🎯 FlowNote - AI 기반 PARA 분류 시스템

> **당신의 문서를 AI가 자동으로 분류합니다**
> PARA 방식 (Projects, Areas, Resources, Archives)으로 스마트하게 정리

📖 목차

---

1. 📖 프로젝트 소개

FlowNote는 AI 기반 문서 자동 분류 시스템입니다. 사용자의 직업과 관심 영역을 학습하여, 업로드된 문서를 PARA 방식으로 지능적으로 분류합니다.

💡 핵심 아이디어

```
사용자 온보딩 (직업, 관심 영역)
    ↓
GPT-4o가 사용자 맥락 학습
    ↓
파일 업로드 (PDF, TXT, MD)
    ↓
AI가 PARA 방식으로 자동 분류
    ↓
분류 결과 + 신뢰도 + 키워드 제시
    ↓
🎉 완료!
```

예시:

- **입력**: "프로젝트 제안서_2025.pdf"
- **결과**: 📋 Projects (신뢰도 95%)
- **키워드**: 프로젝트, 마감일, 목표

---

2. ✨ 핵심 기능

2.1 🚀 스마트 온보딩

- GPT-4o가 사용자 직업 분석
- 10개 영역 추천 → 5개 선택
- 사용자 맥락 저장 및 활용

2.2 📄 AI 기반 자동 분류

```
📋 Projects (프로젝트)
   → 기한/마감일이 있는 구체적 목표
   예: "11월까지 대시보드 구현"

🎯 Areas (분야)
   → 지속적 책임 영역
   예: "팀 성과 관리 지속 업무"

📚 Resources (자료)
   → 참고용 정보/학습 자료
   예: "Python 최적화 가이드"

📦 Archives (보관)
   → 완료된 프로젝트 보관
   예: "2024년 프로젝트 결과"
```

2.3 🔍 키워드 검색 시스템

- FAISS 벡터 검색 엔진
- OpenAI Embeddings 활용
- 실시간 유사도 점수 표시
- 검색 히스토리 저장

2.4 📊 실시간 대시보드

- PARA 분류 통계 시각화
- 파일 트리 구조 표시
- 최근 활동 로그
- 메타데이터 관리

2.5 🎯 맥락 기반 분류

- 사용자 직업/관심 영역 반영
- 신뢰도 점수 (0-100%)
- 키워드 태그 자동 생성
- 분류 근거 설명

---

3. 💻 기술 스택

3.1 Backend

기술	버전	용도
Python	3.11.10	개발 언어
FastAPI	0.120.4	REST API 프레임워크
LangChain	1.0.2	AI 체인 관리
SQLite	3	메타데이터 저장
Uvicorn	0.38.0	ASGI 서버

3.2 Frontend

기술	버전	용도
Streamlit	1.51.0	웹 UI 프레임워크
Plotly	6.3.1	데이터 시각화
Pandas	2.3.3	데이터 처리
st-aggrid	1.1.9	테이블 렌더링

3.3 LLM & AI

기술	모델	용도
OpenAI API	GPT-4o	PARA 분류, 영역 추천
OpenAI API	GPT-4o-mini	경량 작업
OpenAI Embeddings	text-embedding-3-small	벡터 임베딩

3.4 검색 & 데이터

기술	버전	용도
FAISS	1.12.0	벡터 검색 엔진
pdfplumber	0.11.0	PDF 파싱
python-dotenv	1.1.1	환경변수 관리

---

4. 📁 프로젝트 구조

```
flownote-mvp/
│
├── requirements.txt                    # Python 의존성
├── .env.example                        # 환경변수 예시
├── .gitignore
│
├── backend/                            # FastAPI 백엔드
│   ├── main.py                         # FastAPI 메인 앱
│   ├── embedding.py                    # 임베딩 생성
│   ├── chunking.py                     # 텍스트 청킹
│   ├── faiss_search.py                 # FAISS 검색
│   ├── metadata.py                     # 메타데이터 관리
│   ├── search_history.py               # 검색 히스토리
│   ├── data_manager.py                 # 데이터 관리
│   ├── export.py                       # 마크다운 내보내기
│   │
│   ├── classifier/
│   │   ├── para_classifier.py          # PARA 분류 로직
│   │   └── para_agent_wrapper.py       # LangChain 통합
│   │
│   ├── database/
│   │   ├── connection.py               # DB 연결
│   │   └── metadata_schema.py          # 메타데이터 스키마
│   │
│   └── dashboard/
│       └── dashboard_core.py           # 대시보드 로직
│
├── streamlit/                          # Streamlit Frontend
│   ├── app.py                          # 메인 앱 (온보딩 + 분류 + 검색)
│   └── pages/
│       └── dashboard.py                # 대시보드 페이지
│
├── data/                               # 데이터 저장소
│   ├── exports/                        # 분류된 파일 저장 (.gitignore 처리되어있
음)
│   │   ├── Projects/
│   │   ├── Areas/
│   │   ├── Resources/
│   │   └── Archives/
│   │
│   ├── classifications/
│   │   └── classification_log.csv      # 분류 기록 누적 기록 (.gitignore 처리되
어있음)
│   ├──
│   │
│   ├── context/
│   │   └── user_context_mapping.json   # 사용자 컨텍스트 누적 기록(.gitignore 처
리되어있음)
│   │
│   ├── log/
```

```
│   │   ├── classification_연도_월_일_시간_분_초_밀리단위.json
│   │   ├── ...
│   │   └── classification_20251110_124623_127.json   # 사용자 컨텍스트 누적 기록
(.gitignore 처리되어있음)
│   │
│   ├── users/
│   │   └── users_profiles.csv          # 사용자 시용 누적 기록 (id별)
(.gitignore 처리되어있음)
│   │
│   ├── faiss_indexes/                  # FAISS 인덱스 (.gitignore 처리되어있음)
│   ├── metadata.json                   # 파일 메타데이터 (.gitignore 처리되어있
음)
│   └── search_history.json             # 검색 히스토리 (.gitignore 처리되어있음)
│
├── docs/                               # 문서
│   ├── constitution.md                 # 프로젝트 헌장
│   ├── specs/                          # 기능 명세서
│   └── troubleshooting/                # 트러블슈팅 가이드 문서들
│
├── assets/                             # 정적 리소스
│   ├── images/
│   └── figures/
│
├── USER_GUIDE.md                       # 사용자 가이드
└── README.md                           # 본 문서
```

---

5. 🚀 설치 및 실행

5.1 사전 요구사항

- **Python**: 3.11+
- **OpenAI API Key**: platform.openai.com

5.2 설치 방법

```
# 1. 저장소 클론
git clone https://github.com/jjaayy2222/flownote-mvp.git
cd flownote-mvp

# 2. 가상환경 생성 및 활성화
python -m venv venv
# 본 프로젝트에서는 pyenv로 가상환경을 관리하였음
source venv/bin/activate            # Windows: venv\Scripts\activate

# 3. 패키지 설치
pip install -r requirements.txt

# 4. 환경변수 설정
cp .env.example .env
# .env 파일에 OpenAI API 키 입력
```

```
# 5-1. Backend 실행 (FastAPI)
cd backend
python app.py
# → http://127.0.0.1:8000 에서 실행

# 5-2. Frontend 실행 (Streamlit) - 새 터미널
cd streamlit
streamlit run app.py
# → http://localhost:8501 에서 자동 실행

# 5-3. 대시보드 실행 (선택사항) - 새 터미널
streamlit run streamlit/pages/dashboard.py
# → http://localhost:8502 에서 실행
```

---

6. 📖 사용 방법

6.1 **Step 1: 온보딩**

1. `Tab 1: 온보딩` 이동
2. 이름과 직업 입력
3. GPT-4o가 추천한 10개 영역 중 5개 선택
4. 완료!

6.2 **Step 2: 파일 업로드 & 분류**

1. `Tab 2: 파일 분류` 이동
2. PDF/TXT/MD 파일 업로드
3. `분류 시작` 버튼 클릭
4. AI가 자동 분류 (사용자 맥락 반영)
5. 결과 확인:
    - 카테고리 (Projects/Areas/Resources/Archives)
    - 신뢰도 점수
    - 키워드 태그
    - 분류 근거

6.3 **Step 3: 키워드 검색**

1. `Tab 3: 키워드 검색` 이동
2. 파일 업로드 및 처리
3. 검색어 입력
4. 유사도 기반 검색 결과 확인
5. 마크다운 내보내기

6.4 **Step 4: 통계 확인**

1. `Tab 4: 분류 통계` - PARA 분포 확인
2. `Tab 5: 메타데이터` - 상세 정보 확인

3. `Dashboard` 페이지 - 실시간 대시보드

---

7. 📈 개발 히스토리

Issue별 개발 진행도

Issue	완료일	핵심 기능	상태
#1	~10/23	환경 구축, API 설정	✅
#2	10/24-25	MVP v1.0 (파일 업로드, FAISS 검색)	✅
#3	10/26-28	PARA 분류 v1.0	✅
#4	10/29	Vision API 통합	✅
#5	10/30-11/01	Dashboard v3.0 (SQLite, 대시보드 UI)	✅
#6	11/02-04	LangChain 통합, GPT-4o 분류	✅
#7	11/05-07	FastAPI Backend	✅
#8	11/08-09	온보딩 플로우 (영역 추천)	✅
#9	11/10	맥락 기반 분류 강화	✅
#10	11/11	대시보드 고도화	✅

주요 커밋 히스토리

- `42a847e` - 초기 프로젝트 구조 설정
- `58cafe2` - FAISS 검색 엔진 구현
- `7a37311` - PARA 분류기 통합
- `e22c323` - LangChain 통합
- `9bf7d28` - FastAPI Backend 구현
- `154a876` - 온보딩 플로우 추가
- `ab032a8` - 맥락 기반 분류 강화
- `20e645b` - 대시보드 고도화 (최종)

---

8. 🗺️ 로드맵

✅ 완료된 기능 (v3.5)

- [x] 스마트 온보딩 (GPT-4o 영역 추천)
- [x] AI 기반 PARA 자동 분류
- [x] 맥락 반영 분류 (사용자 직업/관심)
- [x] FAISS 키워드 검색
- [x] 실시간 대시보드
- [x] 메타데이터 관리
- [x] 검색 히스토리
- [x] 마크다운 내보내기

🚧 진행 중 (v4.0, ~11월 말)

- [ ] 분류 정확도 개선 (Few-shot learning)
- [ ] 배치 처리 기능
- [ ] 에러 처리 강화
- [ ] 태그 자동 생성
- [ ] 유사 문서 추천

🔮 향후 계획 (v5.0, 12월 이후)

- [ ] LangGraph 멀티 스텝 분류
- [ ] Notion 연동
- [ ] Obsidian Export
- [ ] 자동 폴더 구조 생성
- [ ] 배포 (Railway/Vercel)
- [ ] 협업 기능 (공유, 권한 관리)

---

9. ❓ FAQ

Q1. Backend와 Frontend를 동시에 실행해야 하나요?

A: 네, FastAPI (포트 8000)와 Streamlit (포트 8501)을 모두 실행해야 합니다.

Q2. 온보딩을 건너뛸 수 있나요?

A: 아니요. 온보딩을 완료해야 맥락 기반 분류가 작동합니다.

Q3. 어떤 파일 형식을 지원하나요?

A: PDF, TXT, MD 파일을 지원합니다.

Q4. API 비용은 얼마나 드나요?

A:

- 온보딩 영역 추천: ~$0.01/회
- 파일 분류: ~$0.02-0.05/파일
- 검색 (임베딩): ~$0.0001/쿼리

Q5. 로컬에서만 작동하나요?

A: 현재는 로컬 전용입니다. 배포 버전은 v5.0에서 제공 예정입니다.

Q6. 다른 LLM 모델을 사용할 수 있나요?

A: `backend/classifier/para_agent_wrapper.py`를 수정하면 Claude, Gemini 등 다른 모델 사용 가능합니다.

---

10. 🤝 기여하기

10.1 이슈 제보

- 버그: GitHub Issues
- 기능 제안: Discussions

10.2 기여 방법

1. `Fork` 후 새 브랜치 생성
2. 변경 후 커밋 (커밋 메시지: `feat[#이슈번호]: 설명`)
3. `Pull Request` 제출

---

11. 📄 라이선스

- `MIT License` - 자유롭게 사용, 수정, 배포 가능합니다.

---

12. 👤 개발자

Jay

- GitHub: @jjaayy2222

---

13. 🙏 감사의 말

이 프로젝트는 다음 기술/도구 덕분에 가능했습니다:

- **OpenAI** - GPT-4o, GPT-4o-mini, GPT-4.1, Text-embedding-small 모델
- **LangChain** - AI 체인 프레임워크
- **FastAPI** - 빠른 API 개발
- **Streamlit** - 빠른 UI 개발
- **Perplexity AI** - 개발 조력
- **Claude** - 멘토링 & 검수

FlowNote - AI가 당신의 문서를 정리해줍니다 🚀

Made with ❤️ by Jay