

프로젝트 폴더 구조 재정리 가이드

작성일: 2025-11-06

소요 시간: 약 1시간

최종 결과: 폴더 구조 정리 + 배포 성공

📋 목차

1. 문제 상황 분석
2. 기존 구조의 문제점
3. 새로운 구조 설계
4. 마이그레이션 단계
5. 배포 설정

1. 문제 상황 분석

1.1 발생한 증상

- ❌ 로컬에서 잘 되는데 배포에서 안 됨
- ❌ `ModuleNotFoundError: No module named '...'`
- ❌ 경로 импорт 오류
- ❌ Streamlit Cloud에서 `app.py` 찾을 수 없음

1.2 근본 원인

기존 구조의 문제:

```
flownote-mvp/  
├── frontend/  
│   └── streamlit/  
│       └── app.py  
├── backend/  
├── README.md  
└── requirements.txt
```

← 깊게 들어가있음!

문제점:

1. `streamlit_app.py`가 루트에 없음
2. 배포 플랫폼이 메인 파일 못 찾음
3. 상대 경로 импорт 복잡
4. 폴더 구조가 MVP에 비해 과도

2. 기존 구조의 문제점

2.1 배포 설정 복잡성

```
# ❌ 이렇게 하면 배포 실패
Streamlit Cloud에서 찾는 파일:
  → streamlit_app.py (프로젝트 루트)
  → 또는 app.py (프로젝트 루트)

기존 위치:
  → frontend/streamlit/app.py ← 못 찾음!
```

2.2 임포트 경로 문제

```
# ❌ 복잡한 상대 경로
sys.path.append("../..../backend")
from backend.services.classifier import KeywordClassifier

# ✅ 간단한 절대 경로 (새 구조)
from services.classifier import KeywordClassifier
```

2.3 개발 흐름 복잡성

```
# ❌ 여러 폴더 관리
- frontend/streamlit/ 편집
- backend/ 편집
- 테스트할 때마다 경로 확인

# ✅ 단순화된 구조
- streamlit/ (현재 MVP)
- backend/ (공유 로직)
- web_ui/ (향후 React)
```

3. 새로운 구조 설계

3.1 MVP 최적화 구조

```
flownote-mvp/
├── streamlit/
│   ├── app.py
│   └── pages/
│       ├── __init__.py
│       └── dashboard.py
```

← 현재 MVP (프로덕션)
 ← 메인 파일
 ← Streamlit 페이지들



3.2 폴더별 역할

폴더	역할	상태
streamlit/	Streamlit 웹앱 (MVP)	🚀 프로덕션
backend/	공유 비즈니스 로직	🔄 재사용 가능
web_ui/	React 버전 (향후)	📅 계획
docs/	문서 & 트러블슈팅	📖 지속 업데이트

4. 마이그레이션 단계

4.1 Step 1: 폴더 생성

```
# 프로젝트 루트에서
mkdir -p streamlit
mkdir -p backend/services
mkdir -p backend/routes
mkdir -p docs/troubleshooting
```

4.2 Step 2: 파일 이동

```
# Streamlit 앱 이동
mv frontend/streamlit/app.py streamlit/app.py
mv frontend/streamlit/config.py streamlit/config.toml

# 백엔드 파일 이동
mv backend/services/* backend/services/
mv backend/routes/* backend/routes/
```

4.3 Step 3: 임포트 경로 수정

```
# ❌ 이전
import sys
sys.path.append("../..backend")
from services.classifier import KeywordClassifier

# ✅ 이후 (Option 1: 절대 경로)
import sys
sys.path.insert(0,
os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
from backend.services.classifier import KeywordClassifier

# ✅ 이후 (Option 2: PYTHONPATH)
# Streamlit 실행 전에:
# export PYTHONPATH="${PYTHONPATH}:${pwd}"
# streamlit run streamlit/app.py
```

4.4 Step 4: streamlit_app.py 생성 (배포용)

```
# 프로젝트 루트에 streamlit_app.py 생성
import subprocess
import sys

# streamlit/app.py 실행
if __name__ == "__main__":
    subprocess.run([
        sys.executable, "-m", "streamlit", "run",
        "streamlit/app.py"
    ])
```

또는 더 간단하게:

```
# Streamlit Cloud 대시보드에서
# "Settings" → "General"
```

```
# → "Custom App Main File Path" 설정  
streamlit/app.py
```

4.5 Step 5: 테스트

```
# 로컬에서 테스트  
cd streamlit  
streamlit run app.py  
  
# 또는 루트에서  
streamlit run streamlit/app.py
```

5. 배포 설정

5.1 requirements.txt 확인

```
# 프로젝트 루트에서 한 번만!  
pip freeze > requirements.txt  
  
# 중복 제거 및 정렬
```

5.2 .streamlit/config.toml 설정

```
[client]  
toolbarMode = "minimal"  
showErrorDetails = false  
  
[logger]  
level = "info"  
  
[server]  
maxUploadSize = 200  
enableXsrfProtection = true
```

5.3 Streamlit Cloud 배포 설정

- 1 Streamlit Cloud 접속
- 2 "New app" → GitHub 리포지토리 선택
- 3 "App URL" 선택
- 4 "Advanced settings" 클릭
- 5 "Main file path" = streamlit/app.py (또는 streamlit_app.py 루트)
- 6 "Deploy" 클릭

5.4 자동 배포 설정

```
# GitHub 리포지토리 Settings
→ Webhooks
→ Streamlit Cloud 웹훅 추가 (자동)
```

또는 수동:

```
→ Streamlit Cloud Dashboard
→ "Rerun" 버튼 클릭
```



마이그레이션 체크리스트

구조 정리:

- ☐ streamlit/ 폴더 생성 및 파일 이동
- ☐ backend/ 폴더 구조 정리
- ☐ 불필요한 폴더 삭제

임포트 수정:

- ☐ streamlit/app.py의 모든 import 수정
- ☐ 상대 경로 → 절대 경로로 변경
- ☐ sys.path 추가 (필요시)

배포 준비:

- ☐ requirements.txt 생성
- ☐ .streamlit/config.toml 생성
- ☐ streamlit_app.py 또는 경로 설정

테스트:

- ☐ 로컬에서 streamlit run 테스트
- ☐ 모든 기능 정상 작동 확인
- ☐ Git 커밋 및 푸시

배포:

- ☐ Streamlit Cloud에서 새 앱 생성
- ☐ 리포지토리 및 경로 확인
- ☐ 배포 완료 후 링크 테스트



향후 계획

현재 (MVP):

- └─ streamlit/ (프로덕션)
- └─ backend/ (공유 로직)

Next Phase:

- └─ web_ui/ (React 추가)

```
└─ backend/ API 확장
```

Final:

```
├─ streamlit/ (선택 사항)  
├─ web_ui/ (주 서비스)  
├─ mobile_app/ (고려)  
└─ backend/ (완성)
```

참고 자료

- [Streamlit 디렉토리 구조 가이드](#)
 - [Python 프로젝트 구조 Best Practice](#)
-