📖 FlowNote 사용자 가이드 (v6.0)

한국어 | English

> FlowNote를 처음 사용하시나요? 이 가이드를 따라 시작해보세요!

📖 목차

1. 시작하기 전에

필수 준비사항

FlowNote를 사용하기 위해 다음 항목들이 필요합니다:

✅ 소프트웨어

- **Python 3.11 이상**: python.org에서 다운로드
- **Node.js 18 이상**: nodejs.org에서 다운로드
- **Redis**: macOS의 경우 `brew install redis`로 설치

✅ API 키

- **OpenAI API Key**: platform.openai.com에서 발급
  - GPT-4o 모델 사용 권한 필요
  - 결제 정보 등록 필요 (사용량 기반 과금)

✅ 권장 환경

- **운영체제**: macOS, Linux, Windows (WSL2 권장)
- **메모리**: 최소 8GB RAM
- **저장공간**: 최소 2GB 여유 공간

2. 첫 번째 실행

2.1 프로젝트 설치

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

# 2. Python 가상환경 생성
python -m venv venv
source venv/bin/activate  # Windows: venv\Scripts\activate

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

# 4. Frontend 패키지 설치
cd web_ui
npm install
cd ..
```

2.2 환경 설정

```
# .env 파일 생성
cp .env.example .env

# .env 파일 편집 (nano 또는 원하는 에디터 사용)
nano .env
```

.env 파일 필수 설정:

```
# OpenAI API 설정
OPENAI_API_KEY=sk-your-api-key-here

# Redis 설정
REDIS_URL=redis://localhost:6379/0

# 데이터베이스 경로
DATABASE_PATH=./data/flownote.db

# 파일 저장 경로
UPLOAD_DIR=./data/uploads
```

2.3 Redis 서버 시작

```
# macOS/Linux
brew services start redis
```

```
# 또는 직접 실행
redis-server
```

2.4 서비스 실행

터미널 1 - Backend API:

```
source venv/bin/activate
python -m uvicorn backend.main:app --reload
# → http://127.0.0.1:8000
```

터미널 2 - Frontend:

```
cd web_ui
npm run dev
# → http://localhost:3000
```

터미널 3 - Celery Worker (자동화 기능용):

```
celery -A backend.celery_app.celery worker --beat --loglevel=info
```

터미널 4 - Flower (모니터링, 선택사항):

```
celery -A backend.celery_app.celery flower --port=5555
# → http://localhost:5555
```

---

3. 온보딩 과정

3.1 첫 화면 접속

브라우저에서 `http://localhost:3000/ko`로 접속합니다.

3.2 사용자 정보 입력

1. **이름 입력**: 본인의 이름 또는 닉네임 입력
2. **직업 입력**: 현재 직업 또는 역할 입력
    - 예: "소프트웨어 개발자", "프로젝트 매니저", "대학생"

3.3 관심 영역 선택

AI가 직업을 분석하여 **10개의 관심 영역**을 추천합니다.

추천 예시 (소프트웨어 개발자):

- 웹 개발
- 데이터베이스 설계
- API 개발
- DevOps
- 코드 리뷰
- 기술 문서 작성
- 프로젝트 관리
- 알고리즘 학습
- 오픈소스 기여
- 성능 최적화

이 중 5개를 선택하여 개인화된 분류 기준을 설정합니다.

3.4 온보딩 완료

선택이 완료되면 자동으로 대시보드로 이동합니다.

---

4. 파일 분류하기

4.1 파일 업로드

1. **파일 분류 탭** 이동
2. **파일 선택** 버튼 클릭
3. 분류할 파일 선택 (PDF, TXT, MD 지원)
4. **분류 시작** 버튼 클릭

4.2 분류 결과 확인

AI가 분석한 결과가 표시됩니다:

```
📋 분류 결과
──────────────────────────────────────
카테고리: Projects
신뢰도: 95%
키워드: 프로젝트, 마감일, 목표, 계획

분류 근거:
- 명확한 마감일이 명시됨
- 구체적인 목표와 결과물 정의
- 프로젝트 단계별 계획 포함
```

4.3 PARA 카테고리 이해

📋 Projects (프로젝트)

- **정의**: 명확한 마감일이 있는 단기 목표

- **예시**: "Q1 마케팅 캠페인", "웹사이트 리뉴얼"
- **특징**: 완료 시점이 명확함

🎯 **Areas (분야)**

- **정의**: 지속적으로 관리해야 하는 책임 영역
- **예시**: "팀 관리", "건강 관리", "재무 관리"
- **특징**: 끝나는 시점이 없음

📚 **Resources (자료)**

- **정의**: 참고용 정보나 학습 자료
- **예시**: "Python 튜토리얼", "디자인 가이드라인"
- **특징**: 필요할 때 참조

📦 **Archives (보관)**

- **정의**: 완료된 프로젝트나 더 이상 활성화되지 않은 자료
- **예시**: "2024년 프로젝트 결과", "구 버전 문서"
- **특징**: 보관용, 비활성

4.4 분류 수정

AI 분류가 부정확한 경우:

1. **수동 재분류** 버튼 클릭
2. 올바른 카테고리 선택
3. **저장** 버튼 클릭

---

5. 검색 기능 사용하기

5.1 키워드 검색

1. **검색 탭** 이동
2. 검색어 입력 (예: "API 문서")
3. Enter 키 또는 **검색** 버튼 클릭

5.2 검색 결과 이해

```
🔍 검색 결과 (3건)

1. REST API 설계 가이드.pdf
   카테고리: Resources
   유사도: 92%
   키워드: API, REST, 설계, 가이드

2. API 개발 프로젝트 계획.md
   카테고리: Projects
   유사도: 87%
```

```
    키워드: API, 개발, 프로젝트

3. API 문서 작성 템플릿.txt
    카테고리: Resources
    유사도: 85%
    키워드: API, 문서, 템플릿
```

5.3 고급 검색 팁

- **정확한 구문 검색**: 큰따옴표 사용 `"REST API"`
- **카테고리 필터**: 특정 PARA 카테고리만 검색
- **날짜 범위**: 특정 기간 내 파일만 검색

---

6. 대시보드 활용하기

6.1 대시보드 개요

대시보드는 3개의 주요 섹션으로 구성됩니다:

📊 통계 (Stats)

- **PARA 분포**: 카테고리별 파일 비율
- **주간 추이**: 최근 12주간 파일 처리량
- **활동 히트맵**: GitHub 스타일 연간 활동

🌐 그래프 뷰 (Graph View)

- **파일-카테고리 관계**: React Flow 기반 시각화
- **노드 클릭**: 파일 상세 정보 확인
- **줌/팬**: 마우스 휠 및 드래그로 탐색

🔄 동기화 모니터 (Sync Monitor)

- **Obsidian 연결 상태**: 실시간 표시
- **MCP 서버 상태**: 연결 여부 확인
- **최근 동기화**: 마지막 동기화 시간

6.2 실시간 업데이트 (v6.0)

WebSocket 기반 실시간 업데이트:

- 파일 분류 완료 시 즉시 반영
- 동기화 상태 변경 즉시 표시
- 네트워크 트래픽 50% 감소

MiniMap

- 우측 하단 미니맵으로 전체 구조 파악

- 현재 뷰포트 위치 확인

7. 자동화 기능 설정

7.1 자동 재분류

설정 위치: `backend/celery_app/config.py`

```
# 매일 자정에 신뢰도 낮은 파일 재분류
'daily-reclassify': {
    'task':
'backend.celery_app.tasks.reclassification.daily_reclassify_tasks',
    'schedule': crontab(hour=0, minute=0),
}
```

작동 방식:

1. 신뢰도 70% 미만 파일 자동 선택
2. 최신 AI 모델로 재분류
3. 결과를 로그에 기록

7.2 스마트 아카이빙

```
# 매주 일요일 자정에 오래된 프로젝트 아카이빙
'weekly-archive': {
    'task':
'backend.celery_app.tasks.archiving.weekly_archive_old_projects',
    'schedule': crontab(day_of_week=0, hour=0, minute=0),
}
```

작동 방식:

1. 90일 이상 수정되지 않은 Projects 파일 탐지
2. Archives로 이동 제안
3. 사용자 승인 후 이동

7.3 Flower로 모니터링

`http://localhost:5555`에서 확인:

- 실행 중인 작업
- 작업 성공/실패 통계
- Worker 상태

---

8. Obsidian 연동

8.1 MCP 서버 설정

Claude Desktop 설정 파일 (`~/Library/Application Support/Claude/claude_desktop_config.json`):

```
{
  "mcpServers": {
    "flownote": {
      "command": "python",
      "args": ["-m", "backend.mcp.server"],
      "cwd": "/path/to/flownote-mvp"
    }
  }
}
```

8.2 Obsidian Vault 연동

1. **설정 파일 수정** (`.env`):

```
OBSIDIAN_VAULT_PATH=/path/to/your/vault
```

2. **자동 동기화 활성화**:

```
OBSIDIAN_AUTO_SYNC=true
SYNC_INTERVAL=300  # 5분마다
```

8.3 충돌 해결 (v6.0)

파일 충돌 발생 시:

1. **알림 수신**: 대시보드에 충돌 알림 표시
2. **Diff Viewer 열기**: 변경사항 비교
3. **해결 방법 선택**:
    - **Keep Local**: 로컬 버전 유지
    - **Keep Remote**: Obsidian 버전 유지
    - **Keep Both**: 두 버전 모두 보관 (파일명에 타임스탬프 추가)

Diff Viewer 기능 (v6.0):

- Monaco Editor 기반 Side-by-Side 비교
- Syntax Highlighting
- Markdown 프리뷰
- 인라인 Diff 표시

---

9. 언어 설정 변경

9.1 웹 UI 언어 전환 (v6.0)

1. **우측 상단** 언어 스위처 클릭
2. **한국어** 또는 **English** 선택
3. URL 자동 변경 (`/ko` ↔ `/en`)
4. UI 즉시 업데이트

지원 언어:

- 한국어 (ko)
- English (en)

9.2 API 응답 언어 설정

HTTP 요청 시 `Accept-Language` 헤더 설정:

```
curl -H "Accept-Language: ko" http://localhost:8000/api/classify
curl -H "Accept-Language: en" http://localhost:8000/api/classify
```

---

10. 문제 해결

10.1 자주 발생하는 문제

❌ Redis 연결 오류

```
ConnectionRefusedError: [Errno 61] Connection refused
```

해결 방법:

```
# Redis 서버 시작
brew services start redis

# 또는
redis-server
```

❌ OpenAI API 오류

```
openai.error.AuthenticationError: Incorrect API key provided
```

해결 방법:

1. `.env` 파일의 `OPENAI_API_KEY` 확인
2. API 키 앞뒤 공백 제거
3. 유효한 키인지 platform.openai.com에서 확인

❌ **포트 충돌**

```
Error: listen EADDRINUSE: address already in use :::8000
```

해결 방법:

```
# 포트 사용 중인 프로세스 찾기
lsof -i :8000

# 프로세스 종료
kill -9 <PID>
```

❌ **WebSocket 연결 실패 (v6.0)**

```
WebSocket connection failed
```

해결 방법:

1. Backend API가 실행 중인지 확인
2. 브라우저 콘솔에서 오류 메시지 확인
3. 방화벽 설정 확인

10.2 로그 확인

Backend 로그:

```
# Uvicorn 로그
tail -f logs/uvicorn.log

# Celery 로그
tail -f logs/celery.log
```

Frontend 로그:

```
# Next.js 개발 서버 로그
cd web_ui
npm run dev
```

10.3 데이터베이스 초기화

주의: 모든 데이터가 삭제됩니다!

```
# 데이터베이스 파일 삭제
rm data/flownote.db

# 업로드 파일 삭제
rm -rf data/uploads/*

# 서버 재시작
python -m uvicorn backend.main:app --reload
```

10.4 지원 받기

- **GitHub Issues**: github.com/jjaayy2222/flownote-mvp/issues
- **이메일**: qkfkadmlEkf@gmail.com
- **문서**: README.md

📚 추가 리소스

- **API 문서**: http://localhost:8000/docs (Swagger UI)
- **Phase 문서**: `docs/P/` 디렉토리
  - v6.0 Phase 1: WebSocket
  - v6.0 Phase 2: Diff Viewer
  - v6.0 Phase 3: i18n

FlowNote와 함께 효율적인 문서 관리를 시작하세요! 🚀