생성형 AI 기반 RAG 개발자 과정

문서 업로드 기능

Session 18 / 33 — Day 3

파일을 받아 안전하게 저장한다 — POST /files

박수현 · (사)한국정보공학기술사회 · 2026

LLM 이론 · LangChain · ChromaDB CRUD · Agent — 4일 핸즈온

이번 시간을 마치면

📤 업로드 수신

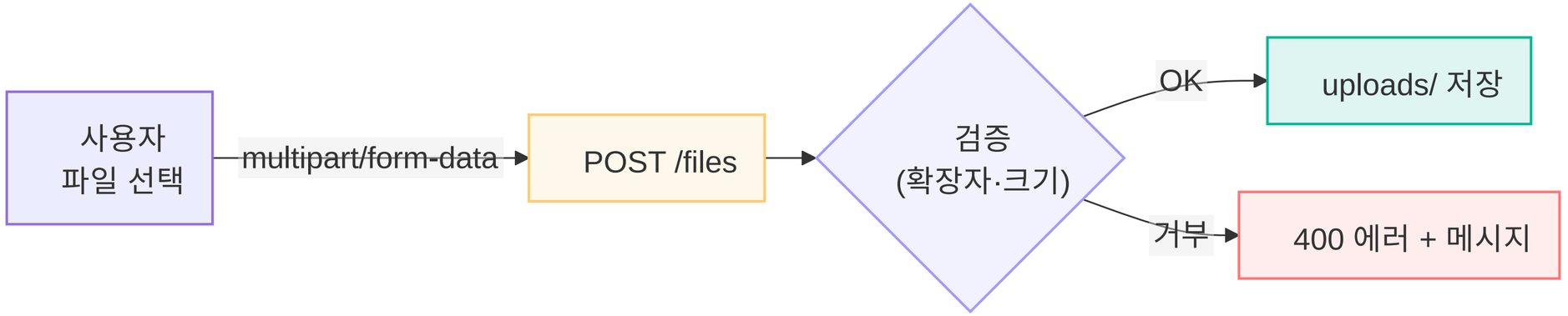
- **multipart/form-data** 개념
- `POST /files` 로 다중 수신
- `uploads/` 에 저장

🛡️ 안전하게

- 확장자 **검증** (PDF/TXT 기본)
- 파일명 **정규화** (보안)
- 빈 파일·중복 처리

> 🎯 목표: 신뢰할 수 없는 입력(업로드)을 **검증·저장**까지 안전하게.

파일은 폼 데이터로 전송된다



- JSON 과 달리 **바이너리(파일)** 를 담는 인코딩
- 한 요청에 **여러 파일** 동시 전송 가능
- 서버는 `request.files` 로 수신

여러 파일 받기 (스니펫)

```
@app.post("/files")
def upload():
    uploaded = request.files.getlist("file")   # 여러 개
    if not uploaded:
        return jsonify({"error": "파일이 없습니다"}), 400

    results = []
    for f in uploaded:
        # … 검증 → 저장 → (색인) …
        results.append({"filename": f.filename})
    return jsonify(results)
```

- `getlist("file")` — 다중 업로드 지원
- 파일이 없으면 **400** 으로 즉시 거부

> 📁 전체: `8.web_app/5.file_manager_restapi/app.py`

신뢰할 수 없는 입력 필터링 (스니펫)

```
from werkzeug.utils import secure_filename

ALLOWED = {".pdf", ".txt"}          # DOCX 는 한 줄로 확장 가능

def save_upload(f):
    if not f or f.filename == "":                # 빈 파일 거부
        raise ValueError("파일이 선택되지 않았습니다")
    name = secure_filename(f.filename)          # 경로 조작 방지
    ext = os.path.splitext(name)[1].lower()
    if ext not in ALLOWED:                       # 허용 형식만
        raise ValueError(f"지원하지 않는 형식: {ext}")
    path = os.path.join("uploads", name)
    f.save(path)                                 # 저장만 — 색인은 다음 단계
    return path
```

- `secure_filename` — `../../etc/passwd` 류 **경로 조작** 차단
- 화이트리스트 방식 — 허용 확장자만 통과 (**PDF/TXT 기본**)
- 빈 파일(`filename == ""`)도 즉시 거부

> ⚠️ 업로드는 외부 입력이므로 **항상 검증**한다. 크기 제한(`MAX_CONTENT_LENGTH`)도 권장. 📎 DOCX 확장: `ALLOWED` 에 `.docx` 추가, `Docx2txtLoader` 분기 추가로 대응.

결과를 명확히 반환한다

성공 응답 예

```
[
  {"filename": "faq.pdf",
   "status": "saved",
   "chunks": 42},
  {"filename": "rule.txt",
   "status": "saved",
   "chunks": 18}
]
```

응답 구성 요소

- 저장된 **파일명**
- 처리 **상태** (saved / failed)
- 색인까지 수행한 경우 **chunk 수**
- 실패 시 **사유**

> 🔑 프론트엔드는 이 JSON 으로 목록·진행상태를 표시한다.

> 💡 이번 단계의 범위는 **수신·검증·저장**까지다. 청킹·임베딩(색인)은 **다음 회차**에서 다루며, `chunk 수`는 색인 단계에서 채워진다.

핵심 5가지

1. ✅ 파일은 **multipart/form-data** 로 전송 → `request.files`
2. ✅ `POST /files` + `getlist` 로 **여러 개** 동시 업로드
3. ✅ **확장자 화이트리스트**(PDF/TXT 기본 · DOCX 확장 가능)로 검증
4. ✅ `secure_filename` 으로 **경로 조작 차단**
5. ✅ 결과는 **파일명·상태·chunk 수** JSON 으로 응답