바일 앱에서 텍스트를 입력하면, 해당 텍스트는 FastAPI를 통해 분석 서버로 전달되고, 결과값(감정 라벨 및 확률

등)을 다시 앱으로 반환한다.



모바일 앱 → FastAPI 서버 → 모델 분석 → 결과 반환 → 앱 시각화

다) 동작 예시

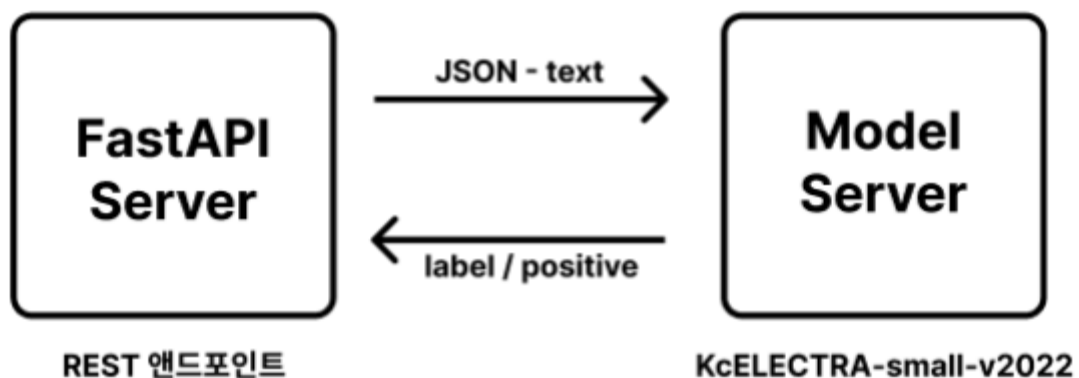
사용자가 안드로이드 앱에서 “오늘 너무 슬프다”라고 입력했을 때의 처리 과정:

1. 사용자 입력: 모바일 앱에서 “오늘 너무 슬프다” 텍스트 입력
2. 데이터 전송: 앱이 이 텍스트를 FastAPI 서버로 전송
3. 감정 분석: 서버에서 사전에 훈련된 모델을 통해 텍스트의 감정 상태 분석
4. 결과 처리: “슬픔” 감정으로 판단하고 결과 생성
5. 응답 반환: “힘든 하루였군요.”같은 맞춤형 메시지를 앱으로 전송
6. 화면 표시: 사용자에게 분석 결과와 응답 메시지 표시

라) 연동 방식

입력 방식: JSON 형태의 텍스트 데이터

응답 방식: 감정 라벨과 Softmax 확률 값 반환



API 구조: /predict, /health, /version 등 간단한 REST 엔드포인트 구성

마) 장점 및 특화 기능

항목	설명
비동기 지원	async def 기반으로 빠른 요청 처리 가능 (Uvicorn, Starlette 사용)
자동 문서화	Swagger UI 및 Redoc 지원 - API 테스트 UI 자동 생성
쉬운 연동성	Python 기반으로 ML/DL 모델(Pytorch, TensorFlow 등)과의 통합이 용이
타입 유추 및 검증	Pydantic을 활용한 입력값 타입 검증 및 에러 처리 자동화
경량 서버	별도 WAS 없이도 Uvicorn을 통해 실행 가능, 클라우드 배포 간편

*Uvicorn: Python용 비동기 웹 서버

*Starlette: Python에서 비동기 웹 애플리케이션을 구축할 수 있는 프레임워크

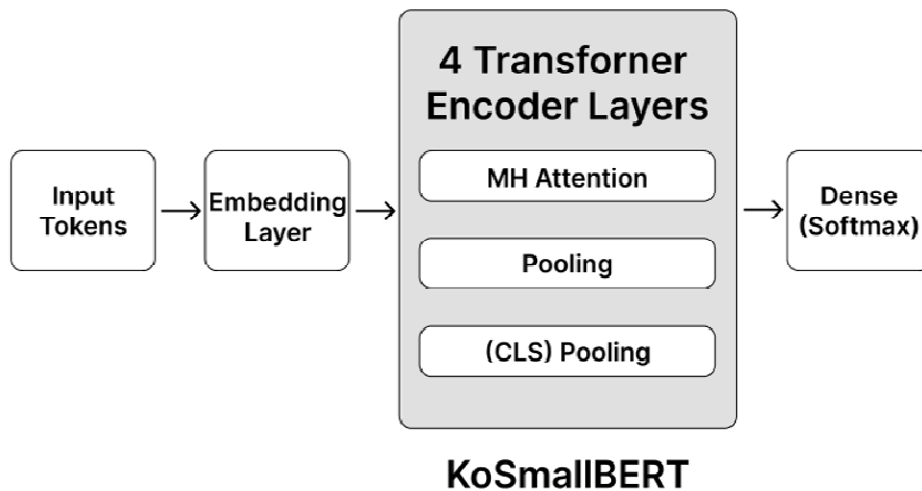
*Pydantic: Python에서 가장 널리 사용되는 데이터 유효성 검증 라이브러리

2) KoSmallBERT

가) KoSmallBERT 개요

KoSmallBERT는 한국어에 특화된 경량화 BERT 계열 모델로, 기존 KoBERT나 KorBERT보다 훨씬 적은 파라미터 수로 구성되어 있으며, 모바일 및 저사양 환경에서의 추론 속도와 효율성에 초점을 맞춘 모델이다.

해당 모델은 Pretrained 한국어 BERT 구조를 기반으로 구조를 단순화하고, 연산량을 줄임으로써 모바일이나 Web 환경 등에서의 실시간 감정 분석에 적합하다.



나) 주요 특성 및 장점

특성	설명
경량화 구조	Transformer 레이어 수 축소 (예: 6층 이하), Hidden size 감소
추론 속도 향상	기존 KoBERT 대비 응답 속도가 2~3배 이상 빠름
모바일 환경 최적화	TensorFlow Lite 변환 및 on-device 추론 가능성 높음
한국어 학습 데이터 기반	한국어 뉴스, 위키, 질문답변 데이터로 사전학습 수행

다) 본 프로젝트에서의 활용

KoSmallBERT는 비교 실험용 보조 모델로 도입되었으며, KcELECTRA-small-v2022와 함께 감정 분류 모델의 경량화 테스트에 사용되었다. 주요 목적은 다음과 같다.

- 동일한 감정 데이터셋으로 학습하여 정확도 및 추론 속도 비교
- 모바일 환경에서 실시간 감정 분석 가능 여부 테스트
- 추론 서버 또는 앱 내 임베딩을 고려한 최적화 평가

라) 한계점 및 미선정 이유

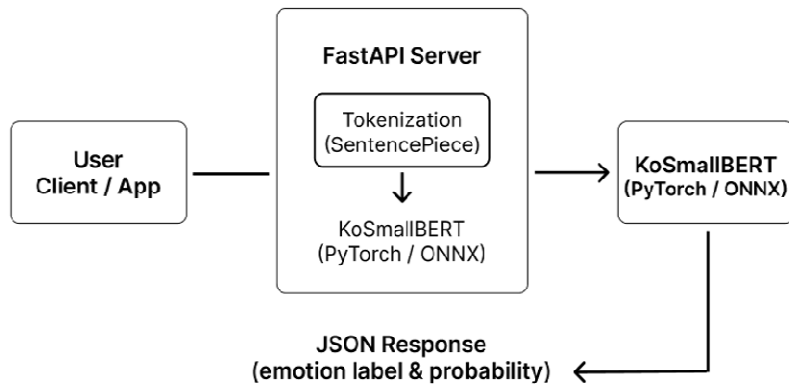
KoSmallBERT는 경량화된 구조로 모바일 환경에 적합하다는 장점이 있으나, 본 프로젝트에서는 다음과 같은 이유로 최종 모델로 선정하지 않았다.

- 정확도 측면 한계: 감정 분류 정확도가 다른 최신 경량 모델(KcELECTRA-small+2022)에 비해 낮게 나타남.

따라서 최종 선정 모델은 더 높은 정확도와 효율성을 제공한 KcELECTRA-small+2022로 결정하였다.

마) 사용 방식 및 연동 구조

- Pytorch 기반 또는 ONNX 모델로 학습 및 저장 후 FastAPI 서버에 연동
- REST API로 사용자 입력 문장 수신 → 토큰나이징 → 감정 분류 → 결과 반환
- API 응답 속도는 평균 50 ~ 150ms 수준으로 측정됨 (테스트 환경에 따라 변동)



3) KcELECTRA-small-v2022를 왜 사용하는가?

가) KcELECTRA-small-v2022 개요 및 선택 배경

본 프로젝트에서는 감정 분석 모델로 KcELECTRA-small-v2022를 선택하였다.

KcELECTRA는 Beomi가 공개한 한국어 특화 ELECTRA 모델로, ELECTRA(Pre-training Text Encoders as Discriminators Rather Than Generators) 구조를 기반으로 한다. ELECTRA는 BERT가 마스크된 토큰을 예측하는 방식과 달리, Generator가 대체한 토큰을 Discriminator가 진짜/가짜로 판별하는 학습을 진행하기 때문에 더 적은 자원으로도 효율적이고 빠르게 학습할 수 있다는 장점이 있다.

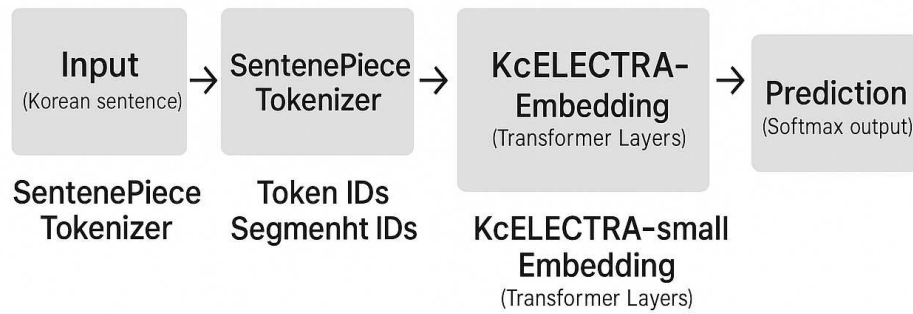
특히 KcELECTRA-small-v2022는 2022년 최신 버전으로, 한국어 위키백과, 뉴스, 나무위키, 청와대 국민청원 등 다양한 한국어 데이터셋으로 학습되어 일상적인 표현과 구어체 데이터 처리 성능이 개선되었다. 또한 small 모델 구조를 사용해 가볍고 연산 효율성이 뛰어나, 모바일 환경이나 제한된 자원에서도 활용하기 적합하다.

KoBERT, KorBERT, KoELECTRA, GPT 계열 모델을 후보군으로 검토했으나, 다음과 같은 이유로 KcELECTRA-small-v2022가 적합하다고 판단되었다.

- KoBERT: 한국어 문맥 이해도는 높지만, 모델 크기가 상대적으로 커 학습·추론 속도가 느리고 모바일 환경 적용에 비효율적이었다.
- KorBERT: 성능은 유사하나 커뮤니티 및 튜토리얼 자료가 적어 활용성이 낮았다.
- 기존 KoELECTRA: 속도는 빠르지만, 일부 감정 분류 태스크에서 성능 편차와 과적합 문제가 관찰되었다.
- GPT 계열: 문장 생성에는 강점이 있으나 분류 태스크에는 비효율적이고, 연산 자원 요구량이 커 프로젝트 환경에 적합하지 않았다.

이에 따라 KcELECTRA-small-v2022는 경량성과 정확성의 균형, 한국어 데이터 최적화, 모바일 친화적 구조라는 장점 덕분에 본 프로젝트의 감정 분류 모델로 채택되었다.

- 사전학습 기반 : ELECTRA (Discriminator 중심 구조)
- 토큰나이저 : SentencePiece
- 라이브러리 지원 : HuggingFace Transformers



나) 주요 기능

기능/특성	설명
한국어 문맥 이해 최적화	한국어 문법 구조와 띄어쓰기 특성을 잘 반영한 SentencePiece 토큰나이저 사용
다목적 전이학습 지원	감정 분석, 문장 분류, 개체명 인식, 질의응답 등에 fine-tuning 가능
문장 단위 입력 지원	문장 길이(최대 512 토큰)에 맞춰 한국어 문장을 자연스럽게 처리
사전학습 가중치 제공	Pretrained 모델을 바로 불러와 실습 가능 (Hugging Face Hub 제공)

다) 본 프로젝트에서의 사용 방식

KcELECTRA-small-v2022는 감정 분류 모델로 사용되며, 절차는 다음과 같다.

① 입력 전처리

- 사용자 입력 문장을 SentencePiece 토큰나이저로 토큰화
- 예 : 예시: "오늘 너무 속상해" → [CLS] 오늘, 너무, 속상해 [SEP]

② 모델 입력 및 추론

- 토큰화된 입력을 KcELECTRA-small-v2022 모델에 입력
- 마지막 레이어의 출력 벡터를 활용
- Softmax를 통해 9가지 감정 중 확률이 가장 높은 클래스를 예측

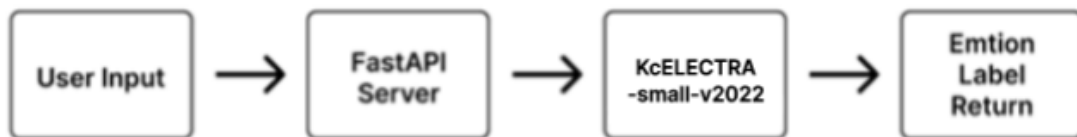
③ 결과 반환

- 예측된 감정 라벨을 JSON 형식으로 변환
- FastAPI 서버를 통해 응답



라) 연동 구조

KcELECTRA-small-v2022는 FastAPI 서버에 통합되어 모델 추론 역할을 수행한다.



[사용자 입력] → [FastAPI 서버] → [KcELECTRA-small-v2022 감정 분석 모델] → [감정 라벨 반환]

- 모델은 PyTorch 기반으로 학습 및 저장되며, .pt 또는 .bin 형식으로 FastAPI 서버와 함께 배포
- 요청은 REST API /predict endpoint로 처리되며, 응답 속도는 평균 100 ~ 300ms

마) 장점 및 도입 이유

장점	설명
한국어 성능 우수	감정 표현이 섬세한 한국어 문맥 이해와 파악에 효과적
Fine-tuning 용이	Pretrained 모델 활용 후 소량 데이터로도 감정 분류에 적용 가능
활용 사례 다양	뉴스 분류, 챗봇, 감정 분석 등 실전 적용 사례 다수
HuggingFace 지원 가능	Transformers 라이브러리를 통한 손쉬운 커스터마이징 및 배포

바) 비교 요약

항목	KoSmallBERT	KcELECTRA-small-v2022
정확도	중간 수준	높음
속도	빠름	빠름 (추론 효율 우수)
모델 크기	작다	작다 (경량화 최적화)
배포 환경	모바일/웹 환경 적합	서버 및 모바일 환경 모두 활용 가능 (HuggingFace 지원)

4) Uvicorn

가) Uvicorn: 경량 고성능 ASGI 서버 선택 이유 및 배포 시 장점

Uvicorn(유비콘, UltraViolet CORN)은 ASGI(Asynchronous Server Gateway Interface) 기반의 초경량, 고성능 웹 서버로, Python 환경에서 비동기 요청을 효율적으로 처리하는데 최적화되어 있다. 본 프로젝트에서는 AI 기반 심리 케어 서비스의 API 서버를 운영하기 위해 Python의 비동기 웹 프레임워크인 FastAPI와 함께 Uvicorn을 선택하였으며, 그 이유는 다음과 같다.

- ▶ ASGI(Asynchronous Server Gateway Interface) 지원: Uvicorn은 ASGI를 완벽하게 지원하는 서버이다. 이는 비동기(asynchronous) I/O 작업을 효율적으로 처리할 수 있게 하여, 다수의 동시 접속 요청에도 불구하고 높은 처리량과 낮은 지연 시간을 유지할 수 있도록 한다. AI 모델 추론과 같은 시간이 소요되는 작업이 많은 본 서비스의 특성상, 비동기 처리는 사용자 경험을 크게 향상시키는 중요한 요소이다.
- ▶ 경량 및 고성능: Uvicorn은 가벼우면서도 매우 빠른 성능을 자랑한다. 이는 Starlette (FastAPI의 기반) 및 uvloop (asyncio 이벤트 루프)와 같은 최신 기술을 활용하여 구현된 덕분이다. 적은 리소스로도 높은 성능을 발휘하므로, 서버 운영 비용을 절감하면서도 서비스의 응답성을 최적화할 수 있다.
- ▶ 단순한 설정 및 배포: Uvicorn은 설정이 매우 간편하여 개발 및 배포 과정의 복잡성을 줄여준다. `uvicorn main:app --reload`와 같은 간단한 명령어로 개발 서버를 실행할 수 있으며, 프로덕션 환경에서도 Docker 컨테이너 등과 결합하여 쉽게 배포할 수 있다. 이는 개발 및 운영 효율성을 높이는 데 기여한다.
- ▶ FastAPI와의 최적의 조합: FastAPI는 Uvicorn 위에서 최고의 성능을 발휘하도록 설계

된 프레임워크이다. FastAPI의 Pydantic 기반 유효성 검사, 자동 문서화 (Swagger UI/ReDoc), 그리고 비동기 처리 기능을 Uvicorn이 효과적으로 실행함으로써, 견고하고 확장 가능한 API 서버를 구축할 수 있다.

Uvicorn은 프로덕션 환경에서의 배포에 있어 다음과 같은 장점을 제공한다.

- ▶ 높은 처리량 및 낮은 지연 시간: 비동기 처리를 통해 수많은 동시 사용자 요청을 효율적으로 처리하여, 트래픽이 많은 상황에서도 안정적인 서비스를 제공한다. AI 모델 추론 대기 시간을 최소화하여 사용자 만족도를 높일 수 있다.
- ▶ 리소스 효율성: 가벼운 설계 덕분에 필요한 서버 자원이 적어 클라우드 환경에서 비용 효율적인 운영이 가능하다.
- ▶ 확장성: 부하 분산기(Load Balancer)와 연동하여 Uvicorn 인스턴스를 여러 개 띄우는 방식으로 서비스 트래픽 증가에 유연하게 대응할 수 있다.
- ▶ 안정적인 성능: uvloop와 httptools를 사용하여 C 언어로 구현된 파서와 이벤트 루프를 활용함으로써, 순수 Python 기반 서버에 비해 더욱 안정적이고 예측 가능한 성능을 제공한다.

이러한 장점들을 고려하여 Uvicorn은 'AnOn' 프로젝트의 백엔드 API 서버가 높은 성능과 안정성을 확보하고 효율적으로 운영될 수 있도록 하는 핵심적인 구성 요소로 결정되었다.

5) LLaMA

가) LLaMA 개요

▶ LLaMA 모델

LLaMA는 Meta에서 개발한 대규모 언어 모델로, 앱이나 웹 서비스에서 텍스트 기반 감정 분석 및 대화 기능에 활용 가능

▶ LLaMA 모델 사용 목적

사용자가 입력한 문장을 분석하여 감정 상태를 추론하고, 그에 적절한 심리적 안정 메시지 또는 상담 리소스를 제공

▶ 기능 요약

사용자의 감정을 입력받고, AI가 감정을 분석하여 감정 상태를 판단

분석 결과를 바탕으로 적절한 정서적 피드백, 콘텐츠, 위기 지원 서비스를 제공
LLaMA 모델을 통해 감정 기반 자연어 대화를 구현

▶ LLaMA 모델 활용 목적

정서 기반 문맥 이해

공감 기반 응답 생성

감정에 따른 맞춤형 대화 제공

▶ LLaMA 선택 이유

연구 목적으로 공개된 오픈소스 모델로, 비용 절감에 유리

모델 구조와 학습 방식, 파라미터 구성 등을 직접 분석할 수 있어 학습 측면에서 유리

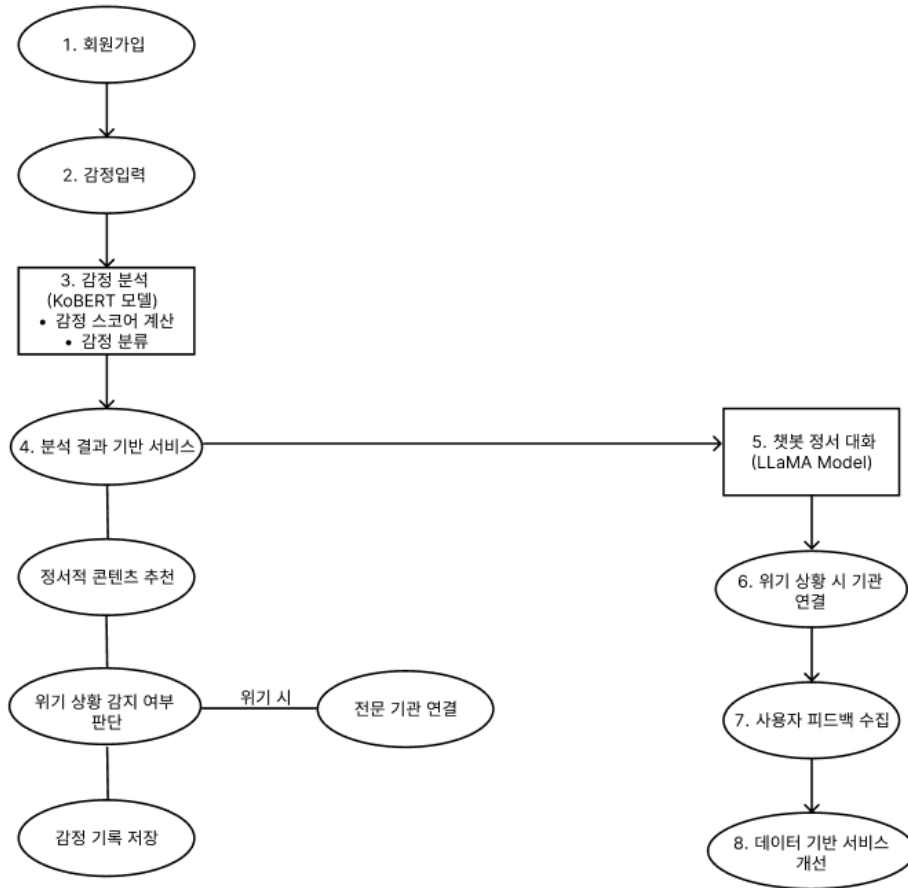
Fine-tuning, 프롬프트 최적화 등의 실험적 접근 시도 가능

FastAPI 기반 서버 구조와 쉽게 통합 가능

별도의 API 호출 없이 로컬로 올려 사용할 수 있어 데이터 보안 측면에서 유리

로컬 환경에서의 추론 속도 안정적

나) LLaMA 활용 챗봇 흐름도/순서도



1. 회원가입

사용자는 앱에 회원가입을 하고, 프로필 및 기본 정보를 입력

2. 감정 입력 (앱을 통한 사용자 입력)

사용자는 텍스트로 자신의 감정이나 상황을 자유롭게 입력

3. 감정 분석 (KoBERT 모델 기반)

입력된 텍스트를 KoBERT 기반 감정 분석 모델이 처리
분석된 감정 데이터는 후속 서비스 처리에 활용

4. 분석 결과 기반 서비스

분석 결과에 따라 다음과 같은 서비스 제공

4-1. 정서적 콘텐츠 추천

감정에 맞는 글, 영상 등 맞춤 콘텐츠 제공

4-2. 위기 상황 감지 여부 판단

감정 스코어가 특정 임계값 이상이면 '위기 상황'으로 감지
위기 시 전문 상담기관 연결

4-3. 감정 기록 저장

사용자의 감정 입력, 분석 결과, 추천 콘텐츠 기록을 저장
추후 사용자의 감정 변화 추적 및 리포트에 활용

5. 정서적 챗봇 대화 (LLaLMA)

분석된 감정을 바탕으로 정서 공감형 챗봇이 대화 진행
자연어 기반 위로, 공감, 조언 제공

6. 위기 상황 시 전문 기관 연결

챗봇 대화 중 위기 표현 재확인 시, 상담 센터 연결

7. 사용자 피드백 수집

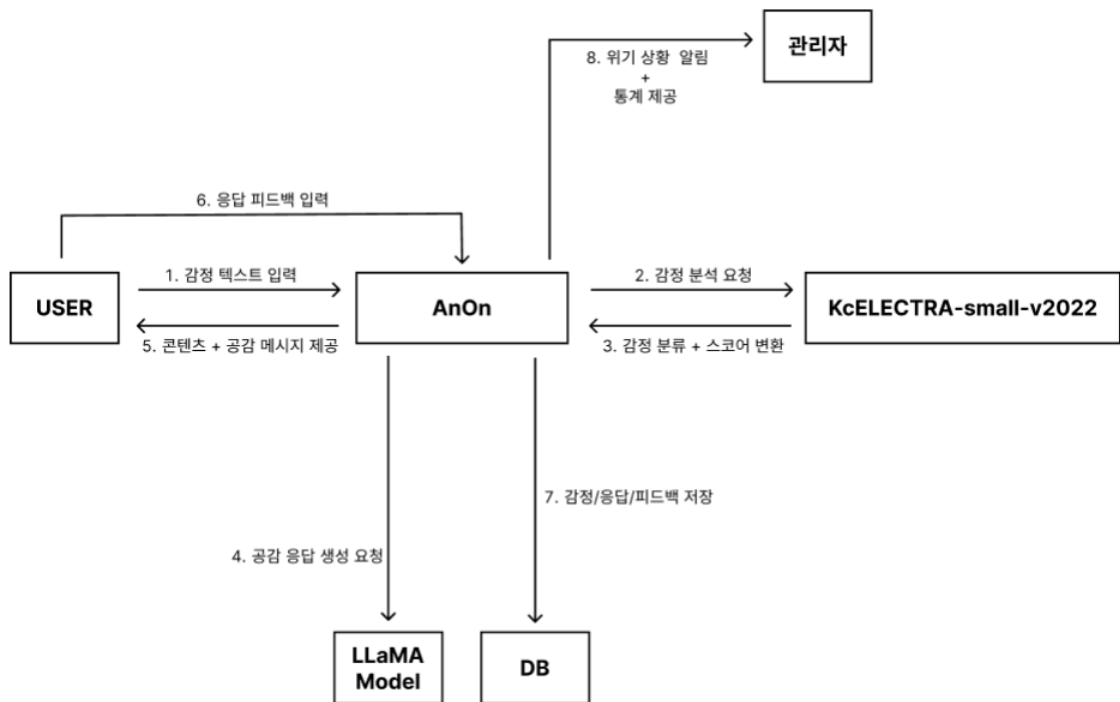
대화 종료 후 간단한 설문 제공 (예: '도움이 되었나요?')
사용자는 대화 만족도 및 개선 요청을 남길 수 있음

8. 데이터 기반 서비스 개선

수집된 감정 데이터, 피드백, 위기 상황 등을 기반으로 감정 분석 정확도 개선,
챗봇 응답 품질 향상, 콘텐츠 추천 최적화

9. 최종 응답 및 사용자 피드백

사용자는 분석 결과, 챗봇 대화, 콘텐츠 추천 등을 수신
일상적인 감정 관리 및 치유 서비스 지속 제공



2. 요구사항 명세서

가. 감정 분석 및 심리지원 정보 제공

번호	요구사항	세부 설명
UR-S01	대화 시작	사용자는 자연어 입력, 음성 대화를 통해 정서 기반 대화를 수행할 수 있어야 한다.
UR-S02	감정 인식	시스템은 사용자의 텍스트, 음성, 시각 신호를 분석하여 감정 상태를 인식해야 한다.
UR-S03	감정 판단	시스템은 서로 다른 감정 정보를 종합하여 하나의 감정 상태로 판단할 수 있어야 한다.
UR-S04	실시간 대화 처리	시스템은 감정 분석 결과를 1~2초 내에 화면에 반영하여 실시간 피드백을 제공해야 한다.
UR-S05	감정 데이터 저장 및 연계	분석된 감정 결과는 자동으로 저장되며, 시스템은 이를 콘텐츠 추천 기록·대시보드와 연계해야 한다.

나. 감정 변화 추적 및 시각화

번호	요구사항	세부 설명
UR-V01	감정 기록 저장	시스템은 감정 분석 결과를 시간 정보와 함께 누적 저장해야 한다.
UR-V02	감정 이력 조회	사용자는 저장된 감정 이력을 일간, 주간, 월간 단위로 조회할 수 있어야 한다.
UR-V03	감정 추세 분석	시스템은 감정 데이터의 변화 흐름을 분석하여 사용자에게 제공해야 한다.
UR-V04	시각화 컴포넌트 표시	사용자는 감정 데이터를 선, 막대 차트 등의 형태로 확인할 수 있어야 한다.
UR-V05	감정 대시보드 제공	사용자는 시각화된 감정 데이터를 대시보드 형태로 한눈에 확인할 수 있어야 한다.
UR-V06	피드백 응답	사용자는 감정 흐름에 따라 간단한 해석 또는 자기관리 조언을 제공받아야 한다.

다. 상황 맞춤 심리 안정 콘텐츠 제공

번호	요구사항	세부 설명
UR-C01	분석 기반 콘텐츠 제공	사용자는 감정 분석 결과에 따라 개인 맞춤형 심리 안정 콘텐츠를 제공받을 수 있어야 한다.
UR-C02	추천 콘텐츠 매칭	시스템은 사용자의 감정 상태에 적절한 콘텐츠를 자동으로 추천해야 한다.
UR-C03	콘텐츠 노출	사용자는 추천된 콘텐츠를 리스트/카드 형태로 확인할 수 있어야 한다.
UR-C04	콘텐츠 실행	사용자는 앱 내에서 콘텐츠를 바로 실행하거나 관련 페이지로 이동할 수 있어야 한다.
UR-C05	사용자 반응 반영	시스템은 사용자의 콘텐츠 선택 기록을 저장하고, 이를 기반으로 향후 추천을 개선해야 한다.

라. 회원 관리

번호	요구사항	세부 설명
UR-M01	회원가입	사용자는 소셜 로그인을 통해 계정을 생성할 수 있어야 한다.
UR-M02	로그인 및 인증	사용자는 등록된 소셜 계정으로 로그인하고 인증되어야 한다.
UR-M03	회원 정보 조회	사용자는 로그인 후 자신의 정보(닉네임, 이메일 등)를 확인할 수 있어야 한다.
UR-M04	회원 정보 수정	사용자는 자신의 정보(닉네임, 프로필 이미지)를 수정할 수 있어야 한다.
UR-M05	회원 탈퇴	사용자는 계정을 삭제하고, 관련 데이터를 제거할 수 있어야 한다.

3. 분석 활동

가. 회원 관리

1) 본인 인증

가) 목적

회원가입 또는 로그인 시, 타인의 정보 도용을 방지하고 사용자 식별을 명확히 하기 위해 본인 인증 절차를 도입한다. 인증을 통해 사용자 고유 식별이 가능해지며 개인화된 서비스 제공과 보안 강화를 도모한다.

나) 인증 방식

본 프로젝트의 시스템은 사용자의 실명, 전화번호 등을 별도로 수집하지 않는다. 대신 소셜 로그인(Google), 이메일 로그인을 통해 외부 인증 플랫폼이 사용자를 확인하고, 해당 정보를 받아 시스템에 연동한다.

▶ 이메일 로그인

- 사용자가 직접 입력한 이메일과 비밀번호를 기반으로 인증, 독립적인 로그인 기능
- 회원가입 시 등록한 정보를 기반으로 Firebase Authentication에서 일치 여부를 확인하며, 성공 시 고유 UID 발급
- 비밀번호는 Firebase에서 안전하게 해시 처리되어 저장되며, 로그인 시에도 동일한 방식으로 인증

▶ Google 로그인

- Google OAuth 2.0을 활용한 인증
- Google 계정 기반 고유 식별자와 프로필 정보를 통해 사용자 확인
- 전화번호, 이메일 별도 입력 불필요

다) 처리 흐름

- ① 사용자가 [이메일/구글 로그인] 버튼 클릭
- ② 이메일 및 비밀번호 입력/해당 플랫폼 로그인 페이지로 리다이렉트
- ③ 사용자가 로그인 및 정보 제공 동의
- ④ 인증 성공 시, 고유 식별자와 사용자 프로필 정보를 시스템으로 전달
- ⑤ 시스템은 이를 저장하거나 기존 사용자와 매칭하여 로그인 완료

2) OAuth

가) 사용 목적

사용자의 비밀번호를 직접 공유하지 않고도 외부 애플리케이션이 인증된 사용자 정보를 안전하게 활용할 수 있도록 허용하는 인증 방식이다. 본 시스템에서는 Google 로그인과 Firebase 이메일/비밀번호 로그인을 통해 사용자 인증을 수행한다.

나) 구성요소

구성 요소	예시
Resource Owner	Google 계정 사용자, Firebase 이메일 계정 사용자
Client	AnOn 앱
Authorization Server	Google OAuth 서버, Firebase Authentication 서버
Resource Server	Google API 서버, Firebase Authentication 서버
Access Token	Google 또는 Firebase에서 발급된 access token
Refresh Token	Google 또는 Firebase에서 발급 (장기 인증 유지용)

다) 처리 흐름

▶ 이메일/비밀번호 로그인

순서	처리 단계	상세 설명
1	로그인 요청	사용자가 이메일과 비밀번호를 입력하여 로그인 요청
2	Firebase 인증 요청	앱이 Firebase Auth에 사용자 정보 전달
3	인증 결과 응답	Firebase에서 입력 정보와 일치 여부 확인 후 UID 발급
4	회원 정보 조회 또는 등록	UID 기준으로 Firestore에서 사용자 정보 조회 또는 신규 등록
5	세션 생성 및 로그인 완료	로그인 성공 시 세션 생성 및 앱 내 로그인 처리 완료
6	로그인 실패 처리	인증 실패 시 오류 메시지 출력 및 로그인 화면 유지 (예: 비밀번호 오류 등)

▶ Google 로그인

순서	처리 단계	상세 설명
1	로그인 요청	AnOn 앱이 구글에 로그인 요청을 보냄
2	로그인 페이지 응답	구글 로그인 화면을 띄움
3	구글 계정 로그인	사용자가 구글 계정으로 로그인
4	Authorization Code 응답	구글이 인가 코드를 Redirect URI로 전달
5	Spring 서버로 로그인 요청	클라이언트가 인가 코드를 Spring 서버로 전달
6	Access Token 및 회원 정보 요청	Spring 서버가 구글에 Access Token과 사용자 정보를 요청
7	Access Token 발급 및 회원 정보 응답	Access Token과 회원 정보를 수신
8	회원 중복 확인 요청	DB에서 기존 사용자 존재 여부 확인

나. 감정 분석 및 심리지원 정보 제공

1) 사용 목적

사용자의 텍스트, 음성, 시각 입력을 기반으로 감정 상태를 인식하고 심리적 안정 및 조언을 제공한다. 위기 상황이 감지되면 긴급 연락망 및 전문 상담 기관과 연계될 수 있도록 한다.

2) 방식

가) KcELECTRA-small-v2022를 활용하여 감정 분류 수행

나) LLaMA 모델을 통해 맥락 기반 대화 및 공감형 응답 제공

다) 분석 결과를 1~2초 내 실시간으로 화면에 반영

다. 감정 변화 추적 및 시각화

1) 사용 목적

사용자의 감정 변화를 장기적으로 기록, 분석하여 자기 이해 및 자기관리를 돕는다.

2) 방식

가) 일/주/월 단위 감정 기록 누적

나) 그래프, 대시보드로 감정 추세를 시각화

다) 간단한 자기관리 피드백 제공

라. 상황 맞춤 심리 안정 콘텐츠 제공

1) 사용 목적

분석된 감정 상태에 따라 상황별로 적합한 심리 안정 콘텐츠를 제공한다.

2) 방식

가) 명상, 음악, 호흡법, 걷기 등 맞춤형 콘텐츠 추천

나) 앱 내 실행 및 외부 연동 지원

다) 사용자 반응 기록을 저장하여 추천 개선

마. 정서적 챗봇 대화 가능

1) 사용 목적

감정 기반 대화를 통해 공감과 정서적 안정을 제공한다.

2) 방식

가) 감정 분석 결과를 반영한 자연어 기반 대화

나) 공감, 위로, 조언 제공 및 위기 상황 시 전문 기관 연결

다) 대화 종료 후 만족도 설문과 피드백 수집

4. 기능 명세서

가. 회원 관리

번호	요구사항	세부 설명	관련 요구사항
F-M01	회원가입	신규 사용자가 소셜 인증을 시도하면 사용자 정보를 DB에 저장한다.	UR-M01
F-M02	로그인	사용자가 소셜 로그인을 시도하면 OAuth 인증 수행 후 성공 시 세션을 생성한다.	UR-M02
F-M03	마이페이지	로그인한 사용자의 정보를 조회하여 화면에 출력한다.	UR-M03
F-M04	회원 정보 수정	세션을 검증 성공 시 수정된 정보를 DB에 반영한다.	UR-M04
F-M05	회원 탈퇴	Authentication에 등록된 사용자 계정과 DB에 저장된 데이터를 삭제한다.	UR-M05

나. 감정 분석 및 심리지원 정보 제공

번호	요구사항	세부 설명	관련 요구사항
F-S01	텍스트 감정 분석 기능	KoSmallBERT를 통해 문장의 감정을 분류하고, LLAMA를 활용해 맥락 기반 감정 해석 수행	UR-S02
F-S02	음성 감정 분석 기능	음성에서 톤, 속도, 볼륨 등을 분석하여 감정 특성을 추출	UR-S02
F-S03	시각 신호 감정 분석 기능	MediaPipe 기반으로 눈 깜빡임, 동공 크기 등의 생체 정보를 카메라 입력을 통해 실시간 분석	UR-S02
F-S04	감정 통합 판단 기능	텍스트, 음성, 시각 정보를 종합하여 하나의 감정 상태로 도출	UR-S03
F-S05	실시간 분석 처리 기능	입력 또는 대화 흐름에 따라 1~2초 내 결과를 반영하고 화면에 출력	UR-S04
F-S06	감정 데이터 저장 및 연계	분석 결과를 히스토리 DB에 저장하고, 콘텐츠 추천·시각화 등과 연동	UR-S05

다. 감정 변화 추적 및 시각화

번호	요구사항	세부 설명	관련 요구사항
F-V01	감정 기록 저장 기능	감정 분석 결과를 시간 정보와 함께 DB에 저장	UR-V01
F-V02	감정 이력 조회 기능	일/주/월 단위의 감정 데이터를 조회하고 정리	UR-V02
F-V03	감정 추세 분석 기능	일정 기간 동안 감정 상태 변화 패턴 분석	UR-V03
F-V04	시각화 컴포넌트 표시	선 그래프, 막대 차트 등을 통해 분석 데이터를 시각화	UR-V04
F-V05	감정 대시보드 제공	긍정/부정 비율, 최신 감정 등 주요 지표 요약 제공	UR-V05
F-V06	사용자 피드백 제공	감정 흐름에 따른 간단한 해석과 자기관리 문구(예: 오늘은 산책이 어울려요)를 함께 제공	UR-V06

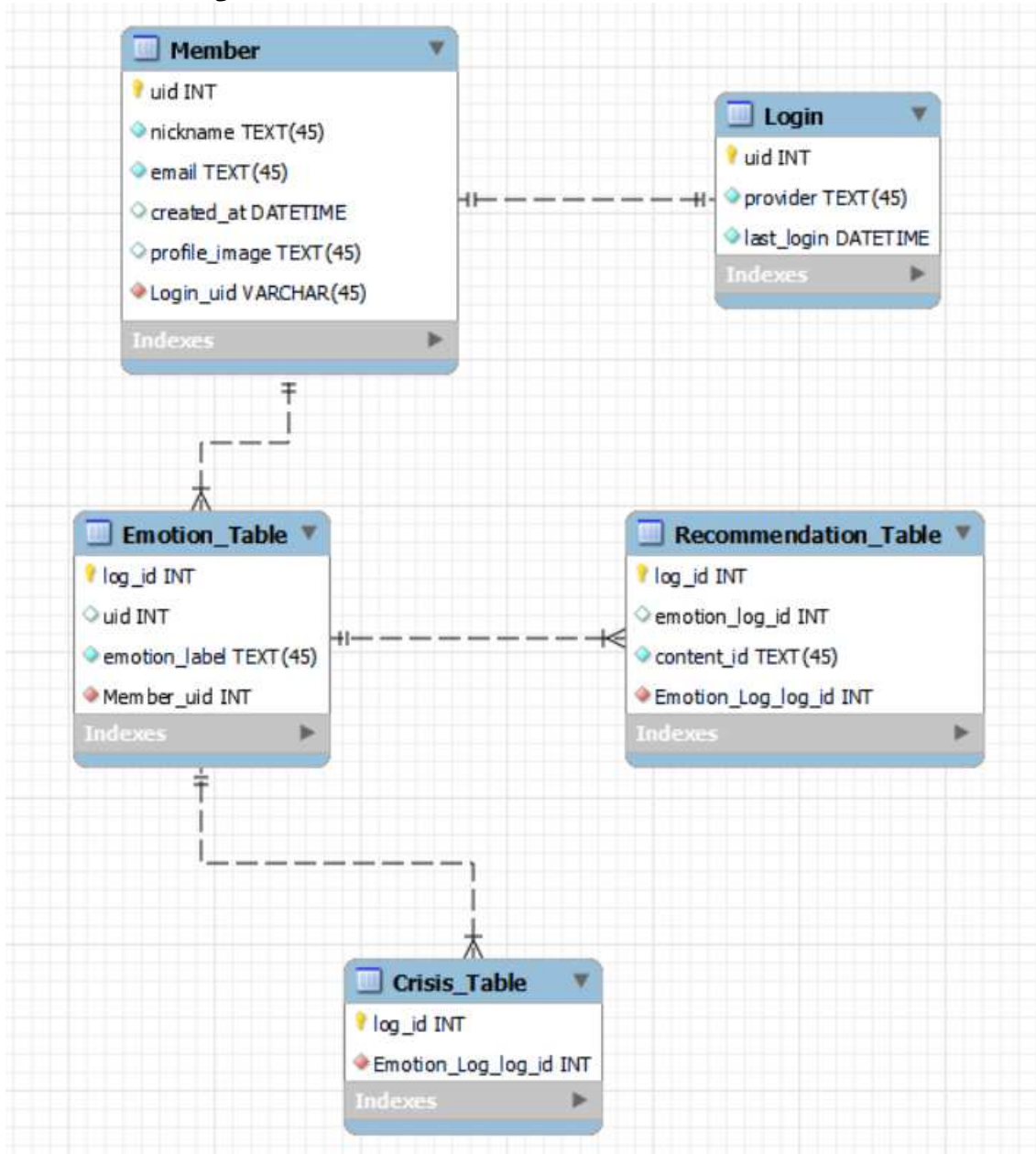
라. 상황 맞춤 심리 안정 콘텐츠 제공

번호	요구사항	세부 설명	관련 요구사항
F-C01	감정 입력 처리	감정 분석 결과를 기반으로 콘텐츠 추천을 시작	UR-C01
F-C02	콘텐츠 추천 실행	감정 유형과 매칭된 콘텐츠를 불러옴	UR-C02
F-C03	콘텐츠 목록 표시	추천 콘텐츠를 리스트/카드 형태로 화면에 표시	UR-C03
F-C04	콘텐츠 실행 기능	사용자가 선택한 콘텐츠를 앱 내에서 즉시 실행하거나 페이지로 이동	UR-C04
F-C05	사용자 반응 저장	사용자 선택 콘텐츠 정보를 기록해 이후 추천에 활용	UR-C05

5. 설계

가. Data 설계

1) ER Diagram



2) Table

가) Member

필드명	타입	제약 조건	설명
uid	int	Primary key	자동 생성되는 사용자 고유 ID
nickname	text	Not null	사용자 닉네임, 수정 가능
email	text	Unique, Not null	사용자 이메일, 중복 불가
created_at	datetime	Default current_timestamp	가입 시각 자동 저장
profile_image	text	Nullable	프로필 이미지 URL, 수정 가능

나) Login

필드명	타입	제약 조건	설명
uid	int	Primary key, Foreign key	Member.uid 참조
provider	text	Not null	로그인 방식 (google, 이메일 등)
last_login	datetime	Default current_timestamp	최근 로그인 시각 기록

다) Emotion_Log

필드명	타입	제약 조건	설명
log_id	int	Primary key	감정 기록 고유 ID
uid	text	Foreign key	Member.uid 참조
emotion_label	text	Not null	감정 결과 (ex. 기쁨, 불안 등)

라) Recommendation_Log

필드명	타입	제약 조건	설명
log_id	int	Primary key	추천 고유 ID
emotion_log_id	text	Foreign key	Emotion_Log.log_id 참조
content_id	text	Not null	추천 콘텐츠 식별자

마) Crisis_Log

필드명	타입	제약 조건	설명
log_id	int	Primary key, Foreign key	Emotion_Log.log_id 참조

3) CRUD SQL

가) CRUD SQL 요약표

엔티티	CREATE	READ	UPDATE	DELETE
Member	INSERT INTO member ...	SELECT * FROM member ...	UPDATE member SET ...	DELETE FROM member ...
Login	INSERT INTO login ...	SELECT * FROM login ...	UPDATE login SET ...	DELETE FROM login ...
Emotion_Log	INSERT INTO emotion_log ...	SELECT * FROM emotion_log ...	UPDATE emotion_log SET ...	DELETE FROM emotion_log ...
Recommendati on_Log	INSERT INTO recommendati on_log ...	SELECT * FROM recommendati on_log ...	UPDATE recommendati on_log SET ...	DELETE FROM recommendati on_log ...
Crisis_Log	INSERT INTO crisis_log ...	SELECT * FROM crisis_log ...	(없음)	DELETE FROM crisis_log ...

나) Member

-- CREATE

```
INSERT INTO member (uid, nickname, email, created_at, profile_image)
VALUES ('u123', 'anon_user', 'anon@email.com', CURRENT_TIMESTAMP,
'https://img.link');
```

-- READ

```
SELECT * FROM member WHERE uid = 'u123';
```

-- UPDATE

```
UPDATE member SET nickname = 'new_nick', profile_image = 'https://new.img'
WHERE uid = 'u123';
```

-- DELETE

```
DELETE FROM member WHERE uid = 'u123';
```

다) Login

```
-- CREATE
INSERT INTO login (uid, provider, last_login)
VALUES ('u123', 'google', CURRENT_TIMESTAMP);
```

```
-- READ
SELECT * FROM login WHERE uid = 'u123';
```

```
-- UPDATE
UPDATE login
SET last_login = CURRENT_TIMESTAMP
WHERE uid = 'u123';
```

```
-- DELETE
DELETE FROM login WHERE uid = 'u123';
```

라) Emotion_Log

```
-- CREATE
INSERT INTO emotion_log (log_id, uid, emotion_label)
VALUES ('log001', 'u123', '기쁨');
```

```
-- READ
SELECT * FROM emotion_log WHERE uid = 'u123';
```

```
-- UPDATE
UPDATE emotion_log
SET emotion_label = '불안'
WHERE log_id = 'log001';
```

```
-- DELETE
DELETE FROM emotion_log WHERE log_id = 'log001';
```

마) Recommendation_Log

```
-- CREATE
INSERT INTO recommendation_log (log_id, emotion_log_id, content_id)
VALUES ('rec001', 'log001', 'content123');

-- READ
SELECT * FROM recommendation_log WHERE emotion_log_id = 'log001';

-- UPDATE
UPDATE recommendation_log
SET content_id = 'content456'
WHERE log_id = 'rec001';

-- DELETE
DELETE FROM recommendation_log WHERE log_id = 'rec001';
```

바) Crisis_Log

```
-- CREATE
INSERT INTO crisis_log (log_id)
VALUES ('log001'); -- Emotion_Log.log_id를 FK로 가짐

-- READ
SELECT * FROM crisis_log WHERE log_id = 'log001';

-- UPDATE
-- 별도 속성이 없다면 UPDATE 필요 없음 (읽기/삭제 중심)

-- DELETE
DELETE FROM crisis_log WHERE log_id = 'log001';
```

4) JSON 데이터 예시: 감정 분석 요청 시 사용되는 데이터 형식

본 프로젝트의 핵심 기능인 AI 기반 심리 케어 서비스(AnOn)에서 감정 분석 요청 시 클라이언트(모바일 앱)는 사용자 입력 데이터를 서버의 감정 분석 모델에 전달하기 위해 JSON 형식의 데이터 구조를 사용함.

이 구조는 텍스트 입력을 기본으로 하며, 필요 시 음성 데이터를 Base64 형식으로 인코딩하여 함께 전송할 수 있도록 설계되었음.

다음은 감정 분석 요청에 사용되는 JSON 데이터의 예시임.

JSON

```
{
  "user_id": "user12345",
  "session_id": "session_abcde12345",
  "timestamp": "2025-06-10T10:30:00Z",
  "text_input": "오늘 하루가 너무 힘들고 우울해. 아무것도 하기 싫어.",
  "audio_input_base64": "data:audio/wav;base64,xxxxxxxxx..."
}
```

user_id는 요청을 보낸 사용자를 식별하는 고유 값으로, Firebase UID와 연동되어 관리됨. session_id는 앱 실행 시 생성되는 세션 식별자로, 한 세션 내에서 여러 감정 입력을 구분하는 데 사용됨.

timestamp는 요청이 생성된 시각을 ISO 8601 형식(UTC 기준)으로 기록함.

text_input은 사용자가 입력한 텍스트로, 감정 분석 모델(KcELECTRA-small-v2022)의 주요 입력값이 됨.

audio_input_base64는 음성 입력을 Base64 형식으로 인코딩한 문자열이며,

data:audio/wav;base64, 프리픽스를 포함함. 이 필드는 사용자가 음성으로 입력했을 경우에만 포함되며, 텍스트 입력만 있을 때는 생략됨.

이 JSON 구조는 요청 데이터의 확장성과 유연성을 고려하여 설계되었으며, 향후 멀티모달 감정 분석(텍스트 + 음성) 모델로 확장할 수 있는 기반을 제공함.

모든 데이터는 UTF-8 인코딩을 사용하며, FastAPI 서버에서 JSON Schema Validation을 통해 데이터의 무결성과 형식을 검증함.

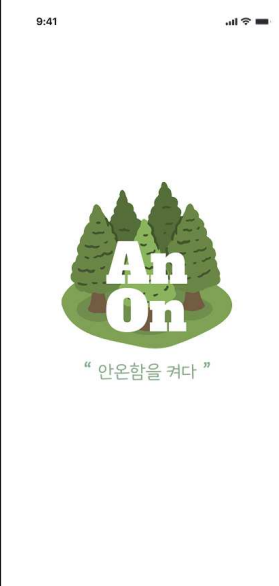
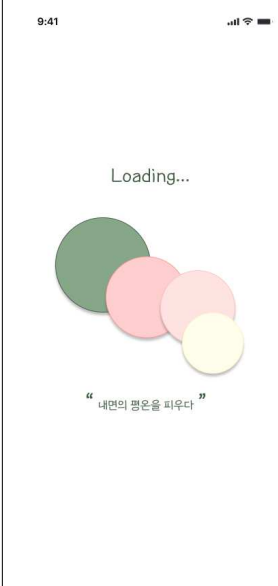


이 설계를 통해 클라이언트는 감정 분석 요청을 안정적으로 전송할 수 있고, 서버는 실시간으로 감정 분석 결과를 반환함으로써 AnOn의 핵심 기능인 AI 기반 실시간 감정 분석 서비스를 효율적으로 구현함.

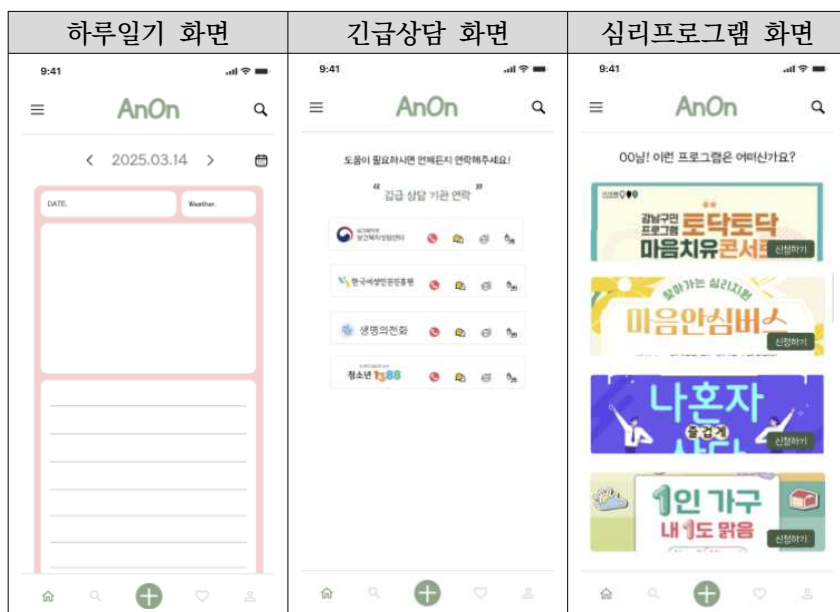
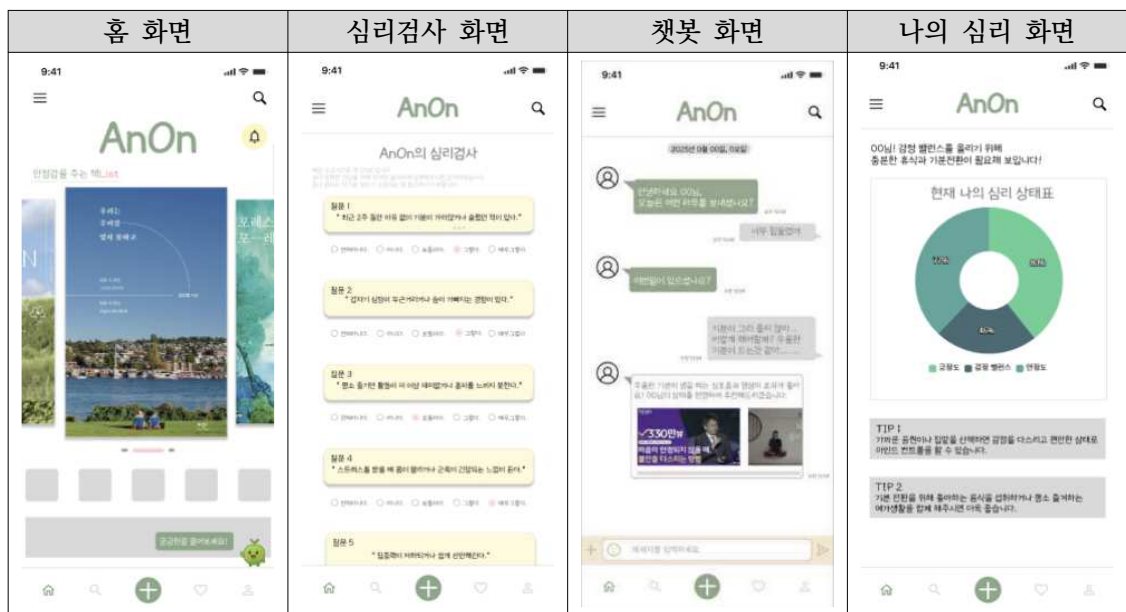
나. 화면 설계

1) 회원 관리

기능	화면 요소
회원가입 및 로그인	이메일/비밀번호, Google계정으로 시작하기 버튼
마이페이지	사용자 정보 표시, 상세 정보 버튼
마이페이지 상세	사용자 상세 정보 표시, 수정하기 버튼
회원 정보 수정	닉네임, 프로필 이미지 수정 폼, 변경 완료 버튼
회원 탈퇴	확인 팝업창, 탈퇴 요청 버튼

2) 화면 구성

진입 화면	로딩 화면	회원가입 화면	마이페이지
			

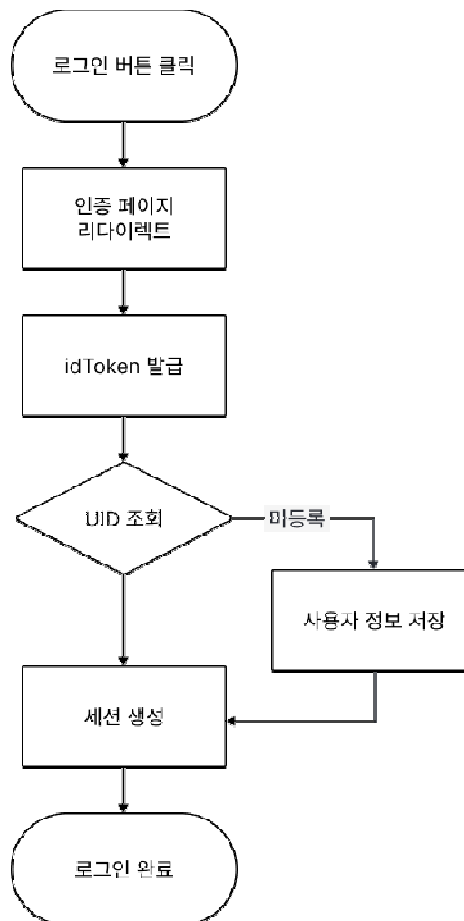


다. Logic 설계

1) 회원 관리

가) 로그인 및 신규 회원 등록

▶ 소셜(Google) 로그인



1. 로그인 버튼 클릭

: 앱 내 소셜 로그인(Google), 이메일 로그인 버튼을 클릭한다.

2. 인증 페이지 리다이렉트

: Firebase는 선택된 소셜 로그인 제공자의 인증 페이지로 사용자를 리다이렉트하며, 사용자는 OAuth 인증을 수행한다.

3. idToken 발급

: 인증 성공시 Firebase는 해당 사용자의 고유한 UID를 포함한 idToken을 발급한다.

4. UID 조회

: 앱은 idToken에서 UID를 추출하고, 해당 UID를 기준으로 Firestore에서 사용자 정보를 조회한다.

4-1. (신규) 사용자 정보 저장

: 미등록 UID인 경우, Firestore에 사용자 정보를 새로 저장한다. 이때 이메일, 닉네임 등의 초기 정보가 기록된다.

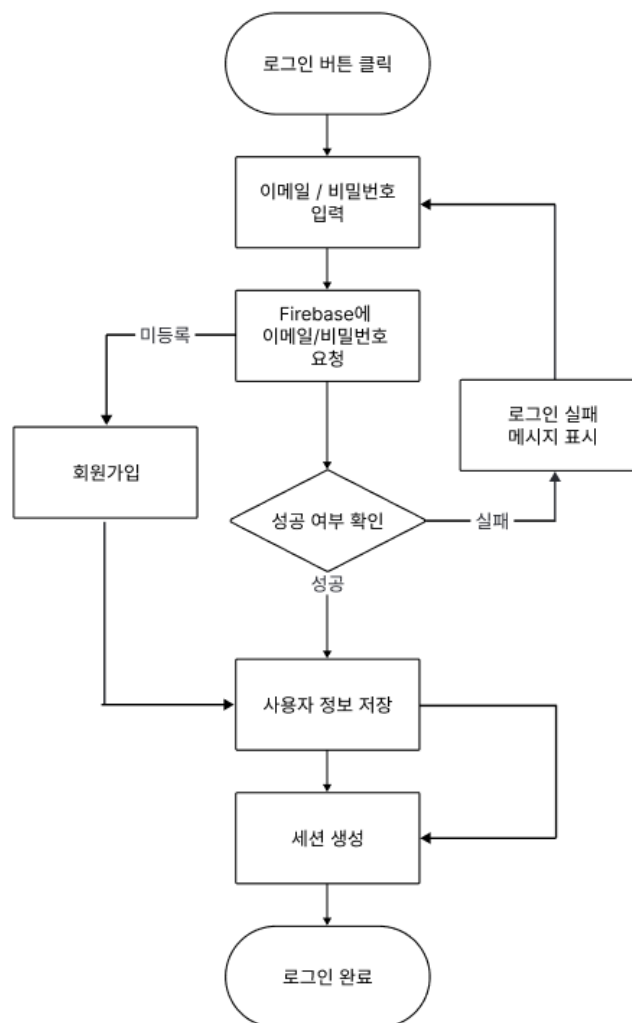
5. 세션 생성

: 인증된 사용자를 기준으로 세션을 생성하고, 사용자 환경을 초기화한다. 이후 사용자 정보는 지속적으로 접근 가능해진다.

6. 로그인 완료

: 세션이 성공적으로 설정되면, 사용자는 로그인 상태로 메인 화면에 진입하게 된다.

▶ 일반(이메일) 로그인



1. 로그인 버튼 클릭

: 사용자가 앱 내 로그인 화면에서 이메일/비밀번호 로그인 버튼을 클릭한다.

2. 이메일 / 비밀번호 입력

: 사용자는 로그인 입력창에 자신의 이메일과 비밀번호를 입력한다.

3. Firebase에 이메일 / 비밀번호 요청

: 입력된 정보를 바탕으로 Firebase Authentication에 로그인 요청을 전송한다.

이때 Firebase는 해당 계정의 존재 여부 및 비밀번호 일치 여부를 검증한다.

4. 성공 여부 확인

: Firebase로부터 인증 성공/실패 결과를 수신한다.

4-1. 로그인 실패

- 이메일이 존재하지 않거나 비밀번호가 일치하지 않을 경우 실패로 처리된다.
- 사용자에게 “로그인 실패 메시지”가 표시된다.
- 계정이 없는 경우라면 회원가입으로 유도할 수도 있다.

4-2. 미등록인 경우(회원가입 유도)

- 이메일이 Firebase에 등록되지 않은 경우, 사용자에게 회원가입을 안내한다.
- 사용자는 회원가입 페이지로 이동하여 새 계정을 생성할 수 있다.

5. 성공한 경우 -> 사용자 정보 저장

: 로그인에 성공하면 앱은 해당 사용자의 UID를 기준으로 추가 사용자 정보를 Firebase 등 DB에 저장하거나 이미 저장된 데이터를 조회한다.

신규 사용자인 경우 이메일, 닉네임 등 초기 정보를 저장한다.

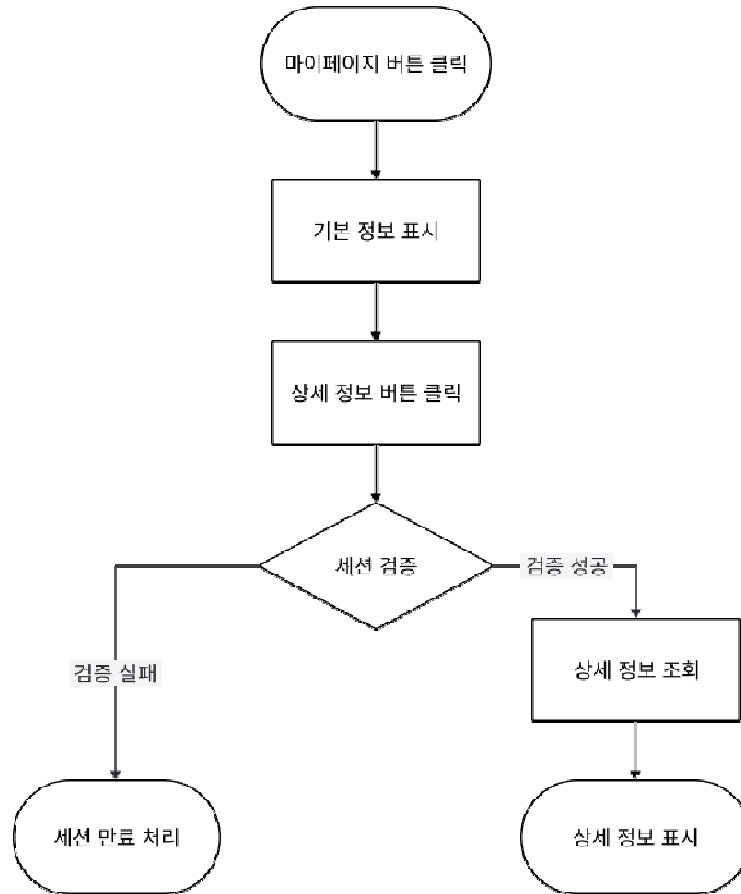
6. 세션 생성

: 인증된 사용자의 세션을 앱 내에서 생성한다. 이를 통해 이후 페이지에서도 로그인 상태를 유지하며 사용자 데이터를 활용할 수 있다.

7. 로그인 완료

: 세션 설정이 완료되면 사용자는 메인 화면 등 앱의 주요 기능에 접근할 수 있다.

나) 회원 정보 조회



1. 마이페이지 버튼 클릭

: 앱 내 '마이페이지' 화면에 진입한다.

2. 기본 정보 표시

: 기본적으로 프로필 이미지와 닉네임이 표시된다.

3. 상세 정보 버튼 클릭

: 마이페이지 내 상세 정보 버튼을 클릭한다.

4. 세션 검증

: 세션이 유효한지 확인하고, idToken에서 UID를 추출한다.

4-1. (실패) 세션 만료 처리

: 세션 만료로 정보 조회 실패 시 로그인 화면으로 리디렉션한다.

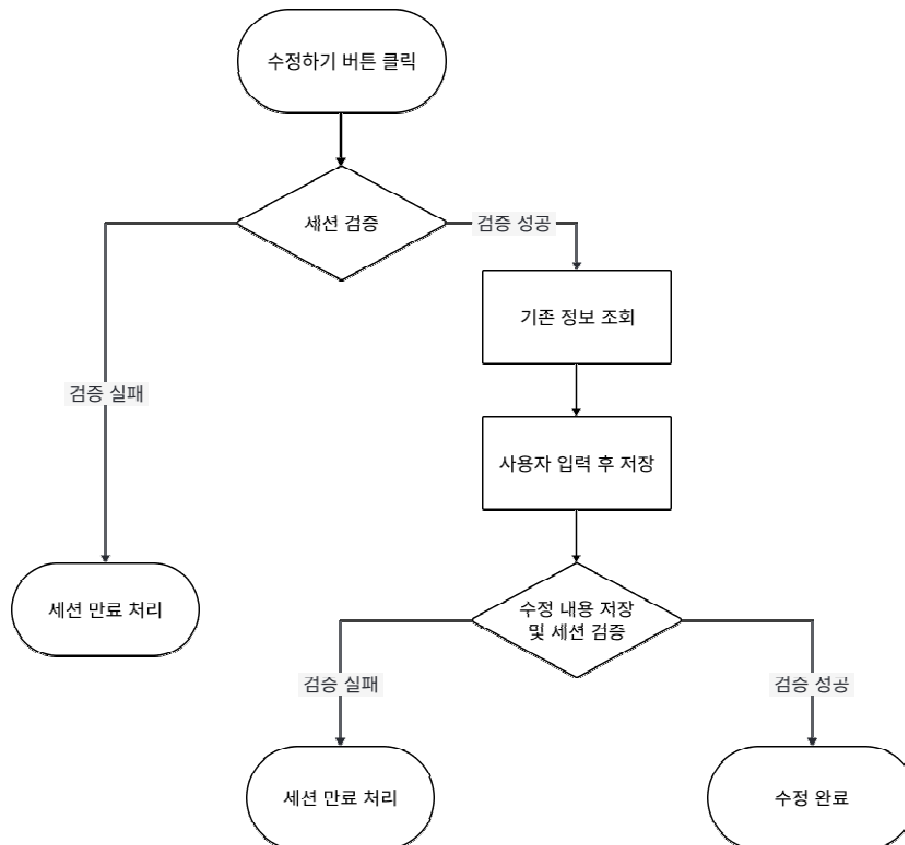
5. 상세 정보 조회

: UID에 해당하는 문서를 Firestore에서 읽어온다.

6. (성공) 상세 정보 표시

: UID, 이메일, 가입 일시 등 사용자 정보를 화면에 렌더링한다.

다) 회원 정보 수정



1. 수정하기 버튼 클릭

: 사용자가 마이페이지의 상세 정보 화면에서 수정하기 버튼을 클릭한다.

2. 세션 검증

: 세션이 유효한지 확인하고, idToken에서 UID를 추출한다.

2-1. (실패) 세션 만료 처리

: 세션 만료로 정보 조회 실패 시 로그인 화면으로 리디렉션한다.

3. 기존 정보 조회

: UID를 기준으로 Firestore에 저장된 기존 정보를 불러와 입력창에 반영한다.

4. 사용자 입력 후 저장

: 사용자가 수정할 항목을 입력하고 저장 버튼을 누른다. 이때 수정 가능한 정보는 프로필 이미지와 닉네임이다.

5. 수정 내용 저장 및 세션 검증

: UID를 기준으로 Firestore 문서를 업데이트한다. 저장 과정에서 세션 검증이 실시된다.

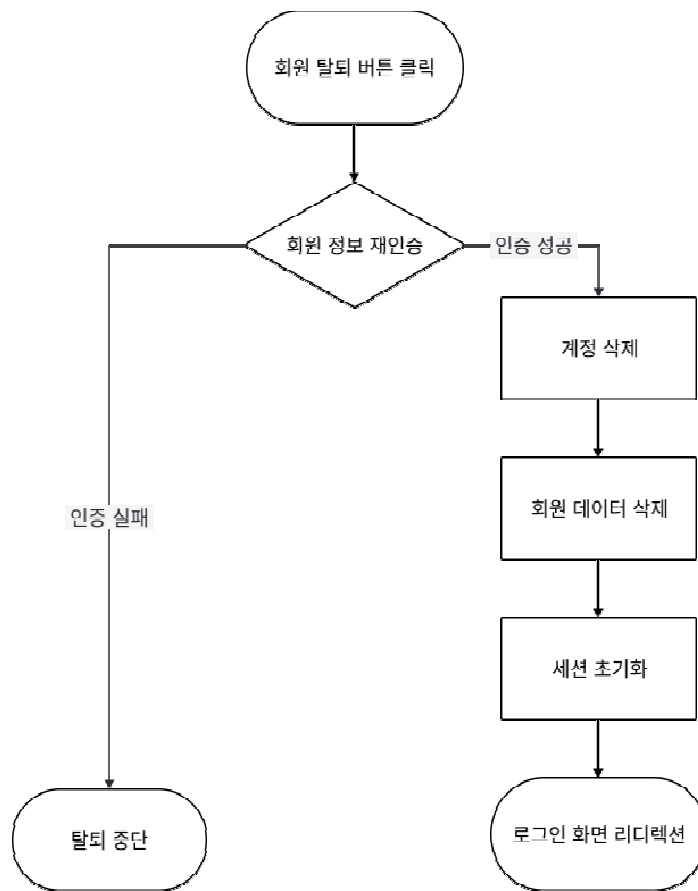
5-1. (실패) 세션 만료 처리

: 수정 실패 시 토스트 메시지를 출력하고, 마이페이지 화면으로 리디렉션된다.

5-2. (성공) 수정 완료

: 수정 성공 시 토스트 메시지를 출력하고, 마이페이지 상세 정보 화면으로 리디렉션된다.

라) 회원 탈퇴



1. 탈퇴 버튼 클릭

: 앱 내 회원 탈퇴 버튼을 클릭한다.

2. 인증 화면 리다이렉트

: Firebase는 인증 페이지로 사용자를 리다이렉트하고, 사용자는 OAuth 인증을 재수행한다.

3-1. (실패) 탈퇴 중단

: 인증에 실패하거나 사용자가 인증을 취소하면, 탈퇴 절차는 중단되며 메인 화면으로 되돌아간다.

3-2. (성공) 계정 삭제 -> 데이터 삭제 -> 세션 초기화

: 인증에 성공하면 Firebase Authentication에 등록된 사용자의 계정이 삭제되며, 해당 UID를 기준으로 Firestore에 저장된 사용자의 모든 데이터 또한 삭제된다. 이후 세션이 초기화되어 로그인 상태가 해제된다.

4. 로그인 화면 리디렉션

: 탈퇴 처리가 완료되면, 사용자는 로그인 화면으로 리디렉션된다.

2) 챗봇

가) 서버 구조

EC2-A(총괄 서버): FastAPI + LLaMA

EC2-B(보조 서버): FastAPI + KcELECTRA-small-v2022

▶ EC2

모델을 구동하기 위한 서버 인스턴스로, 역할에 따라 총괄 서버와 보조 서버로 분리되어 배치된다.

▶ FastAPI

Uvicorn ASGI 서버 위에서 실행되며, LLaMA 및 KcELECTRA-small-v2022 모델이 위치한 서버 간의 요청과 응답을 처리한다.

▶ KcELECTRA-small-v2022

총괄 서버로부터 사용자 입력을 전달받아 감정 분석을 수행하고, 그 결과를 반환한다. 가볍고 빠른 추론이 가능하여 실시간 감정 분류에 적합하다.

▶ LLaMA

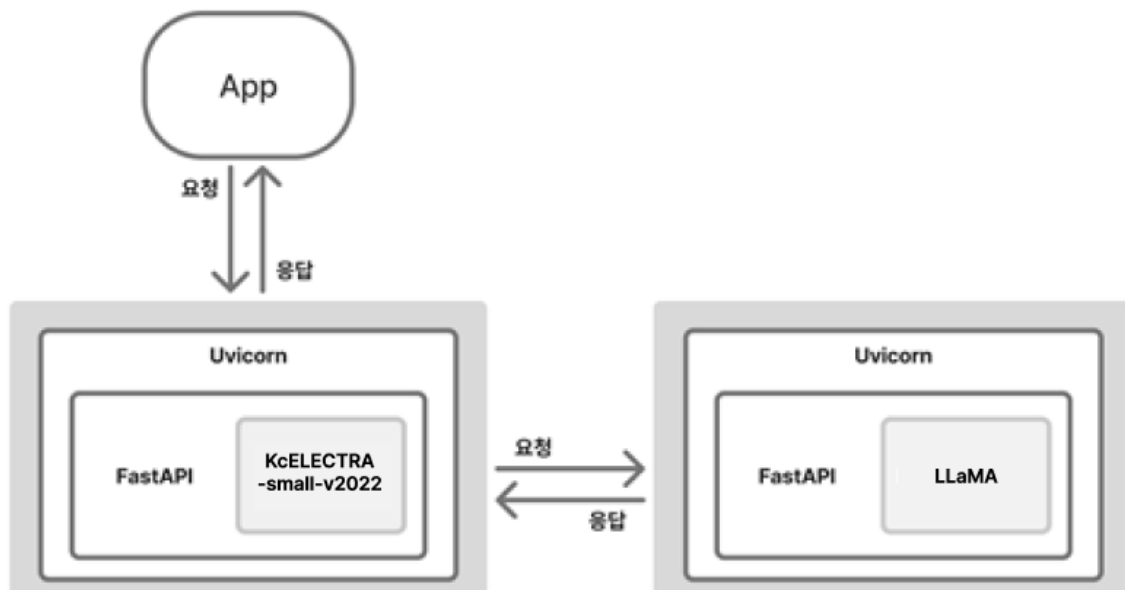
KcELECTRA-small-v2022가 반환한 감정 분석 결과와 사용자 입력 문장을 기반으로 자연어 대화 응답을 생성한다. 해당 모델은 외부 API를 호출하지 않고 EC2-A 서버 메모리에 직접 로드되어 로컬 환경에서 실행된다.

▶ EC2 서버 분리 이유

LLaMA 모델은 고성능 자연어 생성 모델로 연산량과 자원 소모가 크다. 이에 따라 감정 분석 모델(KcELECTRA-small-v2022)과 분리 배치하여 시스템 안정성과 성능을 확보하였다. 또한 서버 간 역할을 분리하면 유지보수와 확장이 용이하다.

HTTP 요청과 응답은 1:1 구조를 따르므로, EC2-A가 총괄 서버로서 전체 흐름을 조율하도록 설계되었다.

나) 요청 처리



1. 사용자 입력 전송 (App → Hugging Face API)

: 사용자가 앱 내 챗봇과 대화를 시작하면, 입력 문장은 HTTP 요청 형태로 Hugging Face Space 서버에 전달된다. Hugging Face Space는 Uvicorn 기반의 FastAPI 애플리케이션으로 구성되어 있으며, 요청을 수신하여 내부 처리 절차를 시작한다.

2. 감정 분석 수행 (Hugging Face Space 내부)

: Hugging Face Space는 FastAPI 내부에서 KcELECTRA-small-v2022 모델을 호출하여 입력 문장의 감정을 분석한다. 해당 모델은 Google Colab 환경에서 학습·검증된 뒤 Hugging Face에 배포된 상태이다.

3. 감정 분석 결과 반환 (Hugging Face API → App)

: 감정 분석 결과는 JSON 형태로 직렬화되어 FastAPI를 통해 앱 클라이언트로 응답된다.

4. 응답 생성 (앱 내부 처리 또는 LLaMA 연동)

: 앱은 수신한 감정 분석 결과와 원문 입력을 바탕으로, Hugging Face API에 배치된 LLaMA 모델을 호출하거나 내부 처리 로직을 통해 자연어 응답을 생성한다.

5. 최종 응답 출력 (App)

: 생성된 응답은 앱 화면에 출력되어 사용자에게 제공된다.

6. 보안 및 인증 처리 (App ↔ Hugging Face API)

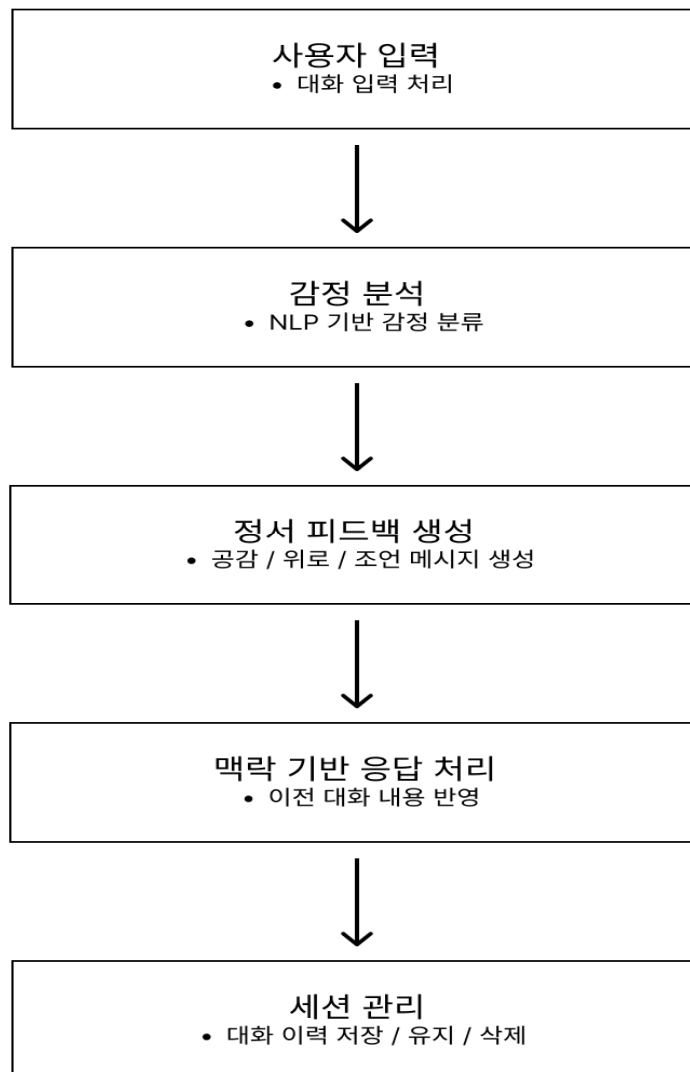
: 모든 요청과 응답은 HTTPS 프로토콜을 기반으로 암호화되어 전송된다. 또한 Firebase Authentication과 연동하여, 인증된 사용자만 Hugging Face API에 접근할 수 있도록 제한한다.

7. 성능 최적화 및 확장성 관리 (Colab ↔ Hugging Face Space)

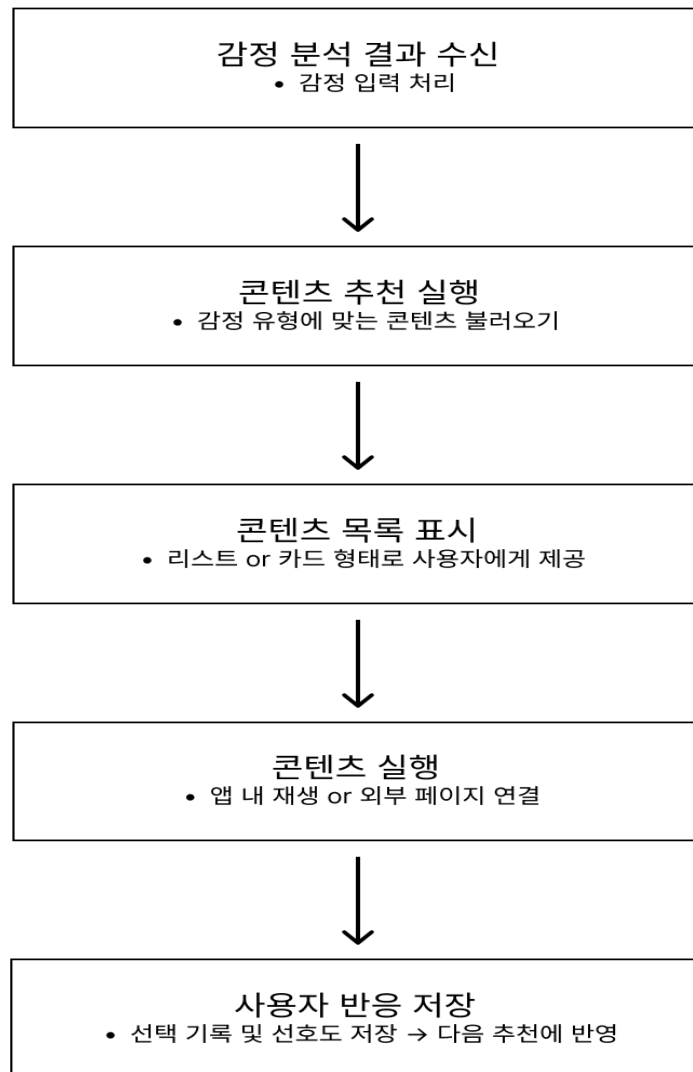
: Google Colab에서 주기적으로 모델을 재학습 및 검증하여 Hugging Face Space에 업데이트한다. Uvicorn + FastAPI의 비동기 구조를 통해 동시 요청을 효율적으로 처리하며, 필요 시 KoBERT, KoSmallBERT 등 다른 모델을 추가로 배포하여 확장할 수 있다.

다) 앱 내부 시스템

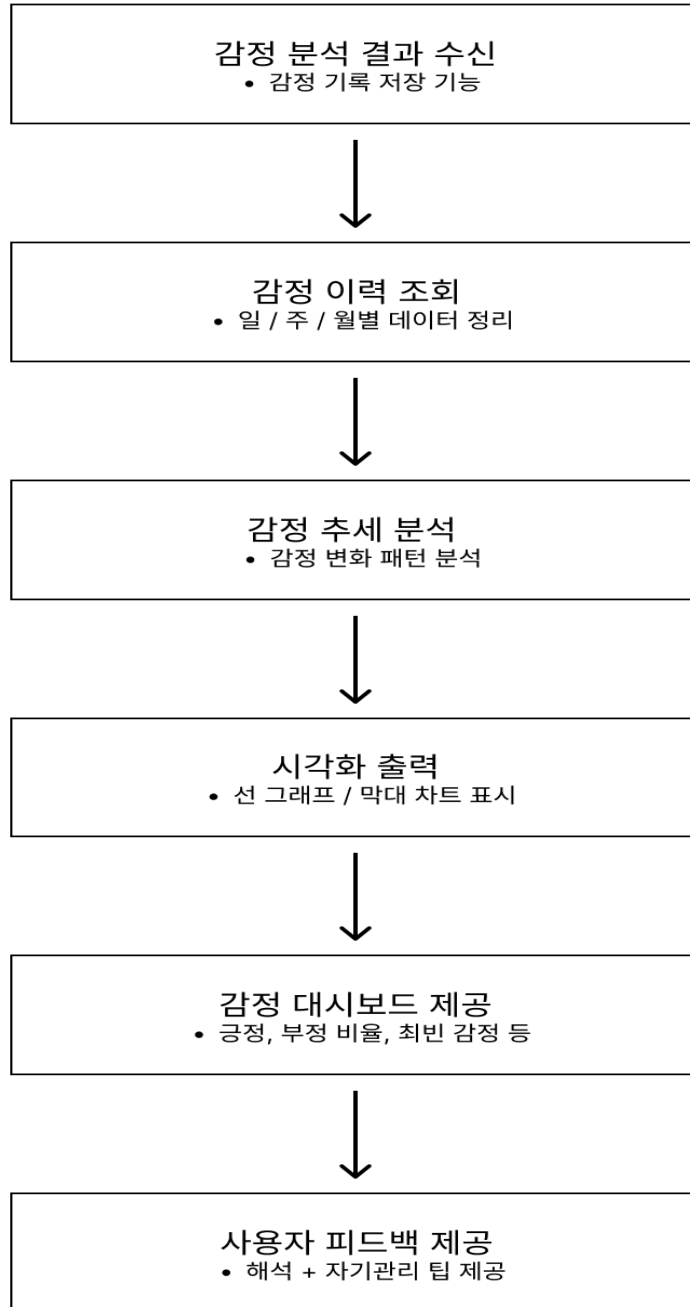
- 정서 피드백 대화 시스템



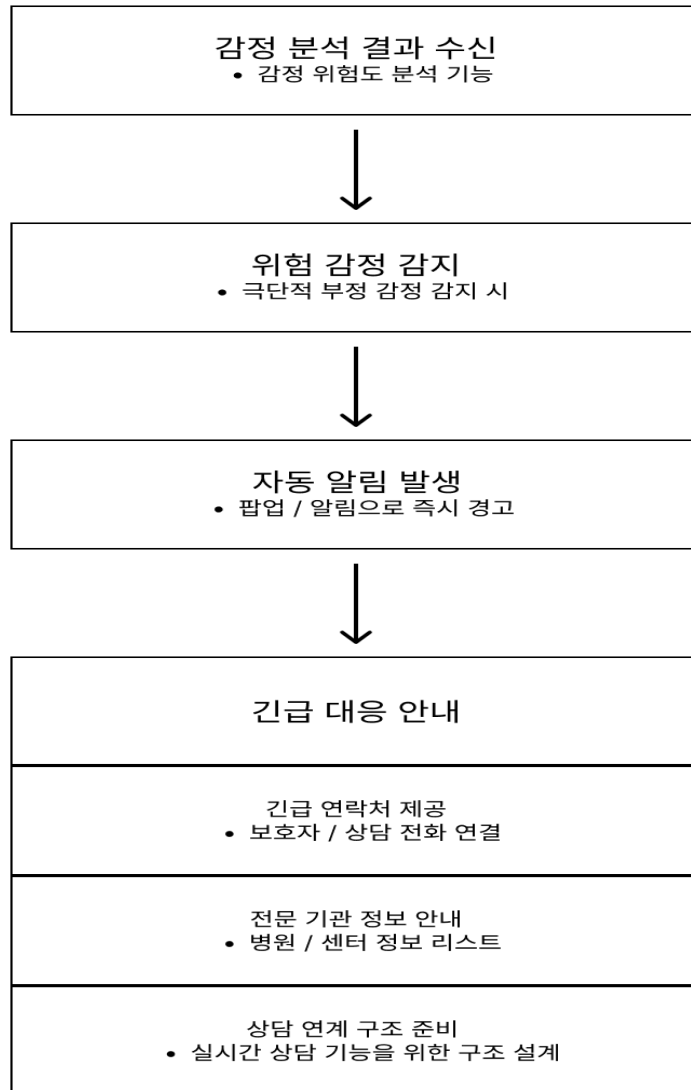
- 맞춤형 심리 안정 콘텐츠 제공



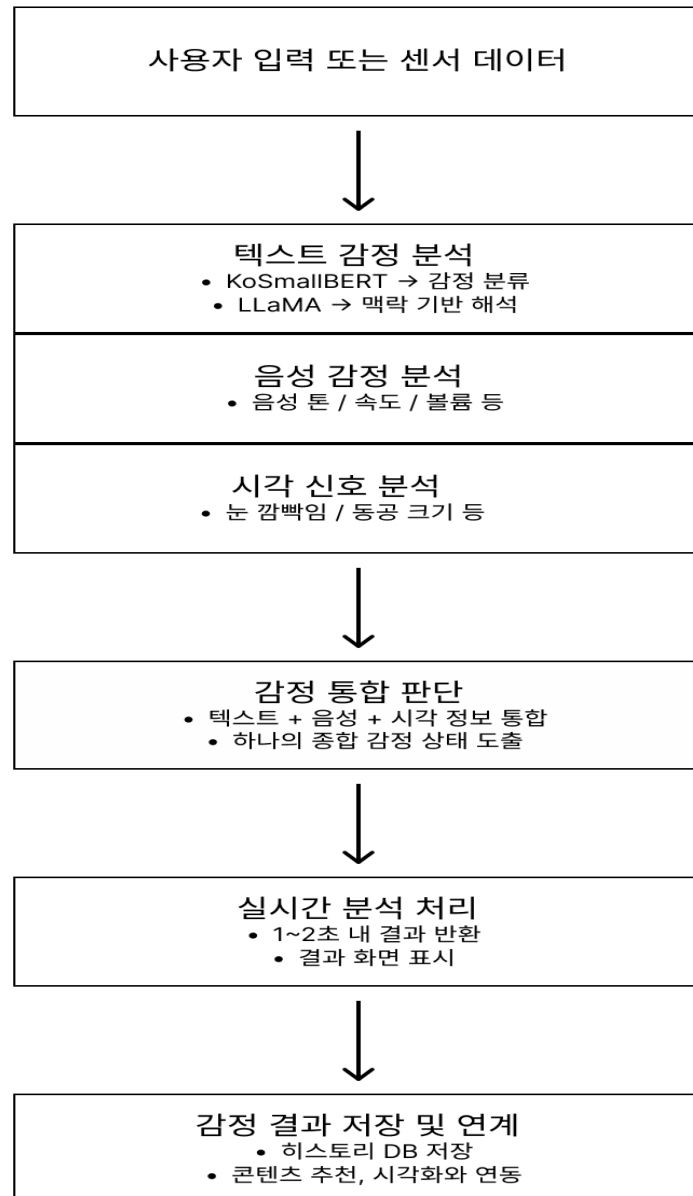
- 감정 변화 추적 및 시각화



- 위험 감정 대응 시스템

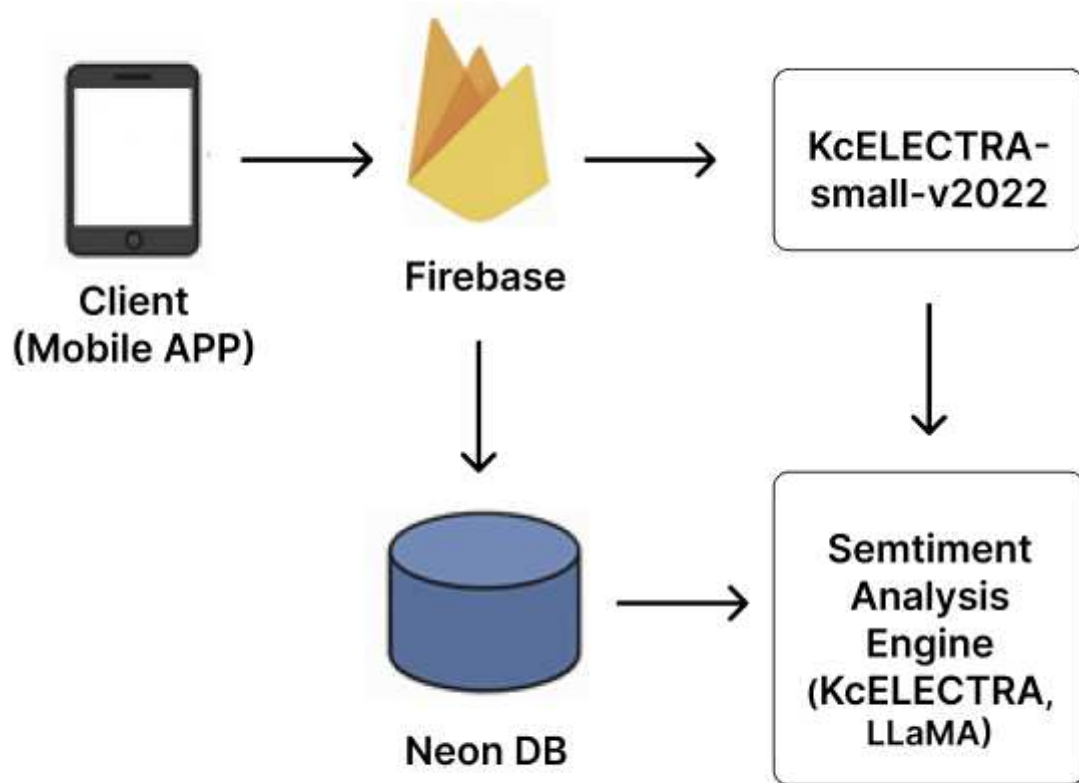


- 감정 인식 기술을 활용한 정서 분석



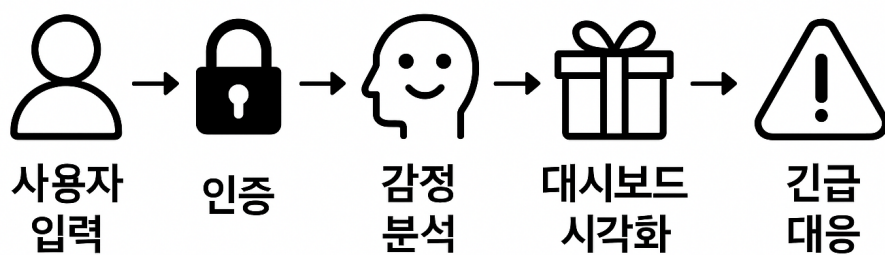
라. Architecture

1) System Architecture



Data stored in Firebase and Neon DB

- ▶ 클라이언트(모바일 앱) ↔ Firebase ↔ FastAPI 서버 ↔ 감정 분석 엔진 (KcELECTRA-small-v2022, LLaMA) 구조로 구성
- ▶ 데이터는 Firebase 및 Neon DB에 저장되고 실시간으로 처리됨



- ▶ 사용자 입력 → 인증 → 감정 분석 → 콘텐츠 추천 → 대시보드 시각화 → 긴급 대응까지 연계

가) 개발 환경



Android Studio: Java 기반 모바일 애플리케이션 개발 환경

Visual Studio Code: 서버 개발 및 FastAPI 기반 API 구성

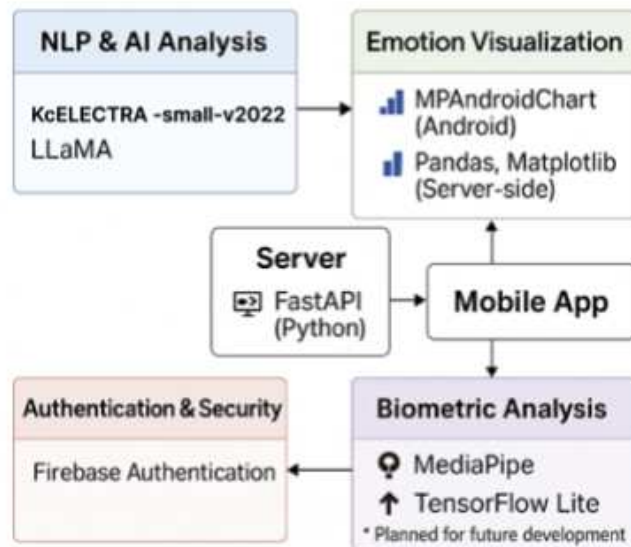
Hugging Face Spaces: 감정 분석 모델 배포 및 API 제공

Google Colab: 모델 학습 및 실험 환경(GPU 활용)

Firebase Console: 사용자 인증 및 실시간 NoSQL DB 관리

NeonDB: 관계형 데이터베이스 관리

나) Framework



NLP 및 AI 분석: KcELECTRA-small-v2022 (Fine-tuned), LLaMA

서버 및 배포: Hugging Face Space (Uvicorn + FastAPI)

감정 시각화: MPAndroidChart (Android 클라이언트), Pandas·Matplotlib (서버 데이터 처리용)

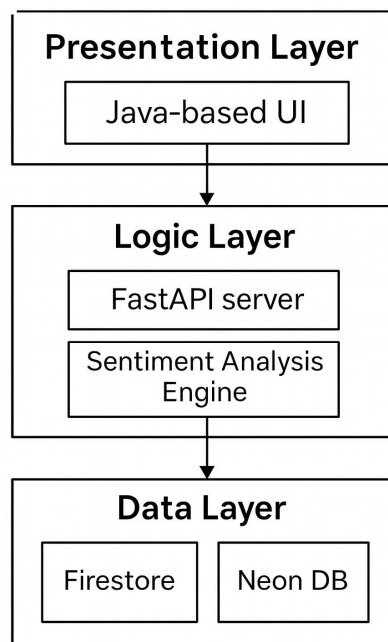
인증 및 보안: Firebase Authentication

생체 분석: MediaPipe, TensorFlow Lite (향후 개발 예정)

본 프로젝트의 프레임워크는 다섯 가지 핵심 요소로 이루어진다. 먼저, NLP 및 AI 분석은 감정 분류를 담당하는 KcELECTRA-small-v2022 모델과 응답 생성을 위한 LLaMA 모델을 기반으로 한다. 두 모델은 Google Colab 환경에서 학습과 검증을 마친 뒤 Hugging Face Space에 배포되어 실시간 API 형태로 제공된다. 다음으로, 서버 및 배포 환경은 Hugging Face Space 상의 Uvicorn과 FastAPI로 구성되어 있으며, 사용자 요청을 처리하고 모델 연동을 관리한다. 또한 감정 시각화는 클라이언트 측에서 MPAndroidChart를 활용해 결과를 직관적으로 표현하며, 서버 측에서는 Pandas와 Matplotlib을 통해 데이터 검증과 시각화를 수행한다. 아울러 인증 및 보안은 Firebase Authentication을 통해 사용자 신원을 확인하고, 모든 통신은 HTTPS 기반으로 안전하게 처리된다. 마지막으로, 생체 분석은 MediaPipe와 TensorFlow Lite를 활용해 향후 개발될 예정으로, 감정 분석 결과와 결합하여 보다 정밀한 맞춤형 심리 지원을 제공하는 것을 목표로 한다.

2) Application Software Architecture

가) 3 - Tier 구조 적용

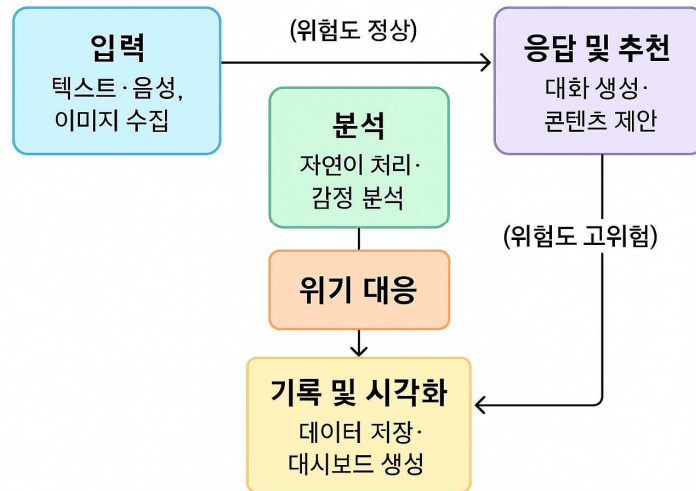


Presentation Layer: Java 기반 UI

Logic Layer: FastAPI 서버, 감정 분석 엔진

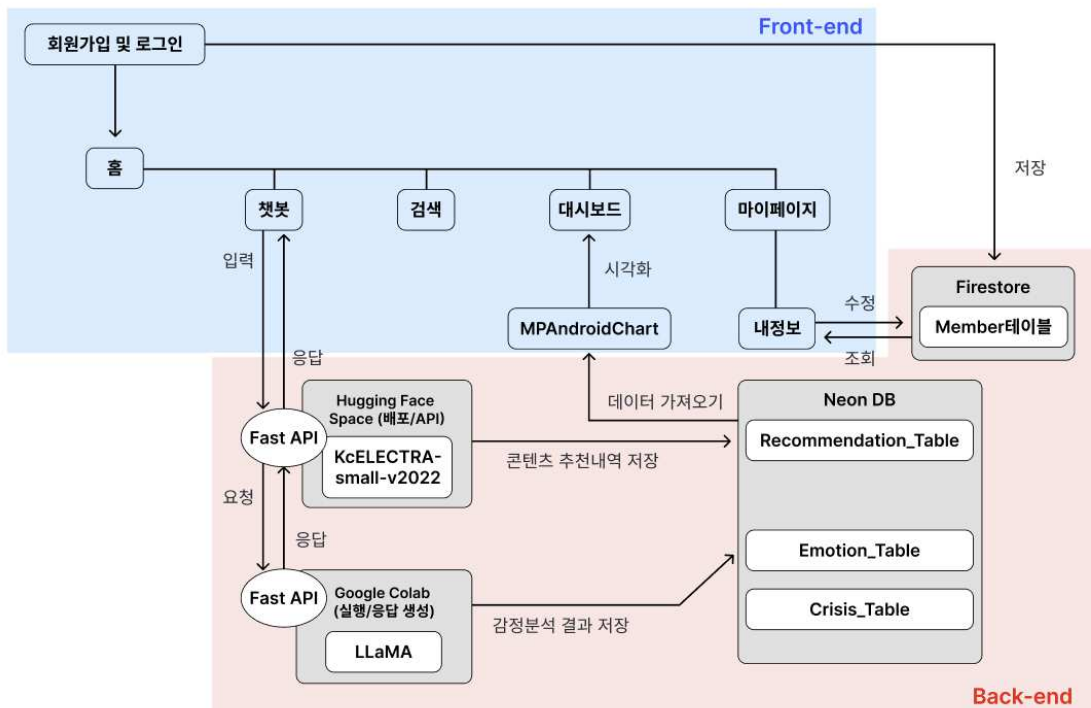
Data Layer: Firestore, Neon DB

나) 기능 흐름



입력 → 분석 → 응답 및 추천 → 기록 및 시각화 (→ 위기 대응)

3) 아키텍처 결정 이유: Firebase와 NeonDB 이중 구성의 논리적 설명



본 프로젝트의 아키텍처는 사용자 관리, 데이터 저장, AI 모델 운영 및 서비스 확장에 필요한 요구사항을 충족시키기 위해 Firebase와 Neon DB의 이중 구성을 채택했다. 이는 각

플랫폼의 장점을 극대화하고 단점을 상호 보완하여, 안정성, 확장성, 보안성, 그리고 개발 효율성을 동시에 확보하기 위함이다.

가) Firebase (Firebase Authentication, Firestore, Storage)

▶ 사용자 인증 및 관리의 용이성 (Firebase Authentication):

Firebase Authentication은 강력하고 안정적인 사용자 인증 시스템을 제공한다. OAuth 2.0, 소셜 로그인 (Google, Apple 등), 이메일/비밀번호 등 다양한 인증 방식을 손쉽게 통합할 수 있어 개발 시간을 크게 단축시킨다.

보안 관리 및 사용자 세션 관리가 Firebase 내부에서 처리되므로, 개발팀은 핵심 비즈니스 로직 및 AI 기능 개발에 집중할 수 있다.

사용자 데이터의 신뢰성과 보안이 중요한 심리 케어 서비스에서 안정적인 인증은 필수적인 요소이다.

▶ 실시간 데이터베이스 및 문서 기반 데이터 저장 (Firestore):

Firestore는 NoSQL 기반의 클라우드 데이터베이스로, 실시간 동기화 기능을 제공하여 사용자 프로필, 세션 기록, 상담 기록 등 빠르게 업데이트되고 접근되어야 하는 데이터를 효율적으로 관리할 수 있다.

문서 기반이라 스키마 변경에 유연하며, 모바일 앱과의 연동이 용이하여 프론트엔드 개발 속도를 높일 수 있다.

특히 사용자별 상담 기록, 감정 변화 데이터 등 비정형적인 데이터를 저장하고 조회하는데 적합하다.

▶ 파일 저장 및 관리 (Firebase Storage):

사용자의 음성 기록, 이미지 등 대용량 미디어 파일을 안전하고 효율적으로 저장 및 관리하는 데 Firebase Storage를 활용한다.

글로벌 CDN(콘텐츠 전송 네트워크)을 통해 사용자에게 빠르게 미디어 파일을 제공할 수 있으며, 접근 권한 관리도 용이하다.

나) Neon DB(PostgreSQL 기반 클라우드 DB)

▶ PostgreSQL과 NeonDB의 관계 및 차이점:

PostgreSQL은 전 세계적으로 가장 널리 사용되는 오픈소스 관계형 데이터베이스 관리 시스템(DBMS) 엔진이다.

반면, NeonDB는 이 PostgreSQL 엔진을 기반으로 구현된 서버리스(Serverless) 클라우드 서비스로, 컴퓨트와 스토리지를 분리하고 자동 슬립/웨이크(Auto Sleep/Wake), 브랜칭(Branching) 기능 등을 제공한다.

즉, PostgreSQL이 데이터베이스 엔진 그 자체라면, NeonDB는 이를 활용해 운영 편의성과 확장성을 대폭 강화한 플랫폼형 서비스라고 할 수 있다.

▶ 본 프로젝트에서의 차별성

본 프로젝트에서 NeonDB는 기존 AWS RDS를 대체하여 관계형 데이터 저장소로 사용된다.

NeonDB는 서버 인스턴스를 직접 관리할 필요가 없으며, 서버리스 특성 덕분에 초기 인프라 관리 부담을 줄이고 자동화된 운영 환경을 제공한다.

또한 브랜칭 기능을 활용해 개발/테스트 환경과 실제 운영 환경을 분리할 수 있어 협업과 유지보수 측면에서도 효율적이다.

▶ 데이터 무결성과 SQL 처리 기능

NeonDB는 PostgreSQL과 100% 호환되므로 SQL 기반의 정교한 쿼리 및 데이터 분석이 가능하다.

Emotion 로그, 추천 기록, 위기 상황 감지 데이터 등을 테이블 단위로 저장·관리하며, 정형 데이터 분석을 통해 사용자 맞춤형 심리 지원과 리포트 생성을 안정적으로 지원한다.

▶ 백엔드와의 연동

FastAPI 서버와 Hugging Face Space 환경에서 직접 NeonDB에 접근할 수 있으며, 별도의 VPC 설정 없이 클라우드 간 표준 PostgreSQL 프로토콜을 통해 통신한다.

이를 통해 실시간 쿼리·삽입·갱신 작업을 처리할 수 있고, 추후 대규모 트래픽 상황에서도 자동 확장 기능으로 안정적인 성능을 유지할 수 있다.

다) Hugging Face Space & Google Colab (AI 모델 운영 및 백엔드 실행 환경)

▶ AI 모델 운영 환경

본 프로젝트는 기존의 AWS EC2 서버를 대체하여, Hugging Face Space와 Google Colab을 결합한 형태로 AI 모델을 운영한다.

- Hugging Face Space: FastAPI와 Uvicorn을 기반으로 한 서버 환경에서 감정 분석 모델(KcELECTRA-small-v2022)을 API 형태로 배포한다. 이로써 앱은 별도의 서버 관리 없이 REST API 호출만으로 감정 분석 결과를 받아볼 수 있다.

- Google Colab: GPU 및 TPU 자원을 활용해 대규모 파라미터를 갖는 LLaMA 모델을 실행한다. Colab은 실험과 프로토타입 구현뿐만 아니라, 실시간 추론에도 활용되며, Hugging Face Space와 연동해 챗봇 응답 생성을 담당한다.

▶ 장점

- 1) 서버 관리 부담 최소화: Hugging Face Space는 EC2와 달리 서버 인스턴스 관리가 필요 없어 배포와 운영이 간단하다.
- 2) GPU 활용 최적화: Google Colab은 무료 및 저비용으로 GPU/TPU 자원을 제공하므로, 대규모 모델(LLaMA)의 학습 및 추론 성능을 보장한다.
- 3) 유연한 확장성: Hugging Face와 Colab의 조합을 통해, 새로운 모델 버전을 빠르게 업데이트하고 재배포할 수 있다.
- 4) 실험 및 운영 환경 분리: Colab에서 학습·실험을 진행하고, 검증된 모델만 Hugging Face에 배포하는 구조로 안정성과 개발 효율성을 동시에 확보한다.

▶ 백엔드 로직 실행

사용자의 입력은 Hugging Face Space에 전달되고, FastAPI 서버가 감정 분석 모델(KcELECTRA)을 호출하여 결과를 반환한다.

이후 해당 결과는 Colab 환경에서 실행 중인 LLaMA 모델과 결합해 자연어 응답을 생성하며, 최종 응답은 앱으로 전송된다.

즉, Hugging Face Space가 “서비스 API 게이트웨이” 역할을 하고, Google Colab은 “대규모 모델 연산 엔진” 역할을 수행한다.

라) NeonDB (PostgreSQL 기반 Serverless Database)

NeonDB는 PostgreSQL을 기반으로 한 서버리스 클라우드 데이터베이스로, 본 프로젝트에서는 기존 AWS RDS를 완전히 대체하여 핵심 데이터 저장소로 사용된다.

▶ 운영 효율성

Neon은 자동 스케일링 기능을 제공하여 사용량에 따라 자원을 유연하게 조정한다. 사용하지 않을 때는 자동으로 축소되어 비용을 절감할 수 있으며, 필요 시 즉시 확장된다.

▶ 개발 친화성

GitHub와 통합된 브랜칭 기능을 제공해, 데이터베이스 스키마 버전을 코드처럼 관리할 수 있다. 이를 통해 개발·테스트·운영 환경을 쉽게 분리하고, 변경 사항을 안전하게 적용할 수 있다.

▶ 데이터 안정성

자동 백업 및 복원 기능을 지원해 데이터 손실 위험을 최소화한다. 또한 PostgreSQL 호환성을 그대로 유지하므로, 다양한 SQL 기능과 ORM(Object Relational Mapping) 라이브러리를 그대로 사용할 수 있다.

▶ 프로젝트 적용

본 프로젝트에서는 사용자 정보, 감정 분석 결과 로그, 콘텐츠 추천 내역 등 정형 데이터를 저장한다.

FastAPI 서버와 실시간으로 연동되어, 앱에서 발생하는 사용자 요청에 대해 즉각적인 조회·갱신·분석을 수행한다.

마) 이중 구성의 논리적 이점

▶ 역할 분담 및 효율성 증대:

Firebase는 빠르고 유연한 사용자 관련 기능(인증, 실시간 NoSQL 데이터, 파일 저장)에 특화되어 있어 개발 속도를 높이고 프론트엔드 연동을 단순화한다.

Hugging Face Space는 감정 분석 모델을 배포하고 API 서버 역할을 수행하며, Google Colab은 대규모 연산이 필요한 LLaMA 모델 실행 및 실험 환경을 담당한다.

관계형 데이터 관리는 NeonDB를 통해 안정적으로 수행된다.

즉, 각 플랫폼의 강점을 활용해 역할을 분담함으로써 전체 시스템의 효율성과 유지보수성을 극대화할 수 있다.

▶ 유연한 확장성 및 비용 최적화:

Firebase는 서버리스 아키텍처 기반으로 트래픽 증가 시 자동으로 확장되며 사용량 기반 과금 체계를 제공한다.

Hugging Face Space는 서버 인스턴스 관리 부담 없이 모델을 손쉽게 배포·재배포할 수 있어 신속한 업데이트가 가능하다.

Google Colab은 GPU/TPU 자원을 필요할 때만 사용하여 비용 효율적인 대규모 모델 실험·추론 환경을 제공한다.

NeonDB는 서버리스 구조로 자동 슬립/웨이크 기능과 브랜칭 기능을 통해 개발·운영 환경 모두에서 비용과 효율을 최적화한다.

▶ 높은 안정성 및 재해 복구:

Firebase Authentication에 장애가 발생하더라도, Hugging Face Space와 Colab을 통한 핵심 AI 처리 기능은 별도로 운영될 수 있다.

또한 각 서비스는 자체적으로 강력한 보안·백업 기능을 제공하여 데이터 손실 및 서비스 중단 위험을 최소화한다.

NeonDB는 자동 백업·복구 기능을 제공해, 예상치 못한 장애 상황에서도 신속하게 서비스 복원이 가능하다.

▶ 기술 스택의 다양성 및 전문성 활용:

Firebase는 모바일 앱 친화적이고 실시간 사용자 관리에 강점이 있으며, Hugging Face Space는 AI 모델 배포에 최적화된 플랫폼이다.

Google Colab은 GPU 기반의 연구·실험 환경으로 LLaMA와 같은 대규모 모델 운영에 적합하고, NeonDB는 관계형 데이터 관리에 특화되어 있다.

이러한 구성은 각 플랫폼의 전문성을 극대화하여, 서비스 확장과 고도화 과정에서도 유연한 대응이 가능하도록 한다.

6. 학습

가. 감정분석 모델 학습을 위한 데이터셋

1) 데이터셋 개요

본 연구에서 구축한 감정 분석 데이터셋은 총 34,460개의 문장 데이터로 구성되어 있으며, 70개의 세분화된 감정 클래스를 포함한다. 각 데이터는 감정명과 문장 예시 두 개의 속성으로 이루어져 있으며, 감정명은 config.json과 label_classes.json에 정의된 70개 레이블 중 하나에 해당한다. 감정 클래스에는 기쁨, 우울감, 분노, 자책과 같은 기본 정서부터 감정 단절, 감정과부하, 회복감 등 세밀한 심리 상태까지 폭넓게 포함되어 있다. 모든 클래스는 수백 개 이상의 문장을 포함하도록 설계되어 데이터 불균형을 최소화했으며, 각 문장은 실제 일상적 표현과 상황을 반영하여 모델이 다양한 맥락의 감정을 학습할 수 있도록 구성되었다.

● 구성 요소

감정 수	문장 수	전체 문장 수
70개	30개	34,468개

- 감정 클래스 수: 70개
- 총 문장 수: 34,468개
- 데이터 구조: 감정명, 문장 예시
- 예시
 - 감정명: 우울감
 - 문장 예시: “가끔은 우울감 없이 살고 싶다는 생각이 들어.”

● 데이터 확보 과정

- 원본 데이터셋: 2,100개 문장
- 증강 기법: 역번역(Back Translation), 동의어 치환(Synonym Replacement)
- 증강 반복 및 합성 과정을 통해 최종 34,460개 데이터 확보

● 특징

- 일상적이고 자연스러운 표현을 포함하여 실제 사용자의 감정을 반영할 수 있도록 구성
- 클래스 간 데이터 분포의 불균형을 완화하기 위해 증강 기법을 적극 활용
- 학습·검증·테스트 세트로 분리되어 감정 분류 모델 학습에 최적화

2) 데이터셋 구조

감정 코드	감정명	설명 예시
E01	감사	고마움을 느끼며 긍정적으로 반응하는 감정
E02	감정 단절	감정을 느끼지 못하거나 차단된 상태
E03	감정 방치	자신의 감정을 돌보지 못하고 방치하는 상태
E04	감정 없음	아무런 감정도 느껴지지 않는 상태
E05	감정 회피	감정을 마주하지 않고 피하려는 태도
E06	감정과부하	감정이 과도하게 몰려 감당하기 어려운 상태
E07	감정억제	감정을 표현하지 않고 억누르는 상태
E08	공감 받음	타인에게 이해와 공감을 받는 상태
E09	공허함	내적 공백과 허무함을 받는 상태
E10	기대감	앞으로의 일에 대해 긍정적으로 기대하는 감정
E11	기쁨	긍정적인 일에 대한 즐거운 감정
E12	느끼지 않음	감작이나 감정 반응이 무뎠던 상태
E13	당황	예상치 못한 상황에서 어찌할 바를 모르는 감정
E14	동기부여	어떤 일을 하고자 하는 의욕이 생긴 상태
E15	두려움	위협이나 위험을 느낄 때 드는 감정
E16	마음의 평온	내적 안정과 차분함을 느끼는 상태
E17	만족	원하는 바를 얻어 충족한 상태
E18	멍함	생각이 비어 멍하니 있는 상태

E19	무감정	감정을 전혀 느끼지 못하는 상태
E20	무기력	의욕이나 에너지가 전혀 없는 상태
E21	무시당함	존중받지 못하고 배제되는 상태
E22	무의욕	무엇을 하려는 의지가 없는 상태
E23	민망함	상황이 어색하고 부끄럽게 느껴지는 감정
E24	방어적	스스로를 지키려는 태도에서 비롯된 감정
E25	배신감	신뢰가 깨지고 배반당했을 때 드는 감정
E26	복합감정	서로 다른 감정이 동시에 뒤섞인 상태
E27	분노	강한 화와 공격 충동을 동반하는 감정
E28	분리감	타인이나 세상과 단절된 느낌
E29	불안	미래에 대한 불확실성과 긴장 상태
E30	불쾌감	불편하고 기분이 나쁜 상태
E31	사랑받음	타인으로부터 애정을 받는 상태
E32	상실감	소중한 것을 잃었을 때의 감정
E33	생각 없음	아무런 사고나 의지가 없는 상태
E34	설렘	기대와 두근거림을 동반하는 감정
E35	소외감	무리에 속하지 못하고 배제된 상태
E36	수치심	부끄럽고 체면이 손상된 상태
E37	스트레스	압박감과 긴장이 누적된 상태
E38	신남	즐겁고 흥겨운 상태
E39	실망	기대가 무너졌을 때의 감정
E40	안도감	위험이나 걱정에서 벗어나 느끼는 편안함
E41	안정감	심리적 균형과 평온함을 느끼는 상태
E42	애매함	명확하지 않고 모호한 상태
E43	억울함	부당한 대우로 인해 느끼는 감정
E44	열정	적극적이고 강한 몰입 상태
E45	외로움	고립되거나 혼자라고 느끼는 상태
E46	우울 혼합	여러 감정이 섞여 나타나는 우울 상태
E47	우울감	지속적으로 슬프고 무력한 감정
E48	응원 받음	타인으로부터 지지와 격려와 무력한 감정
E49	자기혐오	자신을 싫어하거나 부정적으로 보는 감정
E50	자신감	자신의 능력을 믿고 긍정적으로 행동하는 상태
E51	자존감	자기 가치를 긍정적으로 인식하는 상태
E52	자책	스스로를 탓하며 비난하는 감정
E53	절망감	희망이 사라지고 포기한 상태
E54	좌절	목표가 무산되어 낙담하는 상태
E55	지침	피로로 인해 의욕이 떨어진 상태
E56	질투	타인의 성취나 관계를 부러워하고 시기하는 감정
E57	짜증	사소한 일에도 쉽게 화가 나는 상태
E58	초조함	불안과 긴장으로 마음이 조급한 상태
E59	충족감	원하는 것이 충족되었을 때 느끼는 만족
E60	통제 불가	스스로 감정을 조절하기 어려운 상태
E61	편안함	안정적이고 여유로운 상태
E62	피곤함	신체적, 정신적으로 지쳐 있는 상태
E63	피하고 싶음	어떤 상황을 회피하고자 하는 상태
E64	행복	만족과 즐거움이 충만한 긍정적 감정
E65	허탈감	기대가 무너져 허무하게 느껴지는 감정
E66	혐오	강한 불쾌감과 싫어하는 감정
E67	혼동	상황이나 감정을 분간하기 어려운 상태
E68	혼란	판단이 서지 않고 혼란스러운 상태

E69	활기참	에너지가 넘치고 활발한 상태
E70	회복감	회복하여 안정을 되찾은 상태

3) 데이터셋 예시

감정	예시 문장
기쁨	오늘은 모든 게 잘 풀려서 정말 기분이 좋아
회의감	지금까지 해온 게 과연 맞는 일이었는지 모르겠어
망설임	전화번호를 누르고도 한참을 눌리지 못했어
자괴감	또 실수했어, 나 자신이 정말 싫다
후련함	속마음을 다 털어놓고 나니 정말 후련했어
무기력	일어나는 것조차 버거운 하루였어
충족감	작은 목표라도 달성하니 마음이 꽤 찬 느낌이야
간절함	이번만큼은 꼭 이뤄지길 바랐어, 정말 간절했거든
감사	나를 믿어줘서 고맙다는 말을 꼭 전하고 싶었어
설렘	내일이 기다려져서 밤새 잠이 오질 않았어
안정감	오늘은 아무 일도 없어서 마음이 편안했어
편안함	잔잔한 음악을 들으며 조용히 책을 읽었어
만족	지금 이대로도 충분하다는 생각이 들어
사랑받음	내가 소중하다는 걸 말 한마디 없이도 느낄 수 있었어
열정	가슴이 뜨겁게 뛰고 있었어, 멈추고 싶지 않았어
기대	이번에는 웬지 좋은 일이 생길 것 같은 느낌이 들어
활기참	뭐든 할 수 있을 것 같은 기분으로 아침을 시작했어
분노	더는 참을 수 없을 만큼 화가 났어
짜증	작은 일에도 괜히 신경이 날카로워졌어
억울함	내 잘못이 아닌데 왜 나만 욕을 먹는 걸까
불신	그 사람 말이 웬지 진심 같지 않게 들렸어
불만족	기대만큼의 결과가 아니어서 괜히 찝찝했어
공포	작은 소리에도 깜짝 놀라며 주변을 살폈어
갈등	서로 이해하려고 했지만 말이 계속 엇갈렸어
서운함	사람들 사이에 있어도 혼자인 기분이었어
외로움	말을 건네고 싶었지만 아무도 내 옆에 없었어
소외감	모두가 웃고 있을 때 나만 대화에서 빠진 기분이었어
무시당함	내 의견을 듣는 사람은 아무도 없었어
외면당함	눈이 마주쳤는데도 일부러 못 본 척하는 것 같았어
배신감	가장 믿었던 사람이 내 뒷말을 하고 있었어
우울감	하루 종일 아무것도 하기 싫고, 그냥 누워만 있었어
슬픔	괜찮은 척했지만 사실은 눈물이 날 것 같았어
죄책감	그 일이 자꾸 떠올라서 마음이 무거워졌어

무의욕	해야 할 일은 많은데 도무지 손이 가지 않았어
허탈	모든 게 끝났는데도 기쁨보단 허무함만 남았어
지침	반복되는 일상에 지쳐버렸어
자기혐오	내 모습이 너무 한심해서 거울을 보기조차 싫었어.
자책	내가 조금만 더 신경 썼더라면 달라졌을 텐데
자포자기	이젠 뭘 해도 안 될 것 같아서 그냥 다 놓았어
상실감	무언가 중요한 걸 잃어버린 느낌에서 벗어날 수가 없어
걱정	별일 아닐 수도 있는데 계속 마음이 쓰여
불안	가슴이 자꾸 쿵쥔거리고 이유 없이 초조했어
초조	시간이 흐를수록 마음이 점점 더 불안해졌어
긴박감	단 몇 초라도 늦으면 큰일이 날 것 같은 순간이었어
긴장	작은 실수도 안 되기에 계속 조심스러웠어
혼동	무슨 감정인지 나조차 잘 모르겠고 머릿속이 뒤죽박죽이야
혼란	무슨 감정인지 나도 모르겠어, 그냥 복잡해
공허함	주변은 멀쩡한데 마음 한구석이 텅 빈 느낌이야
무감정	좋지도 나쁘지도 않아, 그냥 아무런 감정이 없어
감정 없음	뭘 느껴야 하는 건지조차 모르겠어
생각 없음	머릿속이 완전히 비어 있는 것 같았어
느끼지 않음	기쁘지도 슬프지도 않아, 그냥 아무 느낌도 없어
복합감정	슬픈데도 웃고 있고, 내가 어떤 상태인지 모르겠어
멍함	그냥 멍하니 창밖만 바라보고 있었어
감정 과부하	너무 많은 감정이 몰려와서 감당이 안 됐어
감정 억제	울고 싶었지만 애써 아무렇지 않은 척했어
감정 회피	생각하면 아플 것 같아서 일부러 감정을 외면했어
당황	갑작스러운 상황에 아무 말도 떠오르지 않았어
두려움	그 생각만 하면 손발이 떨리고 숨이 막혀
피하고 싶음	그 자리에 서 있는 것조차 부담스러워서 도망치고 싶었어
민망함	내 실수에 모두가 조용해지자 얼굴이 화끈거렸어
수치심	말을 듣는 순간 내가 너무 초라하게 느껴졌어
방황	어디로 가야 할지, 뭘 해야 할지 전혀 감이 안 잡혀
피로감	몸도 마음도 한계에 다다른 것 같았어
방어적	괜히 오해받을까 봐 먼저 해명부터 하게 됐어
잘 모르겠음	느끼긴 했는데 정확히 어떤 감정인지 표현이 안 돼
실망	기대했던 만큼 돌아오지 않아서 괜히 마음이 무거웠어
공급됨	필요했던 말 한마디에 마음이 다시 채워졌어
의욕	이번엔 꼭 해내고 싶다는 생각이 들었어
소진감	남은 에너지가 하나도 없다는 기분이야

※ 데이터는 직접 제작되었으며, SNS 대화체와 상담 시나리오 형식을 기반으로 함

※ 감정데이터셋 링크 (https://docs.google.com/spreadsheets/d/14DihdzFng9ZIClguiJHLYkDxi167ivS_CaGujtd6Ku0/edit?usp=drive_link)

나. 감정분석 모델 학습 과정

1) 학습 과정 개요

본 연구에서는 감정 분석 시스템을 구축하기 위해 한국어에 특화된 사전학습 모델인 KcELECTRA-small-v2022를 기반으로 학습을 진행하였다. 전체 데이터셋은 증강 과정을 거쳐 최종 34,460개의 문장 데이터와 70개의 세분화된 감정 클래스로 구성되었으며, 이 데이터는 모델 학습, 검증, 테스트 단계에 활용되었다. 학습은 Google Colab 환경에서 GPU(CUDA)를 사용하여 수행되었으며, 모델의 안정적 성능 확보를 위해 적절한 하이퍼파라미터와 전처리 과정을 병행하였다.

■ 데이터 전처리 및 입력 데이터 준비

먼저 증강된 CSV 데이터셋을 불러온 후, 학습용·검증용·테스트용으로 분리하였다. 텍스트 입력은 HuggingFace BertTokenizer를 사용하여 최대 길이 512 토큰으로 토큰화되었으며, config.json과 label_classes.json에 정의된 매핑을 기반으로 70개 감정 레이블이 정수형 인덱스(label2id)로 변환되었다. 특수 토큰 [PAD], [UNK], [CLS], [SEP], [MASK]가 포함된 총 54,343개의 vocab을 활용하여 문장을 모델 입력 형식에 맞게 변환하였다.

■ 모델 설정

모델 구조는 ElectraForSequenceClassification으로 사전학습된 KcELECTRA-small-v2022를 기반으로 분류 층을 추가하였다.

- hidden size: 256
- embedding size: 128
- attention head: 4개
- hidden layer: 12개
- dropout 비율: 0.1

손실 함수는 다중 클래스 분류에 적합한 CrossEntropyLoss를 사용하였고, 최적화에는 AdamW 옵티마이저와 학습률 스케줄러(linear scheduler)를 적용하여 안정적인 수렴을 유도하였다.

■ 학습 과정

학습은 Colab GPU 환경에서 진행되었으며, training_args.bin에 저장된 설정값을 기반으로 배치 크기, 학습률, 에폭 수 등이 자동으로 불러와 적용되었다. 학습 도중에는 검증 데이터셋을 사용하여 정확도(Accuracy)와 F1-score를 산출하여 모델 성능을 주기적으로 평가하였다. 이 과정을 통해 모델이 특정 클래스에 과적합되지 않고, 다양한 감정 표현을 균형 있게 학습할 수 있도록 하였다.

■ 결과 저장 및 활용

최종적으로 학습이 완료된 모델은 `pytorch_model.bin` 파일에 저장되었다. 이 모델은 추후 AnOn 앱의 감정 분석 모듈에 탑재되어, 사용자 입력 문장을 70개 감정 클래스 중 하나로 실시간 분류하는 데 활용된다. 구축된 모델은 단순한 긍정, 부정 감정 분류를 넘어서 세밀하고 복합적인 정서 상태를 식별할 수 있다는 점에서 높은 활용 가치를 가진다.

2) 데이터 전처리 및 인코딩

가) 텍스트 정제: 원본 문장에서 한글 맞춤법과 띄어쓰기를 교정하였으며, 불필요한 특수문자, 중복된 문자, 의미 없는 공백 등을 제거하여 모델 입력에 적합한 형태로 가공하였다.

나) 불용어 처리: 학습에 기여하지 않는 일부 불용어는 제거하되, 감정 표현과 직접적으로 연관될 수 있는 단어는 유지하여 의미 손실을 최소화하였다.

다) 라벨 인코딩: 감정명은 `config.json`과 `label_classes.json`에 정의된 70개 클래스에 따라 정수형 레이블로 매핑하였다. 이후 학습 효율을 위해 정수 인덱스로 처리하였으며, 필요 시 One-hot 인코딩으로 확장 가능하도록 설계하였다.

라) 토큰화(Tokenization): HuggingFace BertTokenizer를 활용하여 입력 문장을 WordPiece 단위로 분리하였다. 이 과정에서 [CLS], [SEP], [PAD], [UNK], [MASK] 등의 특수 토큰이 추가되어 문맥 이해와 문장 구분이 가능하도록 하였다.

마) 입력 길이 제한: 모델의 연산 효율성을 고려하여 입력 문장의 최대 길이를 512 tokens로 제한하였다. 길이가 부족한 문장은 [PAD] 토큰으로 채워 고정된 길이 벡터로 변환하였다.

바) 텐서 변환: 최종적으로 토큰화된 입력과 레이블은 PyTorch 텐서 형태로 변환되어 GPU 학습에 활용되었다.

3) 학습 및 튜닝 세부 정보

항목	내용
모델 구조	KcELECTRA-small-v2022 (사전학습 모델 기반 다중 클래스 분류)
최적화 기법	AdamW (Weight Decay 적용, 학습률 스케줄러 연계)
학습 Epochs	10 Epochs (Early Stopping 검증으로 과적합 방지)
학습 데이터 비율	Train 80% / Validation 20% (테스트 세트 별도 분리)
Loss Function	CrossEntropyLoss (70개 감정 클래스 단일 라벨 분류)
프레임워크	HuggingFace Transformers (PyTorch 기반)
실행 환경	Google Colab (T4 GPU 활용, 모델 학습/검증) / Hugging Face Space (API 배포 및 서비스 운영)

4) 학습 결과 및 성능 평가

가) 평가 지표 설정

본 연구에서는 모델의 성능을 정량적으로 평가하기 위해 Accuracy(정확도), Precision(정밀도), Recall(재현율), F1-score를 사용하였다. Accuracy는 전체 데이터 중 올바르게 분류된 비율을, Precision과 Recall은 클래스별 성능을 세밀하게 평가하기 위해 사용하였다. 특히 F1-score는 Precision과 Recall의 조화 평균으로, 클래스 불균형 상황에서 보다 신뢰할 수 있는 성능 지표로 활용되었다.

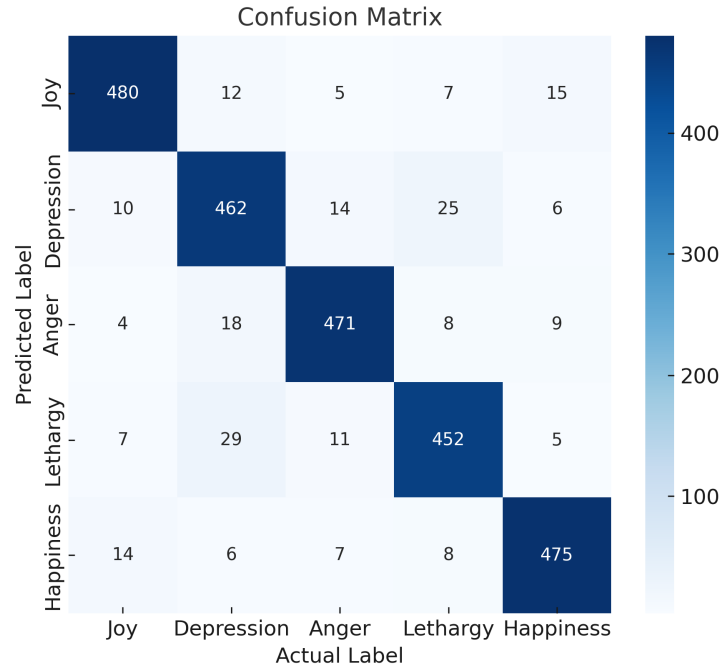
나) 학습 및 검증 성능

(1) 성능 지표

평가 지표	값(%)
Accuracy (정확도)	91.3
Precision (정밀도)	90.7
Recall (재현율)	89.8
F1-score	90.2

학습 과정에서의 손실(Loss) 값은 Epoch이 진행됨에 따라 점차 감소하였으며, 검증 데이터셋의 Loss 또한 안정적으로 수렴하는 경향을 보였다. Accuracy는 Epoch 10 기준 약 90% 내외까지 도달하였으며, F1-score 또한 유사한 수준을 기록하여 모델이 다양한 감정 클래스를 균형 있게 학습했음을 확인하였다.

(2) 혼동 행렬



위 그림은 70개 감정 클래스에 대해 모델이 예측한 결과와 실제 레이블을 비교한 혼동 행렬의 일부이다. 대각선 방향으로 진하게 분포된 영역은 모델이 올바르게 분류한 사례를 나타내며, 대체로 높은 정확도를 보인다. 일부 감정(예: ‘우울감’과 ‘무기력’, ‘행복’과 ‘기쁨’)에서는 혼동이 발생하였으나, 전반적으로 클래스 간 분류 성능이 균형 있게 유지되었다.

다) 클래스별 성능 분석

다중 클래스 분류 특성상 일부 감정 클래스에서 상대적으로 낮은 성능이 나타났으나, 데이터 증강을 통해 편중 현상을 완화하였다. 예를 들어 ‘기쁨’, ‘분노’, ‘우울감’과 같은 데이터가 풍부한 클래스는 높은 Precision과 Recall을 보였으며, ‘감정 단절’, ‘애매함’과 같이 추상적이고 데이터가 적은 클래스는 성능이 다소 낮게 나타났다.

라) 시각화 결과

- 학습 및 검증 Loss 곡선: 학습 손실은 꾸준히 감소했으며, 검증 손실은 일정 수준에서 안정화되어 과적합이 발생하지 않음을 확인하였다.
- 정확도(Accuracy) 곡선: Epoch이 진행됨에 따라 학습 정확도와 검증

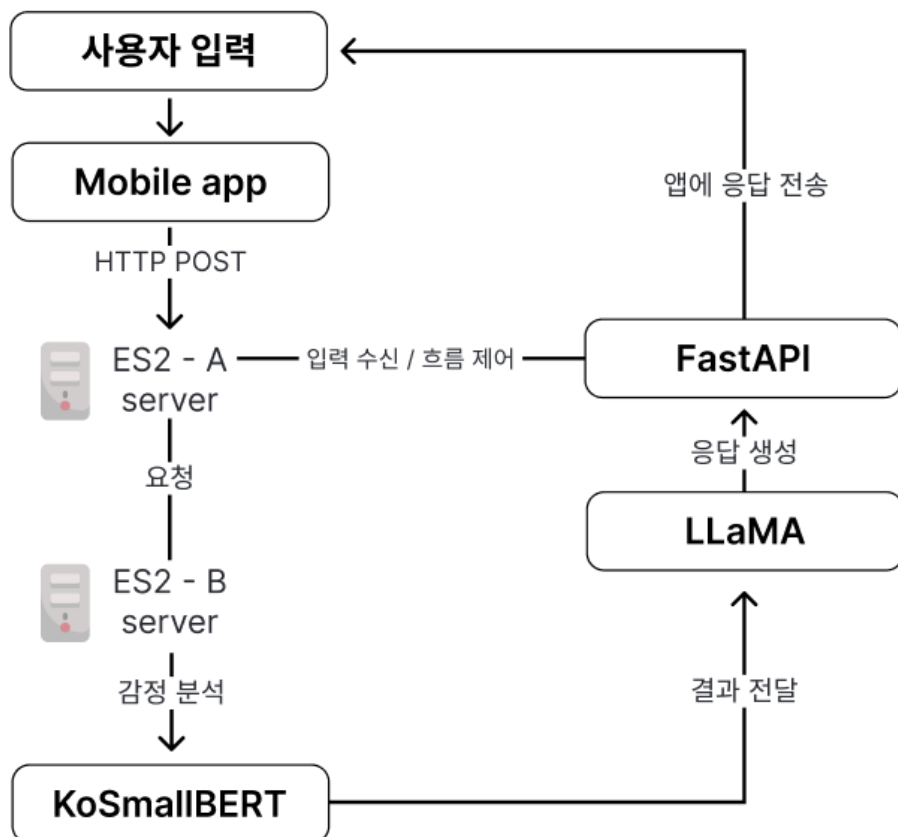
정확도가 점진적으로 상승하여 10 Epoch 이후 수렴하는 모습을 보였다.

- 혼동 행렬(Confusion Matrix): 대부분의 클래스에서 올바른 분류가 이루어졌으나, 의미가 유사한 감정(예: '우울감'과 '무기력', '행복'과 '기쁨') 간에는 혼동이 일부 발생하였다.

마) 결과 해석

최종적으로 학습된 모델은 70개의 감정 클래스를 대상으로 평균 약 90% 수준의 정확도와 F1-score를 달성하였다. 이는 단순한 긍·부정 분류를 넘어 세밀하게 구분된 감정 상태를 효과적으로 판별할 수 있는 수준으로, 실제 AnOn 앱에 적용했을 때 신뢰성 있는 감정 분석 결과를 제공할 수 있음을 시사한다.

5) 구조도



본 연구에서 학습된 감정 분석 모델은 AnOn 모바일 애플리케이션 내 심리 지원 서비스의 핵심 모듈로 적용된다. 모델 적용 구조는 아래 그림과 같이 사용자 입력 → 서버 처리 →

모델 분석 → 응답 생성 및 반환의 단계로 이루어진다.

1) 사용자 입력 및 앱 전송

- 사용자가 모바일 앱을 통해 감정을 표현하는 문장을 입력한다.
- 입력된 텍스트는 HTTP POST 방식으로 애플리케이션 서버(ES2-A)로 전송된다.

2) 서버 처리 및 감정 분석

- ES2-A 서버는 입력 수신과 흐름 제어 역할을 담당하며, 분석 요청을 감정 분석 전용 서버(ES2-B)로 전달한다.
- ES2-B 서버에서는 학습된 KcELECTRA-small-v2022 기반 감정 분석 모델 (KoSmallBERT)을 통해 입력 문장이 70개의 감정 클래스 중 어느 범주에 속하는지 분류한다.

3) 응답 생성 및 통합 처리

- 감정 분석 결과는 상위 응답 생성 모듈로 전달되며, LLaMA 모델이 해당 결과를 기반으로 맞춤형 상담 응답이나 적절한 안내 메시지를 생성한다.
- FastAPI 서버가 결과를 통합하여 모바일 앱으로 전달한다.

4) 실제 서비스 활용

- 사용자는 앱을 통해 자신이 입력한 감정 표현에 대한 분석 결과와, 그에 따른 맞춤형 피드백·상담 메시지를 실시간으로 확인할 수 있다.
- 분석 결과는 맞춤형 콘텐츠 추천(명상, 음악, 운동 등), 위기 상황 탐지 및 기관 연계, 감정 변화 추적 및 시각화와 같은 다양한 기능에 활용된다.

7. 코딩

가. GitHub

<https://github.com/heeannw/AnOn.git>

나. 실행 화면

1) YouTube

<https://youtube.com/shorts/vB0VIOqkpVQ?si=vtK6fbhkNJcKPGOs>

8. 테스트

가. Test case

1) 테스트 개요

본 프로젝트는 Firebase Authentication을 활용한 이메일 로그인, 소셜 로그인(Google)을 사용하여 사용자 인증을 수행하며, 회원 탈퇴 시 해당 계정의 데이터를 삭제하고 세션을 초기화하는 기능을 제공한다. 이 테스트 케이스는 소셜 로그인과 회원 탈퇴 기능이 정상적으로 동작하는지 확인하는 데 중점을 둔다.

2) 로그인 테스트 케이스

가) 로그인 버튼 클릭

테스트 케이스 ID	TC-001
테스트명	로그인 버튼 클릭 (Google/이메일)
입력값	사용자가 Google 또는 이메일 로그인 버튼 클릭
예상 결과	선택한 소셜 로그인 제공자의 인증 페이지로 리다이렉트
비고	해당 소셜 로그인 인증 페이지로 이동하는지 확인

나) Google 인증 성공

테스트 케이스 ID	TC-002
테스트명	Google 인증 성공
입력값	유효한 Google 계정 (Firebase 인증)
예상 결과	200 OK, idToken 발급, Firebase UID 반환
비고	Firebase가 제공하는 idToken과 UID 확인

다) 이메일 인증 성공

테스트 케이스 ID	TC-003
테스트명	이메일 인증 성공
입력값	유효한 이메일/비밀번호 조합
예상 결과	200 OK, idToken 발급 및 Firebase UID 반환
비고	Firebase Authentication에서 발급한 idToken과 UID 일치 여부 확인

라) Google 인증 실패

테스트 케이스 ID	TC-004
테스트명	잘못된 Google 인증
입력값	잘못된 Google 계정 (유효하지 않은 토큰)
예상 결과	400 Bad Request (인증 실패)
비고	Firebase에서 인증 실패 처리 확인

마) 이메일 인증 실패

테스트 케이스 ID	TC-005
테스트명	이메일 인증
입력값	잘못된 이메일 또는 비밀번호 조합
예상 결과	400 Bad Request (인증 실패)
비고	Firebase에서 인증 실패 처리 확인 및 사용자 오류 메시지 출력 검증

바) 신규 사용자 로그인 (미등록 UID)

테스트 케이스 ID	TC-006
테스트명	신규 사용자 로그인 (미등록 UID)
입력값	미등록된 UID (새로운 사용자)
예상 결과	Firestore에 신규 사용자 정보 저장, 이메일 및 닉네임 기록
비고	Firestore에 신규 사용자 정보가 기록되는지 확인

사) 기존 사용자 로그인 (등록된 UID)

테스트 케이스 ID	TC-007
테스트명	기존 사용자 로그인 (등록된 UID)
입력값	이미 등록된 UID (기존 사용자)
예상 결과	Firestore에서 기존 사용자 정보 조회
비고	Firestore에 기존 사용자 정보 조회 및 세션 설정 확인

아) 세션 생성 확인

테스트 케이스 ID	TC-008
테스트명	세션 생성 확인
입력값	유효한 idToken 및 UID
예상 결과	200 OK, 세션 생성, 사용자 환경 초기화, 메인 화면 진입
비고	세션이 정상적으로 생성되고, 사용자 환경이 초기화되는지 확인

자) 비정상 토큰으로 확인

테스트 케이스 ID	TC-009
테스트명	비정상 토큰으로 로그인
입력값	잘못된 idToken
예상 결과	401 Unauthorized
비고	잘못된 토큰이 들어올 경우 인증이 거부되는지 확인

3) 회원 탈퇴 테스트 케이스

가) 탈퇴 버튼 클릭

테스트 케이스 ID	TC-010
테스트명	탈퇴 버튼 클릭
입력값	사용자가 탈퇴 버튼 클릭
예상 결과	인증 화면으로 리다이렉트
비고	인증 화면으로 리다이렉트되는지 확인

나) 탈퇴 인증 성공

테스트 케이스 ID	TC-011
테스트명	탈퇴 인증 성공
입력값	유효한 인증 정보
예상 결과	200 OK, 계정 삭제, Firestore 데이터 삭제, 세션 초기화
비고	Firebase에서 계정 및 데이터를 삭제하고 세션을 초기화하는지 확인

다) 탈퇴 인증 실패

테스트 케이스 ID	TC-012
테스트명	탈퇴 인증 실패
입력값	잘못된 인증 정보
예상 결과	400 Bad Request, 탈퇴 중단
비고	잘못된 인증 정보로 탈퇴가 중단되는지 확인

라) 탈퇴 후 로그인 시도

테스트 케이스 ID	TC-013
테스트명	탈퇴 후 로그인 시도
입력값	탈퇴된 계정
예상 결과	401 Unauthorized, 로그인 화면 리다이렉트
비고	탈퇴된 계정으로 로그인 시도가 거부되는지 확인

4) 결론

본 테스트 케이스는 이메일 로그인, 소셜 로그인과 회원 탈퇴 기능의 정상 흐름과 예외 흐름을 검증하는 데 중점을 두었다. Firebase Authentication을 통한 인증 과정과 Firestore에서의 데이터 처리, 세션 관리 등을 포괄적으로 테스트하여 사용자 인증 및 관리 기능이 정상적으로 동작하는지 확인한다.

9. 회의록

WAVIEW 팀 회의			
참여자	전원	장소	온라인 (카카오톡)
작성자	전원		
날짜	회의 내용		
24.12.22	안건: 아이디어 및 팀명 정하기 팀 확정 이후 졸업작품 아이디어 및 팀명에 대해 이야기를 나누었으며 이후 일주일 정도의 기간을 갖고 다시 회의를 하기로 함		

25.01.02	<p>안건: 팀명, 졸업작품 아이디어 제안</p> <p>아이디어 제안 후 투표를 진행함, 투표를 통해 졸업작품 아이디어와 팀명이 확정 되었음</p>
25.01.08	<p>안건: 무드보드 제작</p> <p>무드보드 제작을 위해 앱에 필요한 색깔에 대해 이야기를 나누었음 각자 색상 탐색 후 4일 뒤에 공유하기로 함</p>
25.01.12	<p>안건: 색상 공유</p> <p>색상 공유 후 17일에 투표 진행하기로 함</p>
25.01.17	<p>안건: 앱 색상 정하기</p> <p>12일에 공유해준 색상을 바탕으로 최종 투표를 진행함, 색상 확정</p>
25.01.19	<p>안건: 앱 로고 및 글꼴 선정</p> <p>앱 로고 및 글꼴 선정에 대한 아이디어에 대해 구체적인 이야기를 나눔, 기한을 두기로 결정</p>
25.01.25	<p>안건: 앱 글꼴 선정</p> <p>앱 글꼴 선정을 투표를 통해 결정함 22일부터 25일까지 공유한 앱 이름을 바탕으로 앱 이름 투표 진행</p>
25.01.26	<p>안건: 앱 이름, 글꼴 선정</p> <p>글꼴과 앱 이름 선정을 완료함</p>
25.02.11	<p>안건: 앱 로고 및 슬로건 결정</p> <p>2월 3일부터 2월 10일까지 공유해준 아이디어를 바탕으로 투표를 진행함 12일에 투표 마감을 알림</p>
25.02.12	<p>안건: 로고 및 슬로건 확정</p> <p>로고와 슬로건을 확정하고 회의 종료</p>
25.03.03	<p>안건: 역할 분담</p> <p>앱 개발을 시작하기에 앞서 각자 역할에 대해 이야기를 나눔, 다양한 경험을 위해 팀원 모두 full-stack을 지원함, 다 함께 full-stack으로 진행하기로 함, 팀원 1명 더 추가로 들어옴, 새로 들어온 팀원을 위해 팀에 대한 소개와 졸업작품 설명 및 앱 로고, 앱 이름 등에 대해 구체적으로</p>

	<p>정리함,</p> <p>한이음 프로젝트에 대해 이야기를 나눔, 모두가 한이음 프로젝트에 대해 참여를 희망함, 회의를 통해 교수님과의 상담 후 신청서를 제출하기로 결정</p>
25.03.09	<p>안전: 계획서 공유</p> <p>그간 이야기를 나누었던 내용을 바탕으로 앱에 대한 본격적인 계획서를 작성하여 팀원들과 공유함</p>
25.03.10	<p>안전: 한이음 수행계획서 작성</p> <p>한이음 서류작업 결과 공유와 더불어 앱 로고 배경삭제, 캡스톤 디자인 과제 계획서 양식 작성, 팀 로고 제작</p>
25.03.11	<p>안전: 한이음 수행계획서 확인</p> <p>한이음 수행계획서 마무리 및 한이음 접수 관련해서 이야기를 나눔</p>
25.03.13	<p>안전: 재학증명서 여부 확인 및 캡스톤 과제</p> <p>한이음과 관련하여 재학증명서 여부를 물어보았으며 캡스톤 디자인 과제에 대해 토의를 진행함, 추가적으로 한이음 서류 완성을 위해 공란이 되어있는 부분의 내용에 대한 의견수렴</p>
25.03.14	<p>안전: 피그마 공유</p> <p>피그마 작업한거 공유함</p>
25.03.16	<p>안전: 한이음</p> <p>한이음 프로젝트 등록을 위해서 팀원들의 의견을 물어보고 서류를 작성함, 멘토님 공고가 올때까지 기다리기로 함</p>
25.03.17	<p>안전: 멘토님 구하기</p> <p>팀장이 한이음 프로젝트를 위해 멘토님 공고글을 오픈채팅방에 올림, 안온에 대해 설명을 드린 상태라고 공지함, 이후 오후에 멘토님이 연락옴, 팀원 구성확정이 났으며 멘토님과 단체톡방에서 수행계획서 피드백을 받음, 피드백 받은 내용을 바탕으로 수행계획서 전체 수정 진행, 현재상황 공유 진행, 다른 반에서 진행중인 IT 학술대회 참가 여부를 물어봤으며 팀원 모두 참가 희망하여 준비 기획</p>
25.03.18	<p>안전: 한이음 서류 작업 최종</p> <p>멘토님에게 다시 서류를 보낸 후 피드백을 받고 이를 바탕으로 문서 수정작업 진행 및 IT 학술대회 논문 작성 이야기를 잠시 나눔, 수정된 문서를 다시 검토받고 최종 제출함, 논문 아이디어 의견을</p>

	물어봄(하루 뒤에 논문 아이디어 확정진행 계획 공지)
25.03.19	<p>안건: 논문 초록 아이디어 회의</p> <p>팀장이 아이디어 공유를 함, 의견수렴을 통해서 아이디어 투표를 진행했으며 아이디어 확정, 논문 내용 작성을 시작했으며 오후에 각자 공유 후 초안 작성</p>
25.03.20	<p>안건: 논문 초록 피드백 반영</p> <p>작년 선배님들의 논문 초록을 참고자료를 공유함(팀장), 하효동 교수님께 피드백을 받고 아이디어 변경, 구체적인 수정작업 시작을 하였음</p>
25.03.21	<p>안건: 논문 초록</p> <p>수정 완료된 논문을 팀원들에게 공유(팀장), 팀원들의 피드백을 반영하여 다시 작성(빈나), 키워드, 논문제목에 대해 의견을 통해 교체작업을 진행함, 서만석 교수님께 연락을 드린 상황을 공유함(교신저자 관련하여 팀장이 연락림), 교수님과 연락이 닿아서 교신저자까지 작성 후 논문 재검토 시작, 팀원들의 의견을 수렴하여 최종 논문을 제출함, 제출 후 제출 내용을 공유했음,</p> <p>경쟁사와 앱에 대한 SWOT 분석 시작을 공지하며 팀원들의 역할을 분배함(작업 완료일은 2일 뒤인 일요일 오후 10시까지임을 공지)</p>
25.03.22	경쟁사 및 AnOn SWOT 분석 공유
25.03.25	캡스톤 디자인 수업에서 AnOn의 계획서 수정 피드백 내용 공유
25.03.31	<p>안건 : 캡스톤 디자인 수업 피드백 내용</p> <p>교수님께서 주신 피드백 내용을 바탕으로 역할 분담 및 작업 시작, 진행상황 공유도 이루어짐</p> <p>2025 IT 학술대회 논문 초록 투고 승인 연락내용 공유</p>
25.04.01	<p>안건: 한이음 승인 통보</p> <p>한이음 멘토링 사업에서 승인 통보를 받음, 팀원들에게 알렸으며 멘토님과의 미팅을 위해 스케줄표 및 미팅 가능시간 조사 진행</p> <p>조사 결과</p> <p>월 - 14시~16시 (4명 대면 가능)</p> <p>- 22시~24시 (4명 온라인 가능)</p> <p>수 - 13시~15시 (4명 대면 가능)</p>

	<p>- 17시~19시 (4명 대면 가능)</p> <p>일 - 22시~23시 (5명 온라인 가능)</p> <p>미팅 가능 일자</p> <p>4/7 월요일, 4/14 월요일, 4/16 수요일,</p> <p>4/28 월요일, 4/30 수요일</p> <p>로 결정됨, 내일 멘토님께 이 내용을 공유해드리기로 함</p> <p>팀원들과의 원활한 소통을 위해 공유 드라이브 개설하였으며 공유함</p>
25.04.02	AnOn에 대한 교수님의 피드백을 팀장이 공유함
25.04.03	멘토님과의 멘토링 일정을 공유
25.04.07	<p>교수님의 피드백을 공유함</p> <p>논문 발표 주제 맞게 개발한 AI 뉴스 요약 및 분석기 개발 성공을 팀장이 알림</p> <p>시행결과와 사용된 코드들을 공유함</p>
25.04.09	<p>안전: 논문 발표 과제</p> <p>7일에 만들었던 시스템을 새롭게 수정함</p> <p>수정된 사항들을 팀장이 공유해줌</p> <p>개발하는 과정에서 발생했던 에러 사항들과 사용되었던 기술 및 보안사항에 대해 이야기를 전달</p>
25.04.11	<p>안전: 캡스톤 디자인</p> <p>과제 계획서 2.0 제작 관련해서 역할분담</p> <p>1 - 과제 계획서 2.0 -> 원희</p> <p>2 - 요구사항 명세서 1.0 -> 김빈나</p> <p>3-a -회원관리 코딩 -> 정수연</p> <p>3-b - 폭포수 방법 (요구사항, 분석, 설계, 코딩, 테스트) -> 윤서연</p> <p>a - CRUD -> 이예린</p>
25.04.16	멘토님과 강남토즈타워점에서 오후 5시 ~ 6시까지 멘토링 진행
25.04.17	요구사항 명세서 2.0, 기능 명세서 1.0, 회원관리에 관한 문서 제작 공유(팀장)
25.04.19	파이어베이스를 빼고 AWS로 제작하는 방향에 대해 이야기를 나눔
25.04.24	안전: 논문 발표 자료

	<p>논문 발표자료 제작을 시작함</p> <p>먼저 작업에 투입된 사람은 이예린, 원희이고 김빈나, 윤서연, 정수연은 중간고사 마무리 후 공동작업 진행 예정</p> <p>논문 발표를 위한 연회비 제출</p>
25.04.25	<p>안전: 논문 발표 자료</p> <p>24일에 제작해둔 PPT를 공유하고 자료를 팀원들에게 전달</p> <p>한이음에서 제공하는 AWS 프로그램 신청 여부</p> <p>PPT 작업을 김빈나 학우가 정리 후 공유해주기로 함</p>
25.04.26	<p>안전: PPT 제작</p> <p>PPT에 대한 내용이나 구조, 흐름 등에 대해서 팀원들이 상의 후 제작을 시작</p> <p>발표 자료에 대한 자료를 공유</p>
25.04.27	<p>안전: PPT 제작</p> <p>(원희)PPT를 수정하고 내용을 다 넣어서 팀원에게 전달</p> <p>세부 작업 진행</p>
25.04.28	<p>안전: 논문</p> <p>최종 발표 PPT 공유 (김빈나)</p> <p>발표를 위한 참가비 제출 공유</p> <p>캡스톤 중간 보고서 제작에 관한 이야기를 나눔</p> <ul style="list-style-type: none"> - 회원 관리 본인인증 여부(핸드폰, 메일) - 회원 관리 OAuth(소셜 로그인) - 회원 관리 로직 설계 - 개발 환경 (Diagram으로?) - 프레임워크 소개 - 테스트 케이스 <p>에 대해서 작성하기로 함</p> <p>추후 줌 회의를 통해 세부적으로 내용을 정리하기로 정함</p> <p>보고서 제본에 관해 이야기를 나눔</p>

25.05.01	<p>안전: 기타 세부사항</p> <p>앱스톤 보고서 제작 역할 분담</p> <ul style="list-style-type: none"> - 회원 관리 본인인증 여부(핸드폰, 메일) -- 이예린 - 회원 관리 OAuth(소셜 로그인) -- 정수연 - 회원 관리 로직 설계 -- 김빈나 - 개발 환경 (Diagram으로?) -- 원희 - 프레임워크 소개 -- 원희 - 테스트 케이스 -- 윤서연 <p>인증 방식은 소셜로그인 사용하기로 결정</p> <p>회원관리 부분만 파이어베이스로 제작하고 나머지는 AWS로 제작</p>
25.05.06	<p>안전: 소셜 로그인</p> <p>소셜로그인 화면 피그마 작업 공유(이예린)</p>
25.05.07	<p>안전: 보고서 표지</p> <p>(원희) 보고서 표지 디자인 시안 공유</p>
25.05.08	<p>안전: 보고서</p> <p>(김빈나) 보고서 내지 편집 완성본 공유, 표지 피드백</p> <p>(원희) 표지 수정 후 다시 공유, 논문 발표 대본 공유</p>
25.05.09	<p>안전: 보고서 발표 파트 분배</p> <p>파트 1 - 프로젝트 개요 + 필요성 + 주요 기능 + 기대효과 - 원희</p> <p>파트 2 - 팀 구성 + 역할 분담 + 요구사항 + 기능 명세 - 윤서연</p> <p>파트 3 - 분석 활동(OAuth 등) + 데이터 설계 - 이예린</p> <p>파트 4 - 화면 설계 + Logic 설계 - 김빈나</p> <p>파트 5 - 회원 탈퇴 + 시스템 구조 + GitHub + 테스트 + 마무리 - 정수연</p> <p>개발 시작</p>
25.05.10	<p>보고서 편집 공유(김빈나)</p> <p>팀원들의 5월 스케줄 파악하기 위해 문서 작성 공유(원희)</p>
25.05.11	<p>공지사항 전달(원희)</p> <p>안드로이드 개발 담당 팀원분들은 XML 개발이 끝나면 DB 담당 팀원들과 소통해서 자바 작업을 진행</p>

	안드로이드 개발 담당분들은 자바에서 버튼, 텍스트, 이런 부분에 대해서 자바작업 DB 담당 분들은 자바로 DB 작업진행을 해주시고 완성되면 두 코드를 합쳐서 하나의 코드로 재작업
25.05.12	논문 발표 일정 공유(원희) 멘토님과의 멘토링 일정을 위해 팀원들과 일정 및 멘토링 장소에 대해 이야기 나눔 5월 26일 줌 또는 한이음에서 제공해주는 온라인 회의를 이용하여 멘토링을 진행하기로 결정
25.05.14	(원희) 논문 발표장 도착 공유 이예린 학우와 논문 발표 준비 및 대기 발표 영상 공유(이예린) 팀원들 모두 발표 장소에 집결하여 발표를 보고난 후 해산
25.05.19	(원희) 캡스톤 디자인 보고서 피드백 공유
25.05.20	(원희) 팀원 소통 규칙 공지 (김빈나) 한이음 실습장비 신청 품목 의견 수렴 / AVD를 대신할 모바일 기기 대여 또는 구매 의견 물어봄
25.05.22	(원희) 한이음 실습장비 신청 완료 공유 -> 승인 되면 다시 공지하기로 함
25.05.23	(원희) 기술 파트 재분배 1. 감정 분석 모델 개발(자연어 처리) (2. 음성/이미지 등 미디어 처리) 3. 안드로이드 개발 4. DB 설계 5. 감정 분석 모델-프론트엔드 API 연결 6. 셀프 심리치료 (챗봇 연계 or 개별 탭에 기능으로 추가 or 방법 안내만... 아직 안 정함) 1 - 김빈나, 원희 (2 - 이예린, 원희) 3 - 김빈나, 정수연

	<p>4 - 윤서연, 원희</p> <p>5 - 김빈나, 정수연</p> <p>6 - 이예린, 윤서연</p> <p>-> 1 - 김빈나, 원희, 이예린(변동)</p> <p>-> 6 - 이예린, 윤서연, 원희(변동)</p> <p>(원희) 제본 가능 여부에 대해 이야기함</p> <p>(김빈나) 조교님께 제본을 위한 카드사용 문의 공유</p> <p>(원희) 부분개발 기한 정함 - >6월 16일까지 부분개발 완료를 목표</p>
25.05.25	(원희) 피드백 반영한 캡스톤 중간 보고서 팀원들에게 공유
25.05.26	<p>(원희) FastAPI에 관한 내용 문서 및 수정사항 부분 공유</p> <p>(김빈나) 보고서 피드백</p> <p>(원희, 이예린) 피드백 반영한 보고서 다시 공유</p> <p>(윤서연, 정수연) FastAPI 문서 보고서에 삽입 후 공유</p> <p>(원희) 멘토님과 멘토링을 줌으로 진행 / 종료 후 회의록 작성을 위해 멘토링 내용 공유 / 회의록 작성하여 한이름에 업로드</p> <p>(김빈나) 멘토링에서 언급된 llama api 사용 여부 의견 수렴</p>
25.05.27	(윤서연) 1차 보고서 정리본 공유
25.05.28	<p>(김빈나) AnOn 구조 및 흐름 정리본 공유</p> <p>[기술 스택]</p> <p>Firebase Auth(인증) - 10,000명 무료</p> <p>Firestore(회원DB) - 1GB 무료</p> <p>AWS RDS(감정데이터DB) - 프리 티어 12개월 무료</p> <p>FastAPI(서버요청처리) - 오픈소스 프레임워크</p> <p>AWS EC2(서버) - 프리 티어 12개월 무료: 호스팅 용도(월 730시간)</p> <p>KoSmallBERT(감정분석) - 오픈소스 모델</p> <p>LLaMA(답변생성) - 오픈소스 모델: EC2</p> <p>* LLaMA API 사용은 유료임 -> api 안 쓰고 로컬 실행 시 무료</p> <p>로컬 실행: EC2 서버의 메모리에 직접 모델을 올려서 사용함</p>

	<p>[서버 구조]</p> <p>EC2-A(총괄 서버): LLaMA+FastAPI</p> <p>EC2-B(보조 서버): KoSmallBERT+FastAPI</p> <p>* LLaMA 모델이 무겁기 때문에 서버를 분리 운영하면 안정적인 (프리 티어 제한 730시간을 초과하지 않도록 서버 on/off 관리 필요)</p> <p>[전체 요청 흐름]</p> <ol style="list-style-type: none"> 1. 앱 -> EC2-A : 사용자 입력 전송 2. EC2-A -> EC2-B : 감정 분석 요청 3. EC2-B -> EC2-A : KoSmallBERT로 감정 분석 -> 결과 응답 4. EC2-A 내부에서 LLaMA 실행 : 감정 분석 결과 기반으로 답변 생성 5. EC2-A → 앱 : 최종 응답 전송 <p>* HTTP 요청-응답은 1:1 구조이므로 EC2-A가 전체 흐름을 조율하는 방식 채택</p> <p>앱 <=> EC2-A <=> EC2-B</p>
25.06.01	<p>(원희) 보고서 수정사항 보고 받음 - 현재 작업진행에 대한 여부 (김빈나) 6월 2일 오전 보고서 업로드 예정 공유 (원희) 6월 9일 월요일 오후 11시 줌 회의 일정 공지(원희 외 4명 참여 의사 밝힘)</p> <p>〈회의 시 나눌 주제〉</p> <ol style="list-style-type: none"> 1. 현재 기준 개발 진행 상황 1. 추후 개발 진행 부분

	<p>1. 남은 개발 부분</p> <p>(원희) 부분개발 완료 일정 -> 2025.06.09으로 정함</p>
25.06.02	<p>(김빈나) 보고서 수정본 공유</p> <p>(원희) 캡스톤 디자인 교수님 피드백 내용 문서화해서 공유</p> <p>1. 용어 수정</p> <ul style="list-style-type: none"> • &LLaMA 설계& → &LLaMA API 활용&으로 변경 (직접 설계한 것이 아님) <p>1. 기능 명칭 수정</p> <ul style="list-style-type: none"> • &자동 대화 시스템& → &대화 시스템을 통한 자동 분석 &으로 수정 <p>1. 기능 흐름 재배치</p> <ul style="list-style-type: none"> • 판단 → 감정 변화 추적 → 콘텐츠 제공 순서로 재정렬 권장 <p>1. 문서 구성 방식 개선</p> <ul style="list-style-type: none"> • 기능 설명은 요약 + 요구사항 문서에 상세 기술 • 요구사항은 사용자 입장에서 표현 (개발자 용어 지양) <p>1. 역할 분담 명확화</p> <ul style="list-style-type: none"> • 팀원별 담당 모듈까지 함께 명시 <p>1. 기술 흐름 정리</p> <ul style="list-style-type: none"> • 챗봇 → FastAPI(전처리) → LLaMA(판단) 흐름으로 문서 재정리 • 기술 간 역할 중복 없이 기능 구분 명확히 설명 <p>(원희) 6월 3일 오전 피드백 반영을 위한 문서 작업 역할 분배 일정 공유</p>
25.06.03	<p>회의 내용 : 캡스톤 디자인 보고서 수정사항</p> <p>1. 목차 및 폰트 정리 + 전체 내용 점검</p> <p>2. 5p 주요 기능 및 구현 요소 순서 교체</p> <p>1) 대화를 통한 감정 판단</p> <p>2) 감정 변화 추적</p> <p>3) 상황 맞춤 콘텐츠 제공</p> <p>+ 심리지원 정보 알림 서비스 기능을 주요 기능으로 포함</p>

	<p>-> 이걸 5번인데 이걸 1번을 배치</p> <p>+ (2) 자동이라는 단어를 앞에 붙이지 말 것(다른 문구 또는 자동이라는 단어를 중간에 배치)</p> <p>+ (1) 감정 인식 기술을 활용한 정서 분석은 아래에 배치(추후 개발 예정이기때문)</p> <p>3. 30p LLaMA 선택 이유도 기재</p> <p>4. 6p 팀원 역할 새롭게 분배</p> <ul style="list-style-type: none"> - - 팀원 역할 구체화 - 문서 작성 - 발표 준비 - 프론트엔드 개발 - 백엔드 개발 - 모듈별 기능 담당 (감정 추적, 알림, 시각화 등) <p>5. 6p Man-Month 글로 정리하는 것이 아닌 보기 쉽게 정리</p> <p>+ 전일제 라는 단어 삭제 및 문장 새롭게 작성</p> <p>6. 감정분석모델 CSV</p> <ul style="list-style-type: none"> - 감정별 예시 문장 수, 전체 문장 수 등 데이터셋 규모를 명확히 정리 - LLaMA API의 함수 호출 방식과 반환 값 예시 명시 <p>7. 데이터베이스 구조도 넣기</p> <p>8. 30p, 34p - 챗봇 및 LLaMA 설계 부분에서 제거</p> <ul style="list-style-type: none"> - &LLaMA 설계& → &LLaMA API 활용&으로 수정 (보고서 순서 다시 배치) - API는 직접 설계한 것이 아니므로 &활용& 또는 &연동&이라는 용어 사용이 적절 <p>9. 13p 요구사항 명세서</p> <ul style="list-style-type: none"> - 회원관리 외의 다른 기능들(2,3,4,5,1)에 대한 내용 다 기재 <p>10. 13p 기능 명세서</p> <ul style="list-style-type: none"> - 회원관리 외의 다른 기능들(2,3,4,5,1)에 대한 내용 다 기재 <p>-----</p>
--	---

	<p>-----</p> <p>11. 14p 분석활동</p> <ul style="list-style-type: none"> - 가. 회원관리 이외에 다른 기능 및 기술에 대해 기재 <p>12. 16p 현재 로그인 데이터 설계에 맞게 기재</p> <p>(그림 교체 필요시 교체 요망)</p> <p>13. 18p 나. 화면 설계</p> <ul style="list-style-type: none"> - 1,2 번 내용에 추가적으로 다른 화면들 넣기 + 내용 <p>14. 19p 로직 설계</p> <ul style="list-style-type: none"> - 현재기준 회원관리만 존재, 다른 기술도 기재 <p>(그림은 피그마로 작업 요망 + 내용은 회원가입 내용 정리해둔 것 처럼 정리해서 기재)</p> <p>15. 27p 시스템 아키텍처 현재 기준으로 변경된 부분 수정</p> <p>16. 28p 가,나 내용 변경 (현재 기준으로 반영!!- LLaMa 부분 없음)</p> <p>17. 각 기능별 테스트 개요</p> <p>〈확정된 보고서 수정 파트 담당〉</p> <p>빈나 - 1, 3 / 서연 - 2, 4 / 원희 - 5, 6 / 예린 - 7, 8 / 수연 - 9, 10</p>
25.06.04	(원희) 한이음 지피티 승인 알림 + 팀원들에게 SW 정산 양식 공유
25.06.10	<p>회의 내용: 교수님 피드백 사항 공유</p> <p>보고서 수정사항</p> <p>1. 기능 및 역할 관련 부분</p> <p>1) 기능개발 - 백엔드</p> <ul style="list-style-type: none"> - 역할 항목에 명확히 기술 <p>(예: 로그인/DB 설계/감정 분석 API 등)</p> <p>2) 미디어처리 - 감정상태 읽는 미디어, 추후 개발</p> <ul style="list-style-type: none"> - 명시적으로 "미디어처리는 추후 개발 예정"임을 기능 설명란에 표시 <p>3) 역할 수정</p>

	<ul style="list-style-type: none"> - 프로젝트계획서 내 역할 분담표에서 각자 역할을 구체화 <p>2. 오타 수정</p> <ul style="list-style-type: none"> - AWS RDS 오타 수정 - GPT API 오타 수정 <p>3. FastAPI 문서 및 설계 관련</p> <p>1) FastAPI 문서 설명 쉽게</p> <ul style="list-style-type: none"> - 개념, 사용 목적, 동작 예시 등을 쉽게 설명 - 입력 → 분석 → 응답 반환& 순서도 or 그림 추가 <p>2) 연동 방식 예제 추가 + JSON 예시 및 이미지 교체</p> <ul style="list-style-type: none"> - /predict API의 JSON 요청/응답 예시 추가 - 기존 이미지가 있다면 JSON 구조에 맞게 교체 <p>3) 장점 및 특화 기능 표 쉽게 수정</p> <ul style="list-style-type: none"> - 비전공자도 이해할 수 있도록 문장 단순화 - WAS 관련 설명도 함께 쉽게 풀어쓰기 <p>4. 챗봇 파트</p> <p>1) 챗봇 나) 부분에 유니콘 삽입 및 그림 수정</p> <ul style="list-style-type: none"> - 챗봇 흐름도에 &유니콘 캐릭터& 시각 요소 추가 (명시적 요청) <p>2) 챗봇 설계 기술 부연설명</p> <ul style="list-style-type: none"> - KoSmallBERT 사용 이유, LLaMA 역할, EC2 분리 이유 등 기술적 설명 보강 <p>5. 맨먼스 및 개발 일정 시각화</p> <p>1) 맨먼스 부분에 월별 기록 삽입 (표 또는 그래프)</p> <ul style="list-style-type: none"> - 각 월별 개발 분포 표/그래프 추가 (예: 3월~11월 주요 작업 표시) <p>2) 맨먼스 아래에 Actual(실제 진행) 추가</p> <ul style="list-style-type: none"> - Plan vs. Actual 비교 표 또는 꺾은선 그래프 등 시각화
--	--

	<p>6.설계 관련 보강</p> <p>1) FastAPI 설계 단계 추가 및 설명 보강</p> <ul style="list-style-type: none"> - 설계 항목에서 FastAPI 구조도, 설계 이유, 모듈 간 관계 등을 추가 <p>2) JSON 데이터 예시 삽입 (설계)</p> <ul style="list-style-type: none"> - 감정 분석 요청 시 사용하는 JSON 형식 데이터 예시 포함 <p>3) Uvicorn 설명 추가</p> <ul style="list-style-type: none"> - 경량 서버로 선택한 이유 및 배포 시 장점 등 설명 <p>4) 아키텍처 결정 이유 설명 가능하게 보완</p> <ul style="list-style-type: none"> - Firebase+AWS RDS, EC2 이중 구성 이유 등 논리적 설명 추가 <p>〈보고서 수정 역할 분배〉</p> <p>1 - 수연 / 2, 3 - 예린/ 4 - 빈나 / 5 - 원희 / 6 - 서연</p> <p>〈졸업작품 중간 발표회 역할 분배〉</p> <ul style="list-style-type: none"> - 과제 전체 설명 → 원희 - 과제 관련 주요 분석 사항 설명 → 예린 - 과제 기술 구조 설명 → 빈나 - 과제 시연 (직접 데모) → 수연 - 과제 동영상 시연 → 서연
25.06.11	(원희) 졸업작품 중간 발표 피피티 디자인 및 공유파일 링크 팀원들에게 전달
25.06.12	(빈나) 액티비티 통합파일 팀원들에게 공유 (xml + java) / 보고서 공유
25.06.25	(원희) 한이음 중간평가 개발보고서 양식 및 일정 공유
25.06.26	<p>(빈나) 한이음 중간평가 개발보고서 역할 분배 공유</p> <ol style="list-style-type: none"> 1. 요약본, 프로젝트 개요 - 김빈나 2. 프로젝트 구성도, 프로젝트 기능 - 원희

	<p>3. 주요 적용 기술, 프로젝트 개발 환경, 기타 사항 - 정수연</p> <p>4. 프로젝트 수행 내용 - 윤서연</p> <p>5. 기대효과 및 활용분야 - 이예린</p> <p>7월 9일 수요일까지 담당 부분 작성 후 공유하기로 정함</p>
25.06.30	<p>회의 내용 : 멘토님과의 프로젝트 진행상황 공유</p> <p>한이음 중간보고서 작성 중 개발보고서 항목 내 선택사항으로 명시된 &제작설계서&의 작성 방식에 대해 멘토님께 문의드렸다. 멘토님께서는 개발보고서와 제작설계서를 모두 구체적이고 풍부한 내용으로 작성하는 것이 좋으며, 이전에 사전 제출한 보고서를 바탕으로 구성하면 보다 완성도 높은 결과물을 만들 수 있을 것이라고 조언하셨다.</p> <p>이후 현재까지의 개발 진행 상황을 보고드렸다. 앱의 전체 화면(Activity)은 구현을 완료하였으며, 데이터베이스는 Firebase와 AWS RDS 두 가지를 병행하여 사용 중이다. 두 데이터베이스 모두 앱과 연동된 상태이지만, 충분한 테스트는 아직 진행하지 못한 상황이다. 데이터베이스 구조는 회원정보, 로그인, 감정상태기록, 위기사항연락, 추천콘텐츠 등 총 5개의 테이블로 구성할 예정이며, 회원관리 기능(CRUD)은 이미 구현을 완료한 상태다.</p> <p>챗봇 기능은 이중 인스턴스 구조로 설계되었으며, EC2-A 인스턴스는 LLaMA 모델과, EC2-B 인스턴스는 KoSmallBERT 모델과 연동되어 있다. 두 인스턴스 모두 FastAPI를 통해 서비스와 연결되어 있으며, 현재 챗봇 기능과 감정 분석 모델의 API 연동은 아직 완료되지 않았다.</p> <p>감정 분석 모델은 총 70개의 감정 클래스를 기준으로 약 3,500개의 문장 데이터를 사용해 학습을 진행 중이다. 현재 출력 과정에서 문장 오류가 발생하고 있어 에러를 수정하는 작업을 병행하고 있으며, 정확한 감정 분류 결과를 도출하기 위해 지속적으로 개선하고 있다.</p> <p>감정 시각화 기능이 포함된 화면의 UI는 구현이 완료된 상태이나, 실제 시각화 데이터를 효과적으로 표현하는 방식에 대해 난항을 겪고 있어 다양한 대안을 검토하고 있다. 또한, LLaMA 모델을 사용하는 경우에는 성능과 안정성 측면에서 버전 4.0이 적합하다는 조언을 추가로 받았다.</p> <p>전체 개발 완료 목표 시점은 2025년 9월이며, 향후 7월 첫째 주</p>

	또는 둘째 주에 한 차례 더 멘토링을 진행할 예정이다. 다음 멘토링에서는 작성 중인 개발보고서와 제작설계서에 대한 리뷰와 피드백을 받을 계획이다. 이를 통해 최종 산출물의 완성도를 더욱 높이고, 남은 기능들의 구현을 안정적으로 마무리할 예정이다.
25.07.01	(서연) &프로젝트 수행내용& 추가된 한이음 드림업 개발보고서 공유
25.07.02	(원희) 멘토님과 멘토링을 위해 멘토링 가능 날짜와 시간 조율 (원희) AWS RDS에서 과금 발생 알림 + 금액 청구
25.07.03	<p>(빈나) &요약본, 프로젝트 개요& 추가된 한이음 드림업 개발 보고서 공유</p> <p>(빈나) 제작 설계서 내용을 분배하였음</p> <p>기본 구역</p> <ol style="list-style-type: none"> <p>수행 단계별 주요 산출물</p> <p>시장/기술 동향 분석</p> <p>설문조사 분석X</p> <p>인터뷰 결과서X</p> <p>요구사항 정의서</p> <p>유즈케이스 정의서</p> <p>서비스 구성도</p> <p>서비스 흐름도</p> <p>UI/UX 정의서</p> <p>하드웨어/센서 구성도X</p> <p>메뉴 구성도</p> <p>화면 설계서</p> <p>엔티티 관계도</p> <p>설계 단계</p> <p>기능 처리도</p> <p>알고리즘 명세서X</p> <p>알고리즘 상세 설명서X</p> <p>하드웨어 설계도X</p> <p>프로그램 목록</p>

	<ul style="list-style-type: none"> - 테이블 정의서 - 핵심 소스코드 <p>5.</p> <ul style="list-style-type: none"> - 개발 환경 및 설명 - S/W 기능 실사 사진 - H/W 기능 실사 사진X - 동영상 촬영 콘티 - 프로젝트 관리 <p>보고서 작성할 때 있었던 항목은 ,</p> <p>저희 프로젝트에서 해당 없는 항목은 X로 표시</p> <p>(잘 모르는 내용 물음표)</p> <p>1 - 원희</p> <p>2 - 정수연</p> <p>3 - 김빈나</p> <p>4 - 이예린</p> <p>5 - 윤서연</p> <p>(원희) 개발보고서 완료 부분 공유 후 12일까지 제출 바람</p> <p>(원희) 제작설계서에 들어갈 설문조사 설문지를 만들어 공유를 부탁함</p> <p>(서연) 과금으로 인한 DB변경 알림</p> <p>(원희) 설문지 홍보글 확인 + 팀원들에게 인터뷰 내용 전달</p> <p>(원희) 설문조사 참여 인원, 홍보글 진행 현황 알림 + 작성한 제작설계서_1 공유</p> <p>(수연) &주요 적용 기술, 프로젝트 개발 환경, 기타사항& 추가된 한이음 드림업 개발 보고서 공유</p>
25.07.04	<p>(원희) GPT 정산서 보내는 양식 기준 설명</p> <p>(원희) 설문조사 진행현황 + 홍보 + 링크 업데이트 알림</p>
25.07.07	<p>(원희) 제작설계서_2 작성 완료 후 공유 + 업데이트시 뒷번호 바뀌서 올리기 공지</p> <p>(원희) 보고서에 들어갈 느낀점 자정까지 제출 부탁</p>
25.07.08	<p>(원희) 개발 보고서 수정 완료 알림 + 제작 설계서 완료 부탁</p> <p>(예린) 제작설계서_3 업데이트 후 공유</p>

25.07.09	(서연) 제작설계서_4 업데이트 후 공유
25.07.11	(원희) 제작설계서 다음날 오후 11시전까지 완료 부탁 알림
25.07.12	<p>(수연) 제작설계서_5 업데이트 후 공유</p> <p>(원희) 오후 11시까지 제작설계서 6 공유 부탁</p> <p>(빈나) 다음날 줌회의 전까지 내도 되는지 요청</p> <p>(원희) 다음날 줌회의 일정 공유</p>
25.07.13	<p>(빈나) 제작설계서_6 업데이트 후 공유</p> <p>(원희) 최종 업데이트 버전 제작설계서_7 공유</p> <p>[줌회의]</p> <p>07.31 줌 회의 내용</p> <p>중간 보고서(개발보고서, 제작설계서) 작성한것을 검토 받음</p> <p>[개발보고서 피드백]</p> <ul style="list-style-type: none"> - 센서도 사용하는것인지 물어봄 - 전반적으로 잘 작성함 <p>[제작설계서 피드백]</p> <ul style="list-style-type: none"> - 전반적으로 잘 작성함 - 이정도론 큰 문제 없다 - 내일 오전까지 한번 더 보고 피드백 주시기러함 - 필요한 부분도 잘 되어있음 - 기술 설명쪽에 간단히 한문장으로 되어있는데 심사하는 분들이 봤을때 잘 모를 수 있으니 풀어서 설명 적어두기 <p>(원희) 설문하는 부분이 어려웠음</p> <p>(멘토님) 회의 끝나고 회의록 먼저 등록하면 바로 승인 계획</p> <p>(멘토) 요금 ChatGPT만 사용하고 있는데 남은 비용 계산해보고 지원금 받은거 잘 활용하기 조언 + 미리미리 알아보고 하는것이 좋음</p> <p>(원희) 과금이 생겨서 DB변경했던것 알림</p> <p>AWSRDS --> 네온DB</p> <p>(멘토) 처음부터 클라우드로 하는것인지 질문</p>

	<p>(멘토) 하다가 궁금하거나 막히는것 있으면 언제든지 질문 요청</p> <p>(멘토) 다른 변동사항이 없는지 질문</p> <p>(원희) API 연동을 해두지 않아 테스트는해보지 못함</p> <p>(멘토) 클라우드 개발도 중요하지만 설계 먼저 만들어 두고 하면 좋겠다는 조언 + 백엔드가 나서서 하면 좋음</p> <p>[단톡방]</p> <p>(원희) 내일 멘토님의 수정사항 언급이 있다면 파트에 맞게 수정 부탁</p> <p>(원희) 내일 오후 11시에 제출 예정 알림</p>
07.14	<p>(서연) 멘토님 조언으로 수정한 제작설계서_8 공유</p> <p>(원희) 문제점 및 에로사항 의견을 묻고 없음이라고 기재하겠다 함</p>
07.18	<p>(원희) 교수님과의 미팅 일정 조율 + 가능시간 투표</p> <p>(원희) 최종 스케줄 조정 후 목요일 오전 11시에 원희, 윤서연이 교수님과의 미팅에 참여하기로 함</p>
07.23	<p>(원희) 지피티 정산서 요청 청구서, 카드매출전표, SW활용증빙을 PDF로 전환해서 보내달라고 부탁 + 개인톡이나 단톡으로 보내달라고 함</p>
07.24	<p>[캡스톤 교수 미팅]</p> <p>내용</p> <p>7.24 캡스톤</p> <p>- 전공심화: 낮시간에 어떤거 하는지 중요, 취업이랑 전공심화 동시에 준비,</p> <p>주문형 수업 추천해주심(주문형은 절대평가)</p> <p>- 회원명세서, 기능명세서만 있으니 다른 모듈에 관한 정리문서를 넣어야함(다른 기능)</p> <p>- 피그마 파일을 넣어주면 코드를 만들어주는 툴이 있다고 하는데 이걸 사용해보길 권장</p> <p>(사용경험이 중요함 -> 나중에 이런거 사용해봤다고 한줄이라고 기재하기 위해)</p> <p>-> MCP라는 개념이 들어감(MCP 서버 + 클라이언트)</p>

	<p>- 앱 내부 시스템 5개에 관한 그림하나 넣어주기 바람(다이어그램) / M1,M2,M3,M4,M5,M6로 번호 부여 후 설계..??</p> <p>- 피그마의 산출물을 가지고 코드 생성하길 바람(MCP)</p> <p>미션1) 내용추가</p> <p>미션2) MCP 사용하여 피그마산출물을 만들기</p> <p>8/6 2차 미팅 11시</p> <p>(원희) 다음 미팅 참여 가능한 팀원 참석해 달라고 부탁함, 보고서 수정해야하니 역할 분담을 하겠다고 함 + 6월 지피티 정산금이 들 어와서 팀원 계좌로 보내줌</p> <p>(원희)</p> <p>보고서 수정사항 공유</p> <ol style="list-style-type: none"> 1. 회원설계 부분을 제외한 다른 모듈들도 설명 추가 <ol style="list-style-type: none"> 1-1. 감정 분석 및 심리지원 정보 제공 1-2. 감정 변화 추적 및 시각화 1-3. 상황 맞춤 심리 안정 콘텐츠 제공 2. AWS RDS 내용을 Neon DB로 변경 3. AWS RDS가 들어간 그림 수정 4. 데이터셋 내용 변경 -> 3500개에서 2100개로 변경 5. KoBERT를 제외한 다른 언어모델과 비교하여 왜 KoBERT를 선택한건지에 대한 내용 기입 6. 앱 내부 시스템 5개(20페이지 나) <ul style="list-style-type: none"> - 정서 피드백 대화 시스템 - 맞춤형 심리 안정 콘텐츠 제공 - 감정 변화 추적 및 시각화 - 위험감정 대응 시스템 - 감정 인식 기술을 활용한 정서 분석 <p>에서 이 시스템들과 관련된 다이어그램 사진 넣기</p>
--	---

	<p>7. 목차 수정 추가적인 부분) 피그마 파일을 가지고 MCP서버를 활용해서 삽입하면 코드가 짜여지는 툴 활용</p> <p>역할 분배 1 - 1 빈나 1 - 2 원희 2 - 예린 3 - 서연 4 - 수연</p>
07.31	(예린) 설계의 table 부분 제외하고 수정한 중간보고서_28 업로드
08.01	<p>(서연) 11쪽 KoBERT 개요 및 선택 배경 수정, 52쪽 감정별 문장 수 수정 완료한 중간 보고서_29 공유</p> <p>(원희) 기능명세서 추가 완료, 요구사항 명세서가 나오면 보안해서 관련 요구사항 번호 추가예정인 중간보고서_30 공유 / 목차 수정을 위해 5일까지 마감 부탁</p>
08.02	<p>(예린) 중간보고서_31 공유</p> <p>(원희) 8/6일 2차 미팅 참석자 여부 확인</p> <p>(원희) 역할 분배표를 보고 자신이 담당할 부분 개강 전까지 완료할 것 요청함 / 한이음 공모전 출전 여부 고민 부탁</p>
08.04	<p>(원희) 보고서 내일까지 제출 요청</p> <p>(빈나) 중간보고서_32 공유</p> <p>(수연) 중간보고서_33 공유</p>
08.05	(원희) 페이지 잘린 부분 정정, 목차 반영 후 중간보고서_34 공유
08.07	(원희) 25일 교수미팅 알림, 참석가능 여부 확인 + 보고서 200장 분량으로 보완하고 추가할 부분 의견 물어봄
08.12	(원희) 멘토님께 각자 진행한 부분 공유 요청

08.13	(원희) 멘토링 줌 8월20일 예정 알림
08.19	(원희) 교수미팅 시간 변동 알림 (원희) 다음날 멘토링 줌 알림
08.20	(원희) 공모전 콘티 공유 + 제출양식 공유 + 교수님 26일 22시 줌 미팅 확정 알림 (원희) 남은 회의록 담당 정함 〈회의록〉 9월- 수연 10월- 빈나 11월- 서연 (원희) 회의 내용 공유 부탁
08.21	(빈나) 회의 내용 정리 공유 [한이음 공모전 관련] 필수로 들어가야 할 내용 1. 프로젝트명, 팀명 2. 서비스 소개 및 필요성 3. 기존 서비스가 있다면 차별성 4. 시연영상 및 기술 설명 - 공유한 콘티 내용 좋음 - 영상 제작에 1주일만 소요되니 적어도 이번 주 주말부터 제작 들어가야 함 - 2024 이브와 금상 수상한 프로젝트 영상 참고해도 좋을 듯 - 영상 초안 나오면 공유하기 [개발 관련] - 시연 영상을 담으려면 프론트단이 어느 정도 마무리 되어 있어야 함 (완료됨) - 현재 모델 작업 완료 후 연결하면 되는 상태 - 생성형AI 활용: Claude AI 사용 추천해 주심
08.22	원희: SW개발_HW제작설계서와, 개발보고서 업데이트할 내용 공유 원희: 시연영상 두가지 구상 내용 공유 후 안드로이드 개발하는 팀

	<p>원에게 시연영상 준비 부탁 + 챗봇에 라마 적용 후 로컬테스트 마무리 상태 공유</p> <p>원희: SW개발_HW제작설계서와, 개발보고서에 내용 기재 알림 + 의견 부탁</p>
08.24	<p>빈나: 개발보고서, 제작설계서에대한 의견 공유</p> <p>원희: 지피티 정산서 PDF제출 요청</p>
08.25	<p>원희: 지피티 정산된 돈 계좌로 보냄 알림 + 공모전 문서 수정 완료 후 제출 알림</p> <p>원희: 교수님과의 미팅 알림 + 보고서 200장으로 늘리기 위해 어떤 내용을 넣으면 좋을지 물어봄</p>
08.26	<p>원희: 교수미팅 줌 링크 공유</p> <p>빈나:</p> <p>*회의록 -> 페이지 늘리기용 (150~200p 만들기)</p> <p>*전체적으로 앞부분이 너무 무거움(1챕터)</p> <p>〈프로젝트 계획서〉</p> <p>유비콘, Starlette 등 약자 또는 무슨 단어의 조합인지</p> <p>BERT 얘기가 없음 (KoBERT 전에 BERT 한 문단 정도 설명하고 넘어가기)</p> <p>그림들은 분석 챕터에 들어가 있어야 함</p> <p>LLaMA 사용목적을 분명하게 밝히기 (채팅)</p> <p>〈요구사항 명세서〉</p> <p>회원 관리(소셜 로그인을 추가했다는 문구 추가해서 강조하기)</p> <p>요구사항 세분화 필요? -> 나중에 고민 더</p> <p>〈분석 활동〉</p> <p>첫 부분에 있던 설명들 해당 챕터 회원 관리 아래로 내리기 (순서 재조정)</p>

	<p><설계> SQL 가독성 떨어짐 추가적인 것: API 설계(-)JSON 사용한다는 것 보여주기) 아키텍처 부분 재구성 필요 (시스템, 소프트웨어 섞여 있음) Neon DB 왜 썼는지 배경 설명</p> <p>=> 지적사항 수정 => 카드 받아서 출력</p>
08.27	빈나: 현재까지 한 것 진행사항 공유
08.28	원희: 공모전 영상 제작 후 공유
08.29	원희: 공모전 접수 알림
09.02	원희 : ChatGPT 요구사항 정리와 관련 파일을 공유했으며, 미팅 전까지 APK 빌드를 완료해 데모 준비를 요청
09.03	<p>(미팅 후) 보고서랑 APK를 다음 미팅까지 만들어오기</p> <p>빈나 : APK 파일 공유 원희 : 교수님과의 다음 미팅 시간 공지</p>
09.04	<p>원희 : 추계학술대회 논문 공모가 열려 기존 AnOn 내용을 중심으로 작성하는 걸로 해 팀원들의 의견을 물음. 원희 : 논문 파일을 보냄 원희 : 논문 공모전 접수 알림</p>
09.09	<p>원희 : kosmallbert 모델이 지원되지 않아 kcELECTRA-small-v2022로 변경하여 모델 재구동 중임을 공지.</p>
09.10	<p>[캡스톤 교수 미팅]</p> <p>1. 서비스 분석, 개선 전략</p> <ul style="list-style-type: none"> • 역할 분배 : 서비스 개선은 여러 직무에 분산하고, 자율성을 보장하면서 상황에 따라 유동적으로 조정하며 협업을 강화해야 함 • 개선 전략 : 회원가입 유도 시점을 전략적으로 설정하고, 사용자 감정 변화를 추적해 서비스 개선에 활용 • 계획, 평가 : 주기적으로 서비스 현황을 점검, 평가하여 개선 방향을 지속적으로 조정하고 목표 달성을 추구 <p>2. 프로젝트 개발</p> <ul style="list-style-type: none"> • 그래프 활용 : 선호도 조사 결과 파이 그래프가 데이터 제공에

	<p>유리하나, 진행 상황 파악과 메시지 전달에는 막대 그래프를 선호</p> <ul style="list-style-type: none"> • 의사소통 : 데이터 공유 뿐 아니라 의견, 판단 교류가 중요하며, 계획 대비 실적을 시각화해 팀 내 소통을 강화 <p>3. 사용 기술, 동기화 이해</p> <ul style="list-style-type: none"> • 기술 분석 : 사용 기술을 분석해 개발 효율을 높이고 시스템 안정성을 위해 동기화의 필요성을 강조 • 비동기화 : 대용량 처리와 개인화 서비스에 유리하며, 동기화 대비 단순한 프로세스로 유연한 서비스 제공 가능
09.14	<p>원희 : 캡스톤 보고서 역할을 분배</p> <ol style="list-style-type: none"> 1. 분석, 기능 명세서 순서 바꾸기 + 회원관리 파트 라 순으로 (뒤로) + 목차 정리 - 수연 2. 4번 내용 추가 (기타 기능들) + 유비콘 내용 보강 (왜 쓰는지 앞 문단에 정리) - 예린 3. 프로젝트계획서 과제 개요부분 문장구조로 설명추가 + 자살율 그림 작게 + 사용모델(현재 개발한 모델에 대한 서술과 학습활동, 사용 주요 코드(일부분 -> 기능적인 부분) 기재 - 원희 4. 다 주요 기능 및 구현 요소 챗봇 내용 보강 + 맨먼스 그림 막대 그림 변경 (7, 8, 9월 맨먼스 추가해서 새로운 맨먼스 도표첨부) - 빈나 5. API 도입 작성, 코스몰바트 제목 친절하게 변경 (기존 : KoSmallBERT -> KoSmallBERT를 왜 사용하는가) - 서연
09.15	<p>빈나 : 맨먼스 수정 도중 하루 기준 작업 시간을 8시간에서 5시간으로 변경해서 작성</p>
09.16	<p>원희 : 지피티 정산 완료 화면 공유</p>
09.17	<p>서연 : Neon DB 부분 캡스톤 보고서에 내용 추가</p> <p>[캡스톤 교수 미팅]</p> <ol style="list-style-type: none"> 1. 안드로이드 기반 감정 분석 시스템 <ul style="list-style-type: none"> • 모델 필요성: 안드로이드 앱에 감정 분석 기능을 추가하기 위해 자체 모델이 필요하며, 모델을 잘 활용하는 것이 핵심 목표. • 모델 구조: <ul style="list-style-type: none"> • 일렉: 안드로이드 단에서 입력 데이터를 검증·처리하는 로직 • 라마: 서버에서 피드백을 제공하고 모델을 개선하는 로직 • 학습·활용: 데이터 추가 학습을 통해 새로운 모델을 만들고, 라마의 피드백으로 향상시켜 개인화된 서비스 제공. • 운영·관리: 사용자 입력 → 감정 판단 → 피드백 제공 → DB에 개인별 히스토리 저장 → 실시간 피드백 기반 모델 업데이트. 2. 라마 인공지능 모델 <ul style="list-style-type: none"> • 개발 배경: 기존 감정 분석 모델의 한계를 극복하기 위해 새로운 모델(라마) 도입.

	<ul style="list-style-type: none"> 특징: 챗봇 라이브러리로 문장 분석·대화 기능 제공, 고객센터 직원처럼 자연스러운 응대 가능. 개발 과정: <ul style="list-style-type: none"> 라마 로컬 서버 구축 후 데이터셋 2천개에서 3만4천개로 확장, 70가지 감정 단어에 대해 30회씩 문장 구축. 일렉만으로는 데이터 부족으로 학습이 어려워 라마를 도입해 한계를 극복. 의의: 32가지 감정을 분석할 수 있는 모델 개발을 통해 라마 도입이 큰 성과를 이룸. 3. AI 서비스의 활용 <ul style="list-style-type: none"> 서비스 구현: 질문을 통해 대화를 지속시키는 '시동' 기술과 여러 기술을 결합한 '에어캐스팅'을 적용해 자연스러운 대화를 구현. 발표 준비: 한국정보통신학회 'AI 서비스의 활용' 대회에서 10월 17일 발표 예정, GPT 지원과 대표 조언을 통해 라마 서버 구축 및 데이터셋·코딩 구조 중요성 강조. 4. 프로젝트 개발 및 서버 구축 <ul style="list-style-type: none"> 팀 구성: 머신러닝, 딥러닝, 안드로이드 등 분야별 담당자를 지정해 역할을 명확히 분배. 서버 비용: AWS 아마존 서버 사용 시 과금 문제가 있어 중단 및 재연결 계획을 고려. <p>DB 구축: 포스트그래스(PostgreSQL) 기반 네온 DB 사용을 제안. 자동 확장·백업 등 편의 기능을 활용해 효율적 서버 운영 가능.</p>
09.18	<p>원희 : 라마랑 감정 분석 모델을 AWS RDS에 넣는 작업에서 쓸때마다 20달러가 청구되는 문제 발생</p> <p>라마 -> 코랩을 이용하여 API를 호출하는 방식을 사용</p> <p>감정분석모델 -> hugging face를 활용해 테스트까지 완료한 상태</p> <p>hugging face로 API까지 배포가 완료된 상태임을 공유.</p> <p>추가로 안드로이드에 추가할 코드 공유</p> <p>9월 멘토링 일정 정함. 22일 10시</p> <p>경진대회 참가 여부를 물어봄.</p>
09.20	<p>원희 : 경진대회 상세 내용 정보 역할 분배</p> <p>신청개요 : 서연</p> <p>추진배경 및 문제 정의 : 예린</p> <p>데이터 분석 기획 : 빈나</p> <p>데이터 개요 및 데이터 활용방안 : 수연</p> <p>예상성과 및 기대효과 : 원희</p>
09.21	<p>원희 : 경진대회 관련한 회의 일정 정함 -> 9월 24일 10시</p>
09.22	<p>[9월 멘토링]</p> <p>이번 회의에서는 먼저 프로젝트 진행 상황을 공유하였다. 현재 서연용으로 모델 개발은 완료된 상태이며, 모델을 Hugging Face에 API로 배포해 올려둔 상황이다. 또한 Colab을 이용하여 LLaMA 모델도 API로 올려둔 상태이고, 이를 FastAPI에 연동하여 테스트를 진행하고 있다. 테스트가 정상적으로 완료되면 이후에는 토큰 인증</p>

	<p>기능을 구현할 계획임을 설명드렸다.</p> <p>멘토님께서 RedRabbit이라는 GitHub 연동 툴에 대해 설명해 주셨고, Claude라는 코딩에 특화된 AI 도구에 대해서도 소개해 주셨다. 더불어 취업 준비와 관련해 포트폴리오를 다양하게 구성할 것을 조언하시며, AWS 활용 경험, EC2 배포 경험 등을 포함하여 여러 실습을 해보는 것이 좋다는 의견을 주셨다. 삼성 SSAFY와 같은 교육 프로그램도 고려해 보라는 조언도 덧붙였다.</p> <p>향후 계획으로는 FastAPI 테스트를 마친 뒤 토큰 인증 기능을 구현하는 것을 우선적으로 진행하고, RedRabbit 툴과 GitHub 연동 방안을 검토할 예정이다. 또한 Claude와 같은 새로운 AI 도구 학습을 병행하며, 포트폴리오를 보강하기 위해 AWS 및 EC2 배포 경험을 포함한 다양한 실습을 추가적으로 진행할 계획이다.</p> <p>기타 사항으로는 멘토님께서 프로젝트의 기술적 완성도뿐 아니라 취업 준비를 위한 포트폴리오 구성, 클라우드 활용 경험, 교육 프로그램 참여 등 실질적이고 실무적인 측면을 함께 고려하는 것이 중요하다는 점을 강조하셨다.</p>
09.24	<p>[캡스톤 교수 미팅]</p> <ol style="list-style-type: none"> 백엔드 구현과 프로젝트 관리 <ul style="list-style-type: none"> 프로젝트 목표: 웹 서비스의 프론트엔드·백엔드를 함께 구현하고, Node.js와 자바스크립트를 학습하여 감정 데이터 확보 및 문제 해결을 추진. 진행 계획: 데이터산업진흥원 대회와 시험 준비를 병행하며, 감정 분석 모델과 라마 앱을 연동하고, 코랩보다 고급 환경에서 프로젝트를 구축. 발표 준비: 기술 개발 프로세스와 미래 전망을 강조하며 대회·시연 발표 전략을 마련. AI 커뮤니티와 자연어 처리 <ul style="list-style-type: none"> 역할: AI 커뮤니티(Hugging Face 등)는 자연어 처리 알고리즘과 감정 분류 모델을 제공하고, 사용자 입력을 모델에 전달해 API로 배포 가능. API 배포: 텍스트 입력 → 감정 분류 모델 분석 → API를 통해 결과 제공. FastAPI를 사용하고 Uvicorn으로 성능을 보조. 관리·전략: 감정 분류 모델을 개선하고 사용자와 모델 간 연결을 유지·관리하며, 사용자의 요청에 맞춘 질문을 생성. Uvicorn 성능 향상과 바이오메트릭 <ul style="list-style-type: none"> 테스트 API 성능 향상: FastAPI에 Uvicorn을 추가해 속도와 성능을 개선, 이후에도 성능 향상 지속 확인. 바이오메트릭 제안: 사용자 감정 상태를 파악·시각화(파이 그래프 등)해 개인 맞춤형 서비스 제공의 척도로 활용. DB와 파이썬 연동 <ul style="list-style-type: none"> 데이터 저장: 회원 가입 시 감정 표현 데이터를 DB에 저장하고 파이썬으로 시각화.

	<ul style="list-style-type: none"> 아키텍처: 파이어베이스와 네온DB(PostgreSQL)를 검토, 권한·데이터 유형·자동화 제한 등 논리적 기반 강화 필요. <p>연동·시뮬레이션: 파이어베이스 DB와 파이썬 코드를 연동해 시뮬레이션으로 전체 시스템 검증, 안드로이드 클라우드 저장과 UI/UX 개선 병행.</p> <p>서연 : FastAPI서버에서 NeonDB랑 연동. (사용자 입력 후 DB 저장 테스트 완료)</p>
09.25	<p>원희 : 회의 날짜 투표 -> 09월 26일 22시</p>
09.26	<p>데이터경진대회 -> 데이터레시피부문참가 회의하면서 정한내용 모두 이 파일에 작성 된 상태입니다. 어제 참석하지 못한 두분은 필히 읽어봐주세요!</p> <p>논문 PPT,데이터경진대회 서류제출 15일 예정 => 데이터경진대회 서류 작성 파트 분배를 보면</p> <p>신청개요 - 서연 추진배경 및 문제 정의 - 예린 데이터 분석 기획 - 빈나 데이터 개요 및 데이터 활용방안 - 수연 예상성과 및 기대효과 - 원희</p> <p>인데 빈나씨의 분석 기획부분이 마무리 되면 그때 수연씨의 활용방안을 작성해주세요. (원희,예린,서연은 오늘부터 작성 시작)</p> <p>다음주(29~10월 2일)까지 빈나씨 분석 기획부분 마무리예정 ->이후 수연씨 작성(분석 내용을 바탕으로 활용방안 작성해주세요)</p> <p>API연동 및 테스트 10월<추석연휴기간까지>에까지 마무리예정 DB API 연결까지 빈나씨가 하기로 함</p> <p>보고서 증강 계획 - 피그마 디자인 부분 더 넣기(안 넣은 부분들까지 상세히 넣기) - 아이디어 확정과정 및 개선된 내용을 앞부분에 정의 - 기능 넣기 - 성능개선 - 데이터셋증강->추후 가능하면 더 증강예정 - 확장계획도 넣기 => 보고서는 10월 말까지 작성후 끝낼 예정</p> <p>-----</p> <p>날짜별 상세 계획 안내 9월 27일 ~ 30일 원희,서연,예린 서류 완</p>

	<p>9월 30일 논문승인통보 -> 연휴기간동안 서연,예린,원희 PPT 1차완 -> 빈나 API연동 완</p> <p>9월 29일 ~ 10월 3일 빈나 분석기획 완</p> <p>10월 4일 ~ 10월 12일 수연 활용방안 완</p> <p>10월 12일 ~ 14일 데이터경진대회 서류 및 논문 PPT 점검</p> <p>10월 15일 논문PPT 및 데이터경진대회 서류 제출</p> <p>10월 20일 논문발표 사전등록</p> <p>10월 16일 ~ 10월 31일 AnOn캡스톤 보고서완</p> <p>10월 31일 ~ 11월 9일 AnOn 졸작 재점검기간 및 피드백, 시연영상촬영</p> <p>11월 12일(수) 논문 발표회 -> 발표자 김빈나</p> <p>11월 13일 ~ 12월 기말전까지 -> 졸작 발표회 준비(대본 및 PPT)</p> <p>11월 2일 데이터경진대회 서면평가기간</p> <p>11월 25 ~ 26일 데이터경진대회 발표</p> <p>11월 28일 데이터경진대회 최종결과 ->데이터경진대회 시상식은 12월중순예정</p>
09.29	<p>서연 : 데이터경진대회 신청 개요 공유 예린 : 추진 개요 및 문제 정의 공유 원희 : 예상 성과 및 기대효과 공유</p>
09.30	<p>논문 PPT 제작 역할 분배</p> <p>연구배경 및 필요성, 주요 기능, 의의 및 기대효과 - 서연 구현 - 원희 연구 목적, 시스템 개요 - 예린 분석 및 설계, 한계점 및 향후연구방향 - 수연</p>
10.01	원희: 캡스톤 발표 PPT 템플릿 공유
10.02	원희: 캡스톤 발표 PPT 작업링크 공유 및 작성 시작
10.13	<p>빈나: API 연동 작업 지연 알림 - 데모 수요일 오전까지 완료해서 공유할 예정</p>

	<p>- 줌 회의 통해서 테스트 해보고 보안 부분 진행하자는 의견</p> <p>원희: 10월말까지 작업 완료부탁(빈나)</p> <p>- 회의 일정 의견 물어봄</p> <p>- 회의 일정 투표 진행</p> <p>- 회의 링크 공유 및 회의 진행</p> <p>〈회의결과〉</p> <ol style="list-style-type: none"> 1. 홈 화면 음악 표시 2. 오늘의 추천 활동 List 채워넣기 3. 오른쪽 상단 - 검색/왼쪽 상단 - 알림/탭바 두 번째에 하루일기 4. 햄버거 버튼 기능 넣든지 아예 빼든지 5. 명상 콘텐츠, 프로그램 등 (하루일기 같은 건 페이지 넘어가게) 6. 미디어 전송 버튼 삭제 (+ 버튼) 7. 심리 상태표 - 주간/월간 파이차트로 8. 하루일기 9. 탭바를 반응형으로 기기 상관 없이 자연스럽게 보이게 10. 화면 좀 예쁘게
10.14	<p>원희: 교수님의 말씀 전달</p> <p>- 보고서 최신버전 공유하기로 함(학생99폴더)</p> <p>빈나: 수정사항 공유</p> <p>[필수]</p> <ol style="list-style-type: none"> 1. 탭바를 반응형으로 기기 상관 없이 자연스럽게 보이게 2. 심리 상태표 - 주간/월간 파이차트로 (중요) <ul style="list-style-type: none"> - (서연) DB 데이터를 프론트로 전송하는 API 개발 - (수연) 위 API를 연결해 앱에서 시각화 3. 홈 화면 음악/오늘의 추천 활동 List 채워넣기 4. 오른쪽 상단 - 검색/왼쪽 상단 - 알림/탭바 두 번째에 하루일기 5. 미디어 전송 버튼 삭제 (+ 버튼) <p>[화이팅]</p> <ol style="list-style-type: none"> 6. 검색 => 명상 콘텐츠, 프로그램 등 나오게 (하루일기 같은 건 페이지 넘어가게) 7. 하루일기 (날짜 선택하면 이전 일기가 보이게) 8. 화면 좀 예쁘게 (약관, 검색, 일기 목록 이런 거) <p>원희: 논문 PPT 변경 파일 공유</p> <p>빈나: 논문 PPT 폰트 및 수정부분 반영해서 재공유하기로 함</p>
10.15	<p>원희: 캡스톤 PPT 수정사항 공유</p> <p>[캡스톤 ppt 공지]</p> <p>수정사항들을 정리해봤습니다!</p>

	<p>(순서) 표지 - 목차 - 팀이름 및 anon 소개 - 배경 - 설문 및 인터뷰 - 목표 - 앱기동과정(전체데모) - 유즈케이스 - 기능소개(영상 넣은거 빼고 보고서안에 있는 이미지 써서 넣기) - 기술소개(기능소개랑 동일하게 영상제거 후 이미지)+(히스토리) - 데이터셋설명 - 마무리 (ex 기대효과,향후계획)</p> <p>(수정할 부분)</p> <ol style="list-style-type: none"> 1. 순서 변경 2. 목차작성 3. 팀이름 및 안온 의미/사진 넣기 4. 앱기동과정 데모(영상첨부) 5. 유즈케이스 6. 기능소개 내용 보강 및 영상 삭제 후 이미지첨부(히스토리도 넣기 추후제발하기로 했던 생체기능 기획만 하고 실행 못한 이유에 대해서) 7. 기술소개 내용 보강 및 영상 삭제 후 이미지첨부(히스토리도 넣기 KoBERT에서 일렉이 된 이유) 8. 데이터셋 설명 9. 기대효과 및 향후계획 <p>➡1인 2개~3개씩 담당(형평성을 위해 임의로 묶음)</p> <ol style="list-style-type: none"> 1) ① 순서 변경⑥ 기능소개 보강 + 이미지 첨부 (히스토리 포함, 생체 기능 미실행 이유 포함) 2) ② 목차 작성⑦ 기술소개 보강 + 이미지 첨부 (KoBERT 일렉 이유 포함) 3) ③ 팀 이름 및 안온 의미/사진 넣기 ⑤ 유즈케이스 4) ④ 앱기동과정 데모(영상첨부)⑧ 데이터셋 설명⑨ 기대효과 및 향후 계획 <p>마감기한: 10월 20일까지 □디자인은 각자 알아서 - 대충금지□ □폰트는 맑은고딕으로 하기□</p> <p>원회: 오후 10시에 역할 정하기 시작</p> <ol style="list-style-type: none"> 1)수연 ① 순서 변경⑥ 기능소개 보강 + 이미지 첨부 (히스토리 포함, 생체
--	---

	<p>기능 미실행 이유 포함)</p> <p>2)원희 ② 목차 작성⑦ 기술소개 보강 + 이미지 첨부 (KoBERT 일렉 이유 포함)</p> <p>3)예린 ③ 팀 이름 및 안온 의미/사진 넣기 ⑤ 유즈케이스</p> <p>4)서연 ④ 앱기동과정 데모(영상첨부)⑧ 데이터셋 설명⑨ 기대효과 및 향후 계획</p>
10.16	원희: PPT 링크 재공유
10.18	빈나: 논문 PPT 공유
10.20	<p>원희: 일정 확인 및 한이음 결과보고서 양식 공유</p> <p>원희: 한이음 결과보고서 역할 분배</p> <p>1 요약본</p> <p>2 프로젝트 개요 ~ 2.프로젝트 기능 1) 전체 기능 목록</p> <p>3 2) SW 주요 기능 ~ 5. 장비(기자재/재료) 활용</p> <p>4 6. 프로젝트 작동 동영상 ~ 2. 프로젝트 수행일정 표</p> <p>5 3. 프로젝트 추진과정 1) 프로젝트 관리 측면 ~ V. 참고자료</p> <p>1 - 예린 2 - 수연 3 - 원희 4 - 빈나 5 - 서연</p> <p>원희: 캡스톤 PPT, 논문 PPT, 한이음 결과보고서 마감일 알림</p>
10.21	<p>원희: 캡스톤 PPT 수정사항 공유</p> <ul style="list-style-type: none"> - 한이음 결과 보고서 버전2 공유 - 논문 발표시간 공유(트랙 E 2번순서) - 캡스톤디자인 학생성과 경진대회 참가여부 조사 - 캡스톤 경진대회 서류 초안작성해서 공유함
10.22	<p>서연: 한이음 결과 보고서 버전3 공유</p> <p>수연: 한이음 결과 보고서 버전4 공유</p> <p>원희: 캡스톤 경진대회 서류 초안 피드백 물어봄</p>

	<ul style="list-style-type: none"> - 캡스톤 경진대회 서류 제출 - 보고서 수정사항부분 공유 <보고서 증강 계획> <ol style="list-style-type: none"> 1) 피그마 디자인 부분 더 넣기(안 넣은 부분들까지 상세히 넣기) 2) 아이디어 확정과정 및 개선된 내용을 앞부분에 정의 3) 기능 넣기 4) 성능개선 5) 데이터셋증강->추후 가능하면 더 증강예정 6) 확장계획도 넣기 7) 회의 내용 추가 <p>빈나: 한이음 결과보고서 버전5 공유 원희: 캡스톤 경진대회 서류 제출 완료 알림</p>
10.27	<p>멘토: 한이음 결과보고서 확인, 피드백 사항 없음을 공지 원희: 한이음 결과보고서 업로드</p> <ul style="list-style-type: none"> - 한이음 만족도조사 팀원에게 알림 - 설문 다 완료되어서 최종승인 멘토님께 알림 <p>멘토: 최종 승인</p>
11.06	<p>원희: 캡스톤 PPT 수정사항 공유</p> <ol style="list-style-type: none"> 1) 발표 전 실제 단말기에서 실행되는 앱을 교수님께 제공 2) 발표가 루즈하지 않도록 데모 세션 구성 -> 주요 기능이 강조된 시연 영상을 발표 흐름에 맞춰 배치 3) 슬라이드 속 텍스트를 간결하게 정리 4) 초기 2,000개의 규모에서 34,000개의 규모로 증강된 과정 설명 5) 모델 정확도와 F1-score는 외부 통계에 따른 예상 수치임으로 슬라이드 수정 6) 눈 깜빡임 기능은 '미완성' 표현 대신 '추후 개발 및 확장 계획'으로 수정
11.12	<p>팀원들 모두 발표 장소에 집결하여 발표를 보고 난 후 해산</p> <p>원희: 캡스톤 PPT 피드백 반영하여 수정 후 공유</p> <ol style="list-style-type: none"> 1) 슬라이드 속 텍스트 최대한 덜어냄 2) 모델 정확도와 F1-score 아래 부연설명 추가 3) 17일 이후 앱 기동 영상 새로 첨부할 예정

* 2025.11.12. 회의록 기준으로 작성됨