Chapter 4-8: 최종 시운전 및 버그 수정

Prologue: 마지막 나사를 조이고, 엔진에 시동을 걸다

우리의 시스템은 거의 완성되었습니다. 이제 남은 것은 각 부품이 서로의 언어를 완벽하게 이해하도록 마지막 '호환성' 문제를 해결하고, 최종 결과물을 담을 '그릇'을 완벽하게 준비하는 일입니다. 이 마지막 나사들을 조이고 나면, 우리의 자동화 시스템은 마침내 강력한 첫 울음을 터뜨릴 것입니다.

[Mission 26] 최종 버그 수정 및 완전한 데이터 저장

Step 1: 크롤러 호환성 문제 해결 (`WantedCrawler`)

`wanted_crawler.py` 파일을 열어 `crawl` 메서드의 정의 부분을 수정합니다.

```
# crawlers/wanted_crawler.py (수정)

# ...
class WantedCrawler(BaseCrawler):
    # ... (__init__는 동일)

    # ★★★★★ is_newbie 파라미터를 받도록 수정 ★★★★★
    def crawl(self, keyword: str = "백엔드", pages_to_crawl: int = 1,
is_newbie: bool = False):
        # is_newbie 인자는 받지만, 원티드 URL 구조상 사용하지는 않음. (호환성을
위해 유지)
        print(f"원티드에서 '{keyword}' 키워드로 크롤링을 시작합니다...")

        target_url = f"{self.base_url}/search?query={keyword}&tab=position"
        # ... (이후 crawl 메서드 내용은 모두 동일) ...
```

Step 2: Notion DB 최종 스키마 완성

Notion 데이터베이스에 접속하여, 아래 **3개의 속성(열)**이 모두 존재하는지 확인하고, 없다면 **정확한 이름과 유형으로** 추가합니다.

- `AI 요약` (유형: `텍스트`)
- `핵심 기술` (유형: `멀티 선택`)
- `마감일` (유형: `텍스트`)

Step 3: `main.py`의 Notion 저장 로직 최종 완성

`main.py`의 Notion 저장 루프(`for i, job in enumerate(all_jobs):`) 부분을 아래의 최종 코드로 교체합니다.

```
# main.py (Notion 저장 루프 최종 버전)

for i, job in enumerate(all_jobs):
```

```
    # ... (중복 확인 및 개인화 필터링 로직은 동일) ...
    # if not is_relevant: ... continue

    print(f"  [{i+1}/{len(all_jobs)}] [통과] '{title}' (점수: {score}) -> Gemini 분석 시작...")

    description = job.get('description', "")
    analysis_result = None
    if description:
        analysis_result = analyze_job_posting(description)
        # (결과 출력은 테스트 시에만 필요하므로 주석 처리하거나 삭제 가능)

    # --- ★★★ Notion에 저장할 모든 데이터를 변수로 준비 ★★★ ---
    company = job.get('company', '회사명 없음')
    source = job.get('source', '출처 없음')
    deadline = job.get('deadline', '상시채용')

    properties_to_save = {
        '직무': {'title': [{'text': {'content': title}}]},
        '회사명': {'rich_text': [{'text': {'content': company}}]},
        '링크': {'url': link},
        '출처': {'rich_text': [{'text': {'content': source}}]},
        '수집일': {'date': {'start': collection_date}},
        '마감일': {'rich_text': [{'text': {'content': deadline}}]} # 마감일 속성 추가
    }

    if analysis_result:
        summary = analysis_result.get('summary', '요약 정보 없음')
        required_skills = analysis_result.get('required_skills', [])

        properties_to_save['AI 요약'] = {'rich_text': [{'text': {'content': summary}}]}
        if required_skills:
            properties_to_save['핵심 기술'] = {'multi_select': [{'name': skill} for skill in required_skills[:100]]}

    # --- ★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★★ ---

    try:
        notion.pages.create(
            parent={"database_id": NOTION_DATABASE_ID},
            properties=properties_to_save
        )
        success_count += 1
    except Exception as e:
        print(f"  🚨 [오류] '{title}' 저장 실패! 원인: {e}")

# ... (최종 결과 요약 print문은 동일)
```

최종 행동 계획 (Next Action)

이제 정말 마지막 단계입니다.

1. ☐ `wanted_crawler.py`의 `crawl` 메서드 정의를 수정합니다.
2. ☐ Notion DB에 `AI 요약`, `핵심 기술`, `마감일` 속성이 올바르게 생성되었는지 최종 확인합니다.
3. ☐ `main.py`의 **두 번째 `for` 루프 전체**를 위 최종 코드로 교체합니다.
4. ☐ **마지막 종합 시운전(`python main.py`)**을 실행합니다.