# 빅데이터기반분산

중간고사


빅데이터기반분산
**1**분반
소프트웨어학과
**32204041**
정다훈

**1. AI** 서비스를 설계할 때 **‘**기술 기반 사고**’**와 **‘**문제 해결 중심 사고**’**는 서로 다른 출발점을 가진다**.** 두 접근 방식의 차이를 설명하고**, AI** 서비스 기획 과정에서 두 관점을 어떻게 통합해야 사용자에게 실질적인 가치를 제공할 수 있는지 서술하시오**.**

AI 서비스 기획 시 흔히 마주치는 두 가지 접근 방식은 기술 기반 사고와 문제 해결 중심 사고이다. 기술 기반 사고는 최신 기술의 기능적 가능성을 중심으로 새로운 서비스를 기획하려는 경향이 있으며, 주로 연구개발 단계에서 나타난다. 예를 들어, 특정 자연어 처리 기술이나 대규모 언어 모델이 도입되었을 때, 이를 어떤 서비스에 적용할 수 있을지를 먼저 탐색하는 방식이다. 이 접근은 기술 혁신과 선도적 시도에 유리하지만, 사용자 요구와의 불일치로 이어질 가능성도 내포한다.

반면, 문제 해결 중심 사고는 사용자의 불편, 니즈, 업무 흐름 등의 실제 문제를 출발점으로 하여, 해당 문제를 해결할 수 있는 기술적 방안을 탐색하는 방식이다. 이 경우 기술은 수단이며, 문제 해결이 목적이 된다. 결과적으로 사용자 만족도와 서비스 활용도가 높아질 수 있지만, 기술적 제약이나 구현 가능성의 한계로 인해 아이디어가 실현되지 못할 위험도 존재한다.

AI 서비스가 성공적으로 구현되기 위해서는 이 두 사고방식의 유기적 통합이 필요하다. 초기 기획 단계에서는 문제 해결 중심 접근을 통해 주요 사용자 페인포인트를 정밀하게 정의해야 하며, 이후 해당 문제를 해결할 수 있는 기술 요소를 기술 기반 사고를 통해 도입하는 것이 바람직하다. 특히 프로토타입 개발 및 반복 검증 과정에서는 문제 정의와 기술 도입 사이의 간극을 지속적으로 조정하여 사용성과 기술 가능성이 균형을 이루도록 해야 한다. 이러한 통합적 접근은 기술적 참신성과 사용자 중심 설계를 동시에 만족시키는 고품질 AI 서비스의 기획과 구현을 가능하게 한다.

**2.** 다양한 **AI** 프로젝트에서 발생하는 실패 사례 중 하나는 **‘**모델 성능은 우수하나**,** 사용자 요구와 괴리된 서비스**’**이다**.** 이러한 문제가 발생하는 이유를 설명하고**,** 이를 방지하기 위해 수행해야 하는 사용자 니즈 분석**,** 페르소나 설정**,** 사용자 스토리 작성 등의 절차와 중요성에 대해 서술하시오**.**

많은 AI 프로젝트에서는 모델의 정량적 성능이 매우 우수함에도 불구하고 실제 사용자로부터 외면받는 경우가 발생한다. 이는 모델이 해결하고자 하는 문제의 맥락, 사용 환경, 사용자 기대 등과 일치하지 않는 방식으로 설계되었기 때문이다. 기술 개발에 집중하는 과정에서 사용자의 실질적인 요구, 서비스 접점에서의 UX 요소, 도입 환경에서의 맥락이 충분히 반영되지 않은 결과이다.

이러한 문제를 방지하기 위해서는 기획 초기 단계에서 철저한 사용자 기반 분석이 선행되어야 한다. 우선 사용자 니즈 분석을 통해 다양한 사용자군의 문제 상황과 필요 조건을 파악해야 하며, 이를 위해 인터뷰, 설문, 사용 로그 분석 등의 정성·정량적 조사 기법이 활용된다. 이후 대표적인 사용자 유형을 페르소나로 구체화함으로써 기획 방향의 초점을 명확히 할 수 있으며, 다양한 유형의 사용자 행동과 목표를 가상 시나리오로 구성하여 모델 활용 흐름을 시각화할 수 있다.

사용자 스토리는 “사용자 A는 언제, 어떤 목적을 가지고 서비스를 어떻게 사용하는가”를 문장 형식으로 기술하는 방식으로, 실제 사용 맥락과 기능 요구사항을 연결하는 역할을 한다. 이러한 과정을 통해 단순히 성능이 높은 모델이 아니라, 실질적으로 사용자의 삶에 도움을 줄 수 있는 서비스로 기획 방향을 정립할 수 있으며, 나아가 AI 모델 개발 방향성과 평가 기준도 사용자 중심으로 조정될 수 있다.

**3. AI** 모델을 서비스에 적용할 때는 높은 성능만큼 운영 비용도 중요한 고려사항이다**. AI** 서비스 기획 과정에서 성능과 비용의 균형을 고려하여 모델 구조**,** 인프라 선택**, API** 활용 등을 결정할 때 고려해야 할 요소들을 제시하고**,** 그에 따른 **trade-off** 사례를 설명하시오**.**

AI 서비스를 실환경에 적용하는 과정에서는 단순히 모델의 정밀도나 예측 정확도뿐만 아니라, 그에 수반되는 인프라 비용, 응답 시간, 유지보수 효율성 등 다양한 운영적 요소들을 함께 고려해야 한다. 특히 모델이 커지고 복잡해질수록 필요한 연산 자원과 처리 시간이 늘어나기 때문에, 서비스 기획 단계에서 성능과 비용 사이의 균형점을 사전에 정의하는 것이 필수적이다.

모델 구조 측면에서는 목적에 따라 경량화 모델을 도입하거나 양자화 및 지연 허용 구조를 선택할 수 있으며, 이로 인해 GPU 활용 효율이 달라진다. 인프라 측면에서는 고성능 GPU 인스턴스(A100 등)를 사용할지, 상대적으로 저렴한 리소스(T4 등)를 활용할지에 따라 비용 차이가 크게 발생한다. 또한 외부 AI API를 사용하는 경우 단기간에는 비용이 적고 빠르게 적용 가능하나, 장기적으로는 트래픽 증가에 따라 비용 부담이 커질 수 있다. 이 외에도 API 호출 빈도에 따라 캐싱 전략을 병행하는 것이 운영 효율을 높이는 방법이 될 수 있다.

실제 사례로, 한 OCR 서비스에서는 GPT 기반 Vision API를 통해 매우 높은 정확도를 달성했으나, API 호출당 비용이 높아 전체 운영비용이 급증하는 문제가 있었다. 이를 해결하기 위해 Tesseract 기반의 경량 OCR 엔진을 기본값으로 설정하고, 신뢰도가 낮은 입력에 대해서만 고성능 API를 사용하는 방식으로 운영 전략을 변경한 결과, 전체 비용을 약 80% 절감하면서도 품질 수준을 유지할 수 있었다.

또 다른 사례로, 챗봇 서비스를 운영하는 과정에서 LLaMA3 13B 모델을 활용하던 기존 구성은 응답 품질은 우수했으나 24시간 운영에 과도한 GPU 인스턴스 비용이 소모되었다. 이에 따라 LLaMA3 2B 모델을 파인튜닝하여 다수의 사용자 유형에 따라 응답을 미리 캐싱하고, 반복되는 질의는 사전 응답을 제공하는 구조로 전환함으로써 응답 속도와 품질을 유지하면서 비용을 효과적으로 절감하였다.

이러한 사례들은 고정된 기준이 아닌, 상황과 목표에 따라 유연하게 구성 요소를 조합하여 성능과 비용의 균형을 확보해야 함을 시사한다. 이를 통해 지속 가능한 AI 서비스 운영이 가능해진다.

## **4.** 자신이 기획한 서비스를 구현하기 위해 필요한 **End-to-End** 기술 스택**(**프론트엔드**,** 백엔드**, AI/ML,** 인프라 등**)**을 계층별로 구성하고**,** 각 스택을 선택한 이유를 설명하시오**. (UI, API** 서버**,** 모델 학습 환경**,** 서빙**/**배포 플랫폼**,** 데이터 저장 방식 등 포함**) (10**점**)**

제가 기획한 서비스는 사용자의 자연어 기반 질의를 바탕으로 맞춤형 식당을 추천하고, 신뢰도 있는 리뷰 요약 정보를 제공하는 AI 기반 맛집 추천 챗봇입니다. 해당 서비스를 실제로 구현하기 위해, UI, API 서버, AI/ML 모델 학습 환경, 모델 서빙/배포 플랫폼, 데이터 저장 방식 등 전체 구성 요소를 포괄하는 End-to-End 기술 스택을 계층별로 구성하였으며, 각 기술 선택의 이유를 아래와 같이 서술합니다.

**1.** 사용자 인터페이스 **(UI** 계층**)**

- 기술 스택 **:** React
- 선택 이유 **:** React는 단일 페이지 애플리케이션(SPA) 구조를 기반으로 하여, 사용자 발화와 챗봇 응답 간의 상호작용을 빠르고 자연스럽게 처리할 수 있습니다. 챗봇 대화창, 식당 추천 목록, 필터 조건 입력창 등 주요 UI 요소를 컴포넌트 단위로 나누어 구성함으로써 유지보수가 용이하며, 향후 기능 확장에도 유연하게 대응할 수 있습니다. 이러한 특성은 실시간 피드백이 중요한 대화형 인터페이스 설계에 적합합니다.

**2. API** 서버 **(**백엔드 계층**)**

- 기술 스택 **:** Flask (Python)
- 선택 이유 **:** Flask는 가볍고 유연한 Python 기반 웹 프레임워크로, 자연어 처리 모델과의 연동에 매우 적합합니다. 프론트엔드에서 입력된 데이터를 AI 모델에 전달하고, 추론 결과를 다시 클라이언트에 반환하는 과정을 간결한 코드 구조로 구현할 수 있습니다. 또한 Python 생태계와의 높은 호환성을 바탕으로 모델 로딩, 텍스트 전처리, 로그 기록 등 다양한 처리를 통합하여 구성할 수 있어, 백엔드와 AI 모델 간 연결을 매끄럽게 수행할 수 있습니다.

**3. AI/ML** 모델 학습 및 실험 환경

- 기술 스택 **:** PyTorch, HuggingFace Transformers, LLaMA3 2B, Google Colab Pro / 로컬 GPU 서버
- 선택 이유 **:** 기획안에서 설정한 핵심 기능인 의도 분류(intent classification), 슬롯 추출(slot filling), 리뷰 요약(text summarization)을 구현하기 위해, 사전 학습된 LLaMA3 2B 모델을 기반으로 파인튜닝을

수행합니다. PyTorch는 연구 친화적이고 구조가 유연하여 복잡한 자연어 처리 모델 구현과 디버깅에 적합합니다. HuggingFace Transformers는 LLaMA3 모델뿐만 아니라 다양한 토크나이저 및 학습 도구를 통합적으로 제공하며, 학습-서빙 전환을 원활히 할 수 있도록 지원합니다. 실험 단계에서는 Google Colab Pro를 통해 GPU 환경에서 초기 검증을 수행하고, 학습 데이터가 확장되거나 고정형 모델 운영이 필요한 경우 로컬 GPU 서버로 이관하여 자원 활용도를 최적화합니다.

**4.** 모델 서빙 및 배포 플랫폼

- 기술 스택 **:** Docker, AWS (EC2)
- 선택 이유 **:** 학습된 모델은 Docker 컨테이너를 활용하여 일관된 실행 환경에서 배포하며, Flask API 서버와 함께 통합 서빙됩니다. Docker는 운영 체제나 시스템 설정의 차이와 관계없이 동일한 환경을 유지할 수 있어, 실 서비스 배포 시 안정성을 높여줍니다. 서버 운영은 AWS EC2를 활용하여 구성하며, 초기에는 단일 인스턴스로 운영하다가, 트래픽 증가 시 Auto Scaling 또는 Load Balancer를 도입할 수 있어 확장성 측면에서도 유리합니다. 클라우드 기반 배포는 서비스 가용성을 높이고, 관리 편의성을 확보할 수 있다는 장점이 있습니다.

**5.** 데이터 저장 및 로그 관리

- 기술 스택 **:** MongoDB
- 선택 이유 **:** MongoDB는 JSON 형태의 비정형 데이터를 유연하게 저장할 수 있는 NoSQL 데이터베이스로, 챗봇 사용자의 대화 내역, 발화 의도, 추천 결과, 필터링 조건 등 다양한 데이터를 구조화하여 저장할 수 있습니다. 스키마가 유동적이기 때문에 새로운 기능이 추가되거나 데이터 구조가 변화하더라도 별도의 마이그레이션 없이 유연하게 확장 가능합니다. 또한 빠른 검색과 필터링이 가능하여 대화 로그 분석 및 사용자 피드백 기반 서비스 개선에도 효과적으로 활용할 수 있습니다.

이와 같은 기술 스택 구성은 기획안에서 정의한 핵심 기능(자연어 기반 식당 탐색, 리뷰 요약, 필터 조건 입력 등)을 안정적으로 구현하고, 실사용자 환경에서의 성능과 확장성, 유지보수 가능성을 모두 고려한 전략적 선택입니다. 각 계층은 독립적이면서도 유기적으로 연결되어 있어, 서비스 고도화나 유스케이스 확장 시에도 유연하게 대응할 수 있습니다.

## **5.** 자신의 서비스에서 핵심적인 **AI** 기능을 수행할 모델 **1**가지를 선정하고**,** 이 모델을 구현하고 학습시키기 위해 필요한 계획을 서술하시오**. (**모델 종류**,** 학습 목적**,** 학습 데이터의 특징과 전처리**,** 학습 방식**,** 성능 검증 기준 및 테스트 전략 등 포함**) (10**점**)**

기획 단계에서 기획한 서비스는 사용자의 자연어 기반 질의에 따라 신뢰도 높은 식당을 추천하고, 관련 리뷰 정보를 요약하여 제공하는 AI 챗봇입니다. 이러한 기능을 구현하기 위해, 단순 생성형 언어모델이 아닌 RAG(Retrieval-Augmented Generation) 구조를 기반으로 챗봇을 설계하고자 하며, 이 구조에서 핵심적인 역할을 수행할 생성 모델을 선정하고, 이에 대한 학습 및 구현 계획을 아래와 같이 제시합니다.

**1.** 모델 선정 및 학습 목적

본 서비스에서는 사용자 질의에 대해 신뢰도 높은 식당 추천과 설명력 있는 리뷰 요약 응답을 생성하기 위한 핵심 AI 기술로, RAG(Retrieval-Augmented Generation) 구조를 채택합니다. 이 구조 내에서 생성기 역할(Generator) 을 수행할 언어모델로는 Meta의 LLaMA3 2B를 선정합니다.

- LLaMA3 2B는 약 20억 개의 파라미터를 가진 Transformer 기반의 경량 사전학습 언어모델입니다.
- 상대적으로 적은 연산 자원으로도 문맥 이해와 자연어 생성 성능이 우수한 것이 특징입니다.

본 모델은 사용자의 질의(Query)와 검색기(Retriever)가 반환한 관련 식당 정보 및 리뷰 텍스트를 통합하여 입력받고, 그에 대한 자연어 응답을 생성하는 방식으로 학습됩니다.

이때 학습 목적은 단순 문장 생성이 아닌, 조건 기반 맞춤 추천 + 리뷰 요약 정보를 실용적이고 신뢰성 있게 제공하는 모델로 학습시키는 것에 있습니다.

**2.** 학습 데이터 구성 및 전처리 계획

학습 데이터는 다음과 같은 세 가지 구성 요소로 준비됩니다:

- 사용자 질의(Query) 와 그에 대응하는 자연스러운 정답 응답(Response) 으로 구성된 QA 쌍

- 해당 질의에 관련된 외부 문서(Context): 식당 설명, 사용자 리뷰, 태그, 위치 정보 등 포함
- 응답에는 조건 기반 추천, 설명, 리뷰 요약 등의 정보가 포함되도록 구성

전처리 절차:

- HTML 기반 문서나 비정형 리뷰 텍스트는 문장 단위로 정제하고 문단으로 구분
- 문서를 256~512 token 단위로 chunking 처리하여 검색 가능 단위로 구성
- 각 chunk를 FAISS 기반 벡터 데이터베이스에 임베딩 및 저장

**3.** 학습 방식 및 환경

- 학습 프레임워크: PyTorch + HuggingFace Transformers
- 학습 방식: Supervised Fine-tuning(SFT) 방식으로 질의(Query) + 문서(Context)를 입력으로, 이상적인 응답 생성
- Prompt 설계: Instruction 구조 기반
- 출력 제어: EOS 토큰 기반 종료 / Temperature 조정으로 응답 다양성 확보

초기 하이퍼파라미터 설정**:**

- 학습률: 2e-5
- 배치 사이즈: 8 (gradient accumulation = 2로 가상 16 구성)
- 에폭 수: 5
- warm-up 비율: 10%

학습 환경 구성**:**

- 1단계: Google Colab Pro (T4 GPU)
- 2단계: 데이터 확장 시 로컬 GPU 환경으로 전환

**4.** 성능 평가 기준 및 테스트 전략

정량 지표 평가**:**

- ROUGE-L: 문장 구조 유사도
- BLEU: 어휘 일치도
- BERTScore: 의미적 유사도

정성 평가 **(Human Evaluation):**

- 평가 기준: 정확성, 자연스러움, 신뢰도 반영 여부 (5점 척도)

테스트 전략**:**

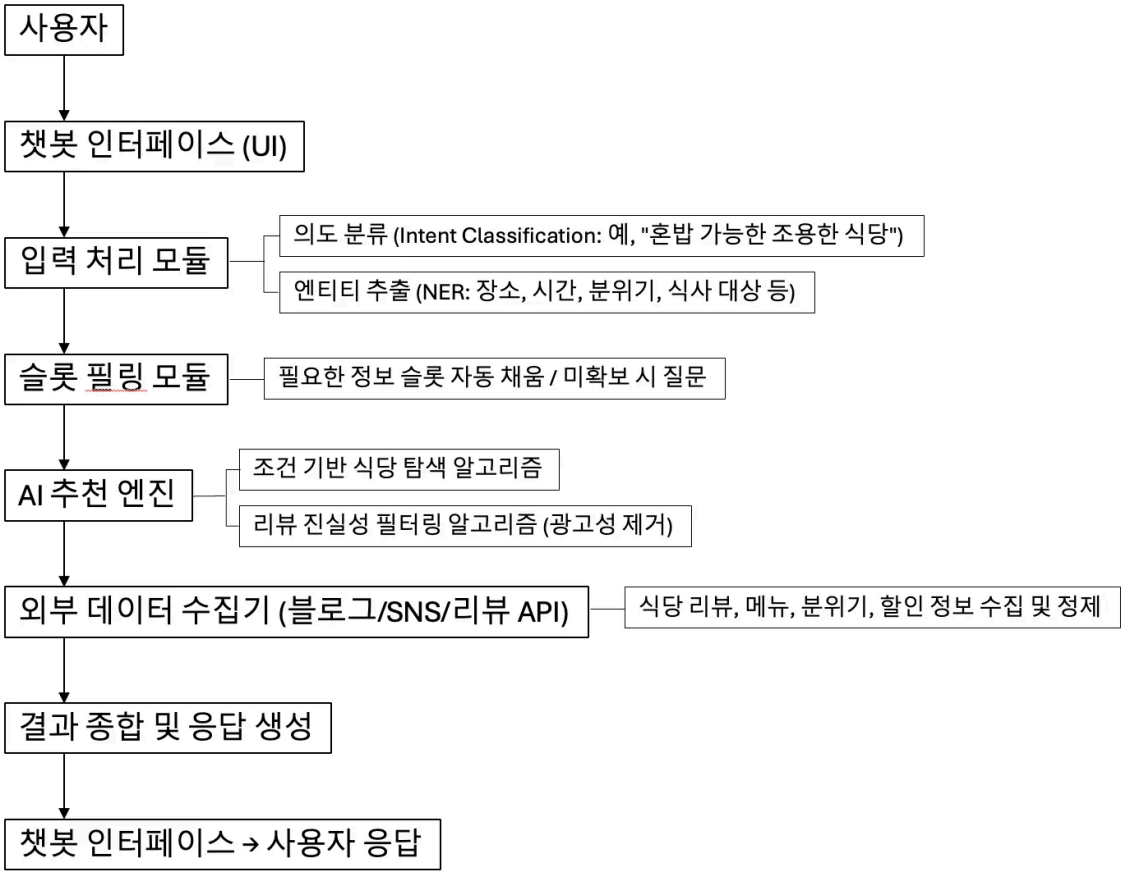
- 데이터셋 8:1:1 비율로 학습/검증/테스트 분할
- 검색기 성능 평가: precision@5
- 생성기 성능 평가: BLEU, ROUGE, Human 평가 병행
- Blind Test: 실제 사용자 질의 기반 응답 평가 시나리오 수행

**5.** 기대 효과 및 향후 계획

본 계획을 통해 구현되는 RAG 기반 생성 모델은 기존의 단순 LLM 응답보다 더 신뢰할 수 있는 정보 기반 응답 생성을 가능하게 합니다. 또한 LLaMA3 2B의 경량성과 문맥 처리 성능을 결합하여, 실제 서비스 환경에서도 빠르고 유의미한 추천 응답을 생성할 수 있습니다.

향후에는 검색기의 정밀도를 향상시키기 위한 Query Rewriting 기법 적용과, 리뷰 신뢰도 기반 필터링 기능을 추가한 문서 선택 전략을 고도화할 예정입니다.

**6.** 자신의 **AI** 모델이 실제 사용자 요청을 처리하는 시스템 아키텍처를 아키텍처 구성도와 워크플로우를 포함하여 설명하시오**. (10**점**)**




1. 사용자의 자연어 입력 수신

   사용자는 챗봇 인터페이스를 통해 일상 언어 형식으로 요청을 입력한다. 예를 들어, “성수동에서 연인이랑 조용하게 저녁 먹을 곳”과 같은 표현이 이에 해당한다. 이때 입력 문장은 구조화되지 않은 자연어로 구성되어 있으며, 이후 단계에서 이를 기계적으로 이해 가능한 형태로 처리해야 한다.

2. 의도 분석 및 정보 슬롯 추출

   입력된 문장은 자연어 처리 모델에 의해 분석되며, 사용자가 원하는 행위의 목적(인텐트)을 분류하고, 문장 내 포함된 주요 정보 항목(슬롯)을 추출한다. 예시에서는 인텐트는 ‘맛집 탐색’으로 분류되며, ‘장소=성수동’, ‘식사 상대=연인’, ‘분위기=조용한’, ‘식사 시간대=저녁’과 같은 슬롯이 추출된다.

3. 정보 보완 질의 수행

   초기 입력 문장에 필요한 슬롯 정보가 일부 누락된 경우, 챗봇은 추가 질문을 통해 누락된 정보를 수집한다. 예를 들어, 가격대 정보가 명시되지 않았을 경우, “예산은 어느 정도인가요?“와 같은 보완 질문을 사용자에게 제시하게 된다.

4. 식당 탐색 및 필터링 수행

   수집된 모든 슬롯 정보를 바탕으로, 추천 엔진은 내부 또는 외부 데이터베이스를 참조하여 해당 조건에 부합하는 식당 목록을 탐색한다. 이 과정에서는 위치, 메뉴, 분위기, 가격대 등의 다중 조건을 동시에 고려한 필터링 알고리즘이 적용된다.

5. 리뷰 기반 식당 신뢰도 분석 및 요약

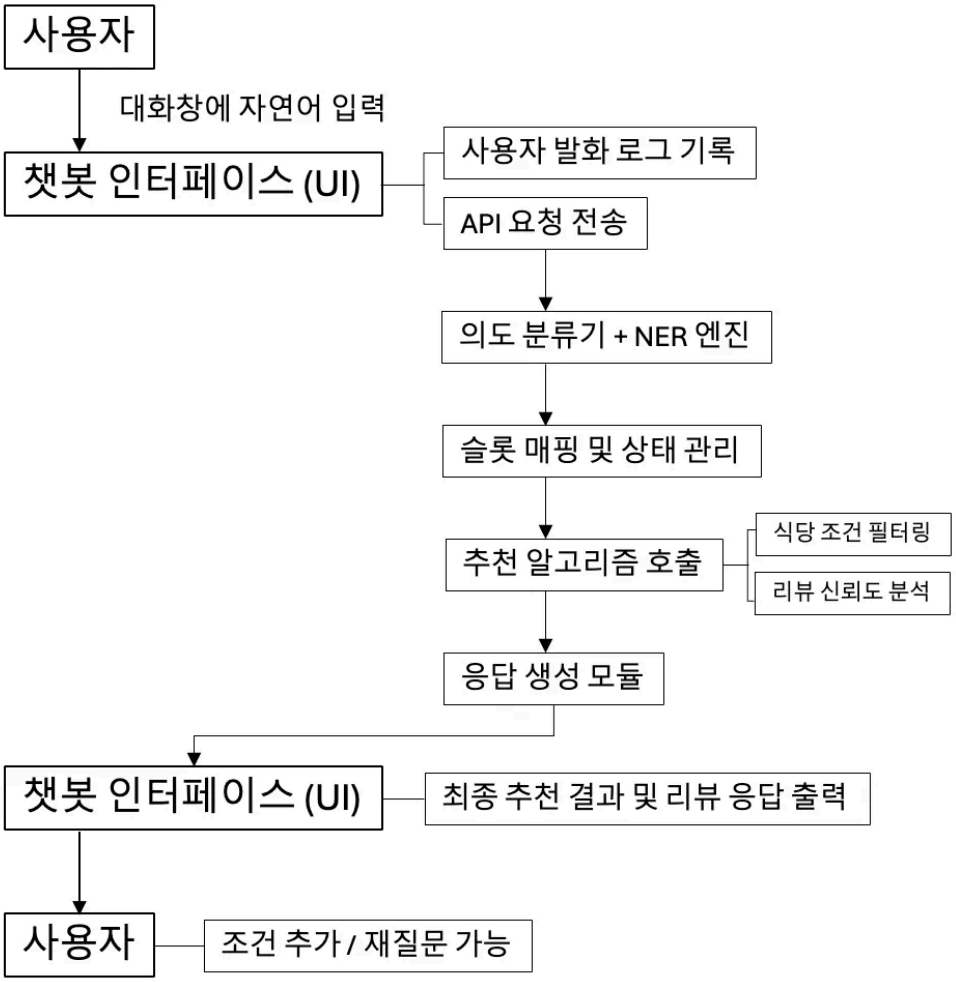
   후보 식당이 선정되면, 각 식당에 대한 사용자 리뷰 데이터를 수집하고 분석하여, 광고성 또는 비정상 리뷰를 제외한 신뢰도 높은 리뷰만 선별한다. 이후, 긍정/부정 키워드 중심으로 요약된 평가 결과를 구성한다.

6. 최종 응답 생성 및 사용자 전달

   모든 분석이 완료된 이후, 챗봇은 추천 식당 정보와 리뷰 요약 내용을 통합하여 최종 응답을 생성한다. 응답은 자연스러운 언어로 구성되어 사용자에게 전달되며, 선택된 식당의 위치, 분위기, 운영 시간 등 다양한 요소가 함께 제공된다.

이와 같은 단계적 절차를 통해, 사용자는 단일 질의를 기반으로 신뢰도 높은 맞춤형 음식점 추천을 제공받게 된다. 본 워크플로우는 AI 기반 대화형 서비스의 실질적 구현 사례로, 자연어 이해와 사용자 맞춤형 추천 기술이 통합된 응답 시스템의 표준적인 구조로 간주될 수 있다.

**7.** 자신의 서비스에서 사용자 인터페이스**(UI)**가 **AI** 결과와 어떻게 연동되는지 흐름도와 시나리오를 이용해서 설명하시오**. (10**점**)**




1. 사용자 입력

   사용자는 “야식으로 라멘 먹을 수 있는 식당 추천해줘”라는 자연어 문장을 입력하였다. 본 발화는 ‘라멘’이라는 특정 메뉴와 ‘야식’이라는 시간대를 명시함으로써, 두 개의 주요 정보 슬롯을 동시에 포함하고 있는 복합 질의에 해당한다.

2. 시스템 내부 처리 흐름

   AI 챗봇 시스템은 입력된 문장을 먼저 인텐트 분류 모델을 통해 ‘맛집 탐색’ 인텐트로 분류한다. 이후 슬롯 추출 과정을 통해 ‘메뉴=라멘’과 ‘식사 시간대=야식’ 정보를 추출한다. 이 과정에서 두 개의 세부 인텐트인 메뉴 키워드 탐색 인텐트와 시간대 기반 탐색 인텐트가 동시에 활성화된다.

챗봇은 이를 바탕으로 내부 음식점 데이터베이스 또는 외부 API를 활용하여 조건에 부합하는 식당을 탐색한다. 탐색된 후보 식당에 대해 리뷰 필터링 및 요약 알고리즘이 적용되며, 광고성 리뷰 제거, 긍정·부정 키워드 기반의 정제된 평가 결과가 함께 도출된다.

3. 챗봇 응답 예시

응답 결과는 자연어 형식으로 사용자에게 제공되며, 추천 식당의 위치, 운영 시간, 식사 분위기, 대표 리뷰 요약 정보가 포함된다. 예를 들어, 챗봇은 “이 식당은 성수동에 위치한 라멘 전문점으로, 야식으로 운영하며 혼밥하기에도 좋다”는 설명과 함께, “국물이 진하고 매장이 조용하다”, “청결 상태가 좋음” 등 핵심 리뷰를 함께 제시한다.

4. 후속 사용자 발화 및 시스템 반응

사용자가 이후 “좀 더 저렴한 곳으로 바꿔줘”라고 요청하는 경우, 챗봇은 이를 ‘조건 수정 및 재탐색 인텐트’로 인식한다. 이때 기존 조건 중 ‘가격대’ 슬롯에 대한 수정 요청으로 판단되며, 나머지 기존 조건인 메뉴, 시간대, 분위기는 유지된다. 챗봇은 수정된 조건을 반영하여 재탐색을 수행하고, 새로운 추천 응답을 생성하여 사용자에게 제공한다.

5. 결론

본 시나리오는 복합 질의를 효과적으로 처리하는 AI 챗봇의 실용적 구조와 자연어 기반 대화 흐름의 유연한 전환 능력을 보여준다. 특히 다중 슬롯 기반의 인텐트 분류와 조건 보존·수정 대응 메커니즘은 사용자의 실시간 피드백을 반영한 동적 추천 서비스를 구현하는 데 있어 핵심적인 요소로 작용한다. 이와 같은 구조는 사용자 중심의 맞춤형 AI 추천 시스템 설계에 있어 모범적인 사례로 간주될 수 있다.