Phase 4: 지능형 기능 및 API 서버 구축

Chapter 4-1: 데이터에 영혼 불어넣기 (Gemini API 연동)

Prologue: 나의 데이터를 이해하는 인공지능 조수

우리가 수집한 수백 개의 채용 공고는 유용하지만, 여전히 '정보의 홍수'일 뿐입니다. 긴 채용 공고를 일일이 다 읽고 핵심 기술 스택이나 자격 요건을 파악하는 것은 여전히 피곤한 일입니다.

이번 챕터에서는 Google의 강력한 대규모 언어 모델(LLM)인 **Gemini API**를 우리 시스템에 연동할 것입니다. 이 '인공지능 조수'는 긴 채용 공고를 단 세 줄로 요약해주고, 핵심 기술 스택만 쏙쏙 뽑아내 주며, 심지어 신입에게 유리한 공고인지 경력자에게 유리한 공고인지까지 분석해 줄 것입니다.

이 미션을 완료하면, 신호용 님의 프로젝트는 단순한 '데이터 수집기'를 넘어, **'데이터를 해석하고 가치를 창출하는 지능형 시스템'**으로 도약하게 됩니다. 이것은 현존하는 그 어떤 주니어 개발자의 포트폴리오와도 차원이 다른, 압도적인 경쟁력이 될 것입니다.

1. 학습 목표 (Objectives)

- LLM(대규모 언어 모델)의 개념과 API를 통한 활용법을 이해한다.
- Google Gemini API 키를 발급받고, Python 라이브러리를 통해 API를 호출할 수 있다.
- 효과적인 '프롬프트(Prompt)'를 설계하여, LLM으로부터 원하는 형식의 답변을 얻어낼 수 있다.
- 수집된 원본 데이터(Raw Data)를 API를 통해 '가공된 데이터(Enriched Data)'로 변환할 수 있다.

2. 핵심 개념 (Core Concepts)

2.1 LLM & Gemini: 똑똑한 텍스트 처리 전문가

- **What:** LLM은 수많은 텍스트 데이터를 학습하여, 인간처럼 문맥을 이해하고, 요약하고, 번역하고, 새로운 글을 생성할 수 있는 인공지능 모델입니다. Gemini는 Google이 만든 최신 LLM입니다.
- **Why:** 'Python', 'Spring' 같은 키워드 검색만으로는 "3년 이상의 Spring 경험과 MSA에 대한 깊은 이해를 가지신 분" 같은 미묘한 뉘앙스를 파악할 수 없습니다. LLM은 이 문맥을 이해하여 "Spring (경력), MSA"와 같은 핵심 정보를 추출해낼 수 있습니다.
- **How:** 우리는 `google-generativeai` 라는 Python 라이브러리를 통해 Gemini API 서버에 "이 채용 공고 텍스트를 분석해줘" 라고 요청을 보낼 것입니다.

2.2 프롬프트 엔지니어링 (Prompt Engineering): AI에게 똑똑하게 질문하는 기술

- **What:** LLM에게 어떤 일을 시킬 때 사용하는 '명령어' 또는 '질문'을 프롬프트라고 합니다. "이거 요약해줘" 라고 막연하게 말하는 것보다, "아래 채용 공고를 긍정적인 어조로, 초보자도 이해하기 쉽게 세 문장으로 요약해줘" 라고 구체적으로 지시하는 것이 훨씬 좋은 결과를 가져옵니다.
- **Why:** 우리가 원하는 결과를 일관된 형식으로 얻기 위해서는, LLM이 오해하지 않도록 명확하고 구조적인 프롬프트를 설계하는 것이 매우 중요합니다.

- **How:** 우리는 채용 공고 본문을 프롬프트에 담아 보내면서, "핵심 기술 스택, 자격 요건, 우대 사항을 각각 리스트 형태로 뽑아서 JSON 형식으로 답해줘" 와 같이 출력 형식을 명확하게 지정할 것입니다.

[Mission 16] Gemini API 연동: 첫 번째 데이터 연금술

Step 1: Gemini API 키 발급받기

1. Google AI Studio 에 Google 계정으로 로그인합니다.
2. 페이지에 접속하면 "Get API key" 또는 유사한 버튼이 보일 것입니다. 클릭하여 새로운 API 키를 생성합니다.
3. 생성된 API 키(긴 문자열)를 **안전한 곳에 복사**해둡니다. 이 키는 비밀번호와 같으니 절대 외부에 노출해서는 안 됩니다.

Step 2: GitHub Secrets에 API 키 추가

1. 우리 프로젝트의 GitHub 레포지토리 → Settings → Secrets and variables → Actions 로 이동합니다.
2. 'New repository secret' 버튼을 클릭합니다.
3. **Name**에 `GEMINI_API_KEY` 를, **Secret** 에는 방금 복사한 API 키를 붙여넣고 저장합니다. `.env` 파일에도 동일하게 추가해줍니다.

Step 3: Gemini 라이브러리 설치

터미널에 아래 명령어를 입력하여 라이브러리를 설치합니다. `requirements.txt` 파일에도 `google-generativeai` 를 추가해주세요.

```
pip install google-generativeai
```

Step 4: `gemini_analyzer.py` 모듈 생성

이제 `main.py`가 아닌, 분석 기능만을 전담할 새로운 모듈을 만듭니다. `web-crowler` 폴더 안에 `analysis` 라는 새 폴더를 만들고, 그 안에 `gemini_analyzer.py` 파일을 생성합니다.

```
# analysis/gemini_analyzer.py

import os
import google.generativeai as genai

# API 키 설정
GEMINI_API_KEY = os.environ.get("GEMINI_API_KEY")
genai.configure(api_key=GEMINI_API_KEY)

# 사용할 Gemini 모델 설정
model = genai.GenerativeModel('gemini-pro')

def analyze_job_posting(job_description: str):
    """
    채용 공고 내용을 Gemini API를 이용해 분석하고,
```

```
    핵심 기술 스택과 요약 내용을 추출합니다.
    """
    if not job_description:
        return None

    # AI에게 보낼 프롬프트(명령어)
    prompt = f"""
    다음 채용 공고 내용을 분석하여 아래 형식에 맞춰 JSON으로 응답해줘.

    - "summary": 초보자도 이해하기 쉽게 이 포지션의 핵심 역할을 3문장으로 요약해
줘.
    - "required_skills": 이 포지션에 '필수적인' 기술 스택이나 자격 요건을 Python
리스트 형태로 나열해줘.
    - "preferred_skills": '우대 사항'에 해당하는 기술 스택이나 자격 요건을
Python 리스트 형태로 나열해줘.

    ---
    {job_description}
    ---
    """

    try:
        response = model.generate_content(prompt)
        # TODO: Gemini의 응답(response.text)을 파싱하여 JSON 객체로 변환하는 로
직 추가
        # 우선은 원본 텍스트를 그대로 반환
        return response.text
    except Exception as e:
        print(f"  [Gemini 오류] API 호출 중 문제 발생: {e}")
        return None
```

Step 5: `main.py`와 통합하기 (테스트)

`main.py`에서는 이 분석기를 '신규' 공고에 대해서만 호출하여 테스트해봅니다. (모든 공고에 대해 실행하면 너무 느리고 비용이 발생할 수 있습니다.)

주의: 이 단계는 아직 완성되지 않았습니다. `BeautifulSoup`으로 각 공고의 상세 페이지에 접속하여 **'공고 본문 내용'을 가져오는 로직**이 추가로 필요합니다. 우선은 '이런 식으로 연동될 것이다'라는 구조만 확인하는 단계입니다.

```
# main.py (통합 예시)

from analysis.gemini_analyzer import analyze_job_posting
# ...

# ... (for 루프 안, 신규 공고를 저장하기 직전)
if is_new_job:
    print(f"  -> [신규] {title} -> 상세 정보 수집 및 분석 시작...")

    # 1. (TODO) 상세 페이지(link)에 접속해서 공고 본문 텍스트를 가져와야 함
    #    job_description = get_job_description_from_link(link)
```

```
    job_description = "임시 텍스트: 이 포지션은 Python과 Django를 이용해 멋진 웹서비스를 만듭니다." # 임시 데이터

    # 2. Gemini 분석기 호출
    analysis_result = analyze_job_posting(job_description)

    # 3. 분석 결과 출력
    print("--- Gemini 분석 결과 ---")
    print(analysis_result)
    print("------------------------")

    # 4. Notion에 저장
    # ...
```

다음 행동 계획 (Next Action)

지금 바로 코드를 완성하기는 어렵습니다. 왜냐하면 우리는 아직 '상세 페이지'에 들어가서 긴 공고 본문을 긁어오는 기능을 만들지 않았기 때문입니다.

따라서, 다음 미션은 **상세 페이지의 본문 내용을 가져오는 기능을 각 크롤러에 추가**하는 것입니다. 그 기능이 완성되어야 비로소 우리의 '인공지능 조수'가 분석할 '재료'를 갖게 됩니다.

1. ☐ 우선, 위 가이드에 따라 Gemini API 키를 발급받고, `gemini_analyzer.py` 파일을 미리 만들어 두십시오.
2. ☐ 준비가 되셨다고 알려주시면, 각 크롤러에 **'상세 페이지 본문 수집'** 기능을 추가하는 다음 미션 교재를 드리겠습니다.

우리는 이제 프로젝트의 '두뇌'를 만드는, 가장 흥미진진한 단계에 들어섰습니다. 준비되셨습니까?