MVP 로드맵

'Fair AI' MVP 개발 로드맵 (AI 엔진 중심)

Phase 0: 기획 및 수동 프로토타이핑 (1주차)

코드를 짜기 전에, AI에게 시킬 일을 사람이 먼저 직접 해보며 '정답'을 정의하는 가장 중요한 단계입니다.

1. **MVP 범위 확정:**
    - **분석 대상 언론사 선정:** 정치/사회/경제 각 분야에서 성향별로 대표성을 띠는 언론사를 20~30개 내외로 확정합니다. (예: 조선/중앙/동아, 한겨레/경향/오마이뉴스, 한국/서울경제, YTN/연합뉴스 등)
    - **언론사 성향 수동 매핑:** 선정한 언론사들을 **스프레드시트**에 정리하고, **'보수', '진보', '중도', '경제지', '페미니즘' 등** 자체적인 기준에 따라 성향을 수동으로 태깅합니다. 이것이 초기 AI의 판단 기준이 됩니다.
2. **수동 분석 및 프롬프트 설계:**
    - **하나의 '핫이슈'를 선정**합니다. (예: '금리 인상' 또는 특정 법안)
    - 선정한 언론사들에서 해당 이슈에 대한 기사를 5~6개 직접 스크랩합니다.
    - **사람이 직접** 다음 작업을 수행합니다.
        - 각 기사에서 '팩트'와 '관점(의견)'을 형광펜으로 구분해 봅니다.
        - 교차 검증되는 팩트들을 모아 담백한 '중립 요약문'을 작성해 봅니다.
        - 보수/진보 진영별로 기사를 묶어 "보수 진영의 핵심 주장은 OOO이다" 와 같이 요약해 봅니다.
    - ➡️ **결과물:** 이 수동 분석 과정 자체가 **나중에 LLM에게 시킬 명령어(프롬프트)의 가장 완벽한 설계도**가 됩니다.
3. **기술 스택 결정:**
    - **LLM API 선정:** 한국어 성능, 비용, 속도를 고려하여 주력으로 사용할 LLM API를 선택합니다. (Google Gemini, OpenAI GPT, Anthropic Claude, Naver HyperCLOVA X 등)

Phase 1: 데이터 파이프라인 자동화 (2주차)

AI가 분석할 '재료'를 자동으로 수집하고 정리하는 단계입니다.

1. **동일 이슈 기사 자동 클러스터링:**
    - **방법:** 선정된 언론사들의 RSS 피드나 포털 뉴스 페이지를 주기적으로 스크래핑합니다. 수집된 기사들의 제목을 **Embedding API(문장을 벡터로 변환하는 기술)**를 사용해 벡터로 변환한 뒤, **코사인 유사도**를 계산하여 비슷한 기사들을 하나의 그룹으로 묶습니다.
    - **기술:** Python, `BeautifulSoup`, `scikit-learn` 또는 벡터 DB(Pinecone, ChromaDB) 활용.
2. **전체 기사 본문 스크래핑 및 저장:**
    - 클러스터링된 기사 그룹이 생성되면, 해당 URL에 접속하여 기사 본문 전체를 스크래핑합니다.
    - *데이터베이스(DB)**에 '이슈 ID', '언론사', '기사 제목', 'URL', '본문 내용', '언론사 성향' 등을 구조화하여 저장합니다.

Phase 2: 핵심 AI 분석 엔진 개발 (3~4주차)

이 부분이 가장 핵심이며, **LLM 프롬프트 체이닝(Prompt Chaining)** 기술이 중요합니다. 하나의 복잡한 작업을 여러 개의 단순한 작업으로 나누어 LLM에게 순차적으로 시키는 방식입니다.

1. **1단계: 개별 기사 분석 (Fact/View 추출)**
    - **입력:** DB에 저장된 기사 본문 1개.
    - **프롬프트 예시:**

> "너는 편향되지 않은 미디어 분석가야. 다음 기사를 분석해서 아래 형식에 맞춰 JSON으로 출력해 줘.
>
> 1. `objective_facts`: 기사 내용 중 육하원칙에 따른 객관적인 사실들을 불렛포인트로 요약.
> 2. `perspectives`: 기자의 논조, 특정 인물의 발언, 수사적 표현 등 관점이나 의견이 담긴 문장들을 불렛포인트로 요약.
> 3. `claims_to_verify`: 통계 수치, 법률 해석 등 외부 검증이 필요한 핵심 주장 2가지.
>
> ---
>
> [기사 본문 내용]"

    - **결과:** 각 기사별로 '사실', '관점', '검증필요 주장'이 구조화된 데이터로 추출됩니다.

2. **2단계: 교차 분석 및 중립 팩트 기사 생성**
    - **입력:** 특정 이슈로 묶인 모든 기사들의 `objective_facts` 데이터.
    - **프롬프트 예시:**

      > "다음은 동일한 사건에 대한 여러 언론사의 팩트 요약 리스트야. 이 내용을 종합해서, 중복되거나 교차 검증되는 사실들을 중심으로 제3자의 시점에서 작성된 하나의 완벽하고 중립적인 스트레이트 기사를 작성해 줘. 감정적 표현이나 확인되지 않은 추측은 절대 포함하지 마."

    - **결과: 'Fair AI'의 핵심인 중립적인 종합 기사**가 생성됩니다.
3. **3단계: 관점별 요약 생성 (AllSides 방식 구현)**
    - **입력:** Phase 0에서 수동 매핑한 언론사 성향과 각 기사의 `perspectives` 데이터.
    - **프롬프트 예시:**

      > "다음은 '보수' 성향으로 분류된 언론사들의 기사에서 추출한 관점들이다. 이 내용을 바탕으로, 이번 이슈에 대한 보수 진영의 핵심 주장과 논리, 주요 우려 사항을 3문장으로 요약해 줘."

    - **결과:** "보수 진영에서는 어떻게 평가할까요", "진보 진영의 입장은?"과 같은 요약 콘텐츠가 생성됩니다.

Phase 3: 프론트엔드 연동 및 MVP 런칭 (5주차~)

AI 엔진이 만든 결과를 사용자가 볼 수 있도록 웹사이트에 연결합니다.

1. **백엔드 API 개발:** '바이브 코딩'으로 제작된 프론트엔드가 호출할 수 있는 API를 만듭니다. (예: `GET /api/issues/latest`)
2. **결과 캐싱(Caching): LLM API 호출은 비용이 비싸고 시간이 오래 걸립니다.** 따라서 한 번 분석한 이슈 결과는 반드시 **캐시(Cache) 서버(예: Redis)**에 저장하여, 새로운 사용자가 요청할 때 빠르게 응답하고 불필요한 API 호출을 막아야 합니다.
3. **UI/UX 구현:**
    - 상단: AI가 생성한 '중립 팩트 기사'를 배치.
    - 하단: '보수', '진보', '중도' 등 칼럼을 나누고, 각 칼럼에는 '관점별 요약'과 원본 기사 링크 목록을 제공.
4. **MVP 런칭 및 피드백:** 소규모 그룹(친구, 지인, 관련 커뮤니티)을 대상으로 서비스를 공개하고, AI가 생성한 기사의 '중립성'과 '정확성'에 대한 피드백을 집중적으로 수집하여 개선합니다.

Fair AI의 핵심 : “AI 기자”

주기적이고 빠른 정보 수집 자동화는 이 서비스의 심장과도 같습니다. 직접 발로 뛰는 기자가 없는 상황에서 기술적으로 어떻게 이를 구현할 수 있을지, 구체적인 **자동화 봇 시스템 설계 방안**을 단계별로 설명해 드리겠습니다.

전체 시스템은 **'수집 → 군집화 → 분석 → 제공'**의 4단계 파이프라인으로 구성됩니다.

자동화 봇 시스템 설계 방안

1단계: 데이터 수집 봇 (Collector Bot)

이 봇의 유일한 임무는 정해진 시간마다 세상에 나온 새로운 뉴스 기사들을 빠짐없이 긁어모으는 것입니다.

- **작동 주기:** 서버에서 **스케줄러(Scheduler)**를 이용해 10~15분 간격으로 자동 실행되도록 설정합니다. (기술: `cron` on Linux, `APScheduler` in Python)
- **수집 대상:**
  1. **RSS 피드 (주력):** 대부분의 언론사는 RSS 피드를 공식적으로 제공합니다. 이는 가장 안정적이고 '착한' 수집 방법입니다. RSS에는 보통 기사 제목, 요약, 원문 링크가 포함되어 있습니다.
  2. **포털 뉴스 (보조):** 네이버 뉴스, 다음 뉴스의 정치/사회/경제 섹션의 '최신 기사 목록' 페이지를 주기적으로 크롤링합니다. RSS에서 놓치는 기사들을 보완할 수 있습니다.
- **수집 방식:**
  - 봇이 실행되면, 수집 대상 목록을 순회하며 새로운 기사 링크들을 수집합니다.
  - 데이터베이스에 이미 저장된 링크인지 확인하여, 새로운 기사일 경우에만 다음 단계로 넘깁니다.
  - **결과물:** 새로운 뉴스 기사들의 URL 리스트

2단계: 이슈 군집화 봇 (Clustering Bot)

수집된 수십 개의 기사들을 '어떤 사건'에 대한 것인지 자동으로 묶어주는, 이 시스템의 핵심 기술 중 하나입니다.

- **작동 방식:** 수집 봇이 새로운 기사 리스트를 전달하면 즉시 작동합니다.
- **핵심 기술: 문장 임베딩 (Sentence Embedding)**
    1. 새로 수집된 기사의 **'제목'**을 AI 임베딩 모델(예: `ko-sbert`)을 사용하여 **수학적인 벡터(좌표 값)**로 변환합니다. 문장의 의미가 벡터 공간의 특정 좌표로 표현되는 것입니다.
    2. 이 벡터 값을 데이터베이스에 저장된 최근 1~2시간 내의 다른 기사 벡터들과 **코사인 유사도**를 계산하여 비교합니다.
    3. 벡터 공간에서 서로 가까이 위치한(유사도가 높은) 기사들을 **'동일 이슈 그룹(Cluster)'**으로 묶어줍니다.
- **판단 로직:**
    - **새로운 이슈 발생:** 기존 그룹과 유사도가 낮은 새로운 기사 그룹이 형성되면, '새로운 이슈 ID'를 부여하고 DB에 저장합니다.
    - **기존 이슈에 추가:** 새로운 기사가 기존에 생성된 이슈 그룹과 유사도가 높으면, 해당 그룹에 기사를 추가하고 '최종 업데이트 시간'을 갱신합니다.
- **결과물:** 동일한 사건에 대한 여러 언론사의 기사들이 하나의 묶음으로 정리된 '이슈 클러스터'

3단계: AI 분석 봇 (Analysis Bot)

군집화된 이슈를 바탕으로, 우리가 논의했던 핵심적인 LLM 분석을 수행합니다. **비용과 시간이 가장 많이 드는 단계이므로, 효율적으로 작동시키는 것이 중요합니다.**

- **작동 조건 (트리거):**
    - 새로운 이슈 클러스터가 생성되고, **최소 3개 이상의 언론사 기사**가 모였을 때.
    - 기존 이슈 클러스터에 새로운 기사가 추가되고, **마지막 분석 시간으로부터 1시간 이상** 지났을 때.
- **작동 방식 (프롬프트 체이닝):**
    1. 트리거가 발동되면, 해당 이슈 클러스터에 속한 모든 기사의 본문을 DB에서 불러옵니다.
    2. **(1차 LLM 호출)** 각 기사를 LLM에게 보내 '팩트'와 '관점'을 분리하여 구조화된 데이터(JSON)로 추출합니다.
    3. **(2차 LLM 호출)** 1차에서 추출된 모든 '팩트' 데이터를 모아 LLM에게 보내, 교차 검증된 내용만을 바탕으로 **'중립 요약 기사'**를 생성합니다.

4. **(3차 LLM 호출)** 1차에서 추출된 '관점' 데이터와 미리 매핑해 둔 언론사 성향을 조합하여, **'보수 진영의 시각', '진보 진영의 시각'** 등 관점별 요약문을 생성합니다.

- **결과물:** '중립 요약 기사', '관점별 요약문', '관련 기사 링크 목록' 등 웹사이트에 보여줄 최종 콘텐츠가 DB에 저장됩니다.

아키텍처 요약

(위 이미지는 개념을 설명하기 위한 예시입니다.)

[스케줄러] -> **[수집 봇]** (RSS/포털 크롤링) -> **[군집화 봇]** (문장 임베딩) -> **[AI 분석 봇]** (LLM API 호출) -> **[데이터베이스]** <- **[사용자 웹사이트]**

이러한 자동화된 파이프라인을 구축하면, 직접 기사를 쓰지 않고도 주기적으로 국내 주요 이슈에 대한 깊이 있는 분석 콘텐츠를 **'반자동'**으로 생성하고 사용자에게 제공할 수 있습니다. 초기에는 봇이 잘 작동하는지, AI가 이상한 결과를 만들지는 않는지 지속적인 모니터링이 필요하지만, 시스템이 안정화되면 최소한의 인력으로 운영이 가능해질 것입니다.