**보고서 초안**

**1. 표지**

- **제목**: FlowMate 모델 구조 및 데이터 흐름도

- **작성자**: [작성자 이름]

- **작성일**: [작성일]

- **목차**

**2. 목차**

1. **제목**: FlowMate 모델 구조 및 데이터 흐름도

2. **1. 입력 파일 수집 및 분기**

3. **2. 파일별 추출 로직 (Extractor Layer)**

4. **3. 이미지 분석 (멀티모달 처리)**

5. **4. 텍스트 청크 분할 (Chunking Layer)**

6. **5. 임베딩/벡터DB 적재 (Vector Store Layer)**

7. **6. 질의응답 및 대화 히스토리 (RAG + Memory)**

8. **7. 오디오 특성 분석(선택적)**

9. **8. 핵심 모듈 간 연결 구조 (요약도)**

10. **9. 실제 예시 흐름**

11. **10. 확장성/장점**

**3. 1. 입력 파일 수집 및 분기**

- 사용자가 문서/이미지/영상 등 다양한 파일을 업로드하면, parsing\_utils.py의 start\_extracting(file\_path) 함수가 확장자에 따라 알맞은 추출(파싱) 모듈로 자동 분기합니다.

- 예) pdf → extracting\_pdf.py, docx → extracting\_docx.py, pptx → extracting\_pptx.py 등

**4. 2. 파일별 추출 로직 (Extractor Layer)**

- 각 파일 타입별로 특화된 추출 함수가 존재하며, 파일 내 텍스트/표/이미지를 가능한 한 순서대로 추출해 텍스트화합니다.

- 주요 파서 역할 예시:

- PDF (extracting\_pdf.py): 페이지별 본문, 표, 이미지를 순서대로 읽음

- DOCX (extracting\_docx.py): 본문, 표, 이미지를 한 번에 처리 (이미지 병렬 분석)

- PPTX (extracting\_pptx.py): 각 슬라이드별 텍스트·이미지 추출, 이미지 분석 포함

- XLSX/XLS (extracting\_xlsx.py): 시트별 표(셀 데이터)와 이미지, 이미지 분석 결과까지

- CSV (extracting\_csv.py), TXT (extracting\_txt.py): 표 데이터(마크다운), 본문 텍스트 추출

- 이미지 (extracting\_img.py / image\_utils.py): OCR+VLM 모델(Qwen2.5-VL)로 표·그래프·텍스트 자동 추출

- 영상/오디오 (video\_processor.py, audio\_analysis.py): 영상에서 음성 추출 → STT 변환 → 요약/분석

**5. 3. 이미지 분석 (멀티모달 처리)**

- 실제 RAG(Retrieval-Augmented Generation) 기반 응답 반환

- question\_answer\_with\_memory() 함수에서 수행

**6. 4. 텍스트 청크 분할 (Chunking Layer)**

- 추출된 전체 텍스트는 chunk\_utils.py의 get\_adaptive\_splitter()로 길이와 구조에 따라 동적으로 청크(단위 텍스트 덩어리) 분할

- → LangChain의 Document 객체 리스트로 변환

- → 이후 임베딩 및 검색의 기본 단위가 됨

**7. 5. 임베딩/벡터DB 적재 (Vector Store Layer)**

- vectordb\_upload\_search.py의 data\_to\_vectorstore()를 통해 각 청크는 Ollama 기반 임베딩 모델(bge-m3:567m)로 임베딩 → Qdrant(VectorDB)에 적재/관리

- 기존 DB에 동일 파일이 있으면 중복 저장 없이 사용

**8. 6. 질의응답 및 대화 히스토리 (RAG + Memory)**

- 사용자가 질문을 입력하면

1. 유사도 검색(벡터DB에서 관련 청크 검색)

2. 이전 대화 내용(BufferMemory)과 합쳐

3. 프롬프트 엔지니어링된 형태로 LLM(Qwen2.5-VL) 호출

→ 실제 RAG(Retrieval-Augmented Generation) 기반 응답 반환

- question\_answer\_with\_memory() 함수에서 수행

**9. 7. 오디오 특성 분석(선택적)**

- 발표 영상 등 오디오 파일이 입력되면 audio\_analysis.py로 피치(음높이), 억양, 말 속도 등 음성 특성 분석 가능

**10. 8. 핵심 모듈 간 연결 구조 (요약도)**

- [사용자 파일 업로드]

│

▼

[parsing\_utils.py] - 확장자 분기

│

▼

[extractor 계열 모듈]

(pdf/docx/pptx/xlsx 등)

│

▼

[image\_utils.py]

(이미지별 멀티모달 분석)

│

▼

[chunk\_utils.py]

(문서 청크 분할)

│

▼

[vectordb\_upload\_search.py]

(Qdrant 임베딩/적재 + 검색)

│

▼

[질문/응답 RAG]

(이전 대화 맥락 BufferMemory 포함)

│

▼

[최종 응답 반환]

**11. 9. 실제 예시 흐름**

1. 사용자가 PDF, DOCX 등 파일 업로드

2. 해당 파일 유형에 맞는 extractor가 텍스트·표·이미지 등 모두 추출

3. 이미지가 있으면 VLM(Qwen2.5-VL)로 분석 결과 생성

4. 전체 텍스트를 의미 단위로 청크 분할

5. 각 청크를 임베딩, 벡터DB에 적재

6. 질문이 들어오면 관련 청크 검색 + 대화 히스토리와 합쳐 LLM 프롬프트 생성

7. 최종 답변 반환

**12. 10. 확장성/장점**

- 모든 확장자 파일에 대응, 자동 추출·분할·벡터화·검색 가능

- 영상/오디오 (video\_processor.py, audio\_analysis.py)

- 영상에서 음성 추출 → STT 변환 → 요약/분석

- 모든 이미지 추출 과정에서는 analyze\_image\_with\_qwen(image\_path) 함수로 Qwen2.5-VL 등 멀티모달 LLM 기반의 이미지 내용 추출/설명을 붙임

- 텍스트 청크 분할 (Chunking Layer)

- 임베딩/벡터DB 적재 (Vector Store Layer)

- 질의응답 및 대화 히스토리 (RAG + Memory)

- 오디오 특성 분석(선택적)

- 핵심 모듈 간 연결 구조 (요약도)

- 실제 예시 흐름

**13. 결론**

- 이 보고서는 FlowMate 모델의 구조와 데이터 흐름을 체계적으로 설명하였습니다. 모델은 다양한 파일 유형을 자동으로 분석하고, 텍스트, 이미지, 오디오 등의 데이터를 분석하여 임베딩, 벡터DB에 적재하고, 이를 기반으로 질의응답 시스템을 구축합니다. 이를 통해 사용자는 다양한 파일 유형을 입력하여 빠르게 정보를 얻을 수 있습니다. 또한, 모델은 이미지 분석, 텍스트 청크 분할, 임베딩/벡터DB 적재, 질의응답 및 대화 히스토리 기능을 통해 더욱 강력한 분석과 응답을 제공합니다.

**14. 참고 문헌**

- [참고 문헌 목록]

---

이 보고서는 FlowMate 모델의 구조와 데이터 흐름을 체계적으로 설명하였습니다. 모델은 다양한 파일 유형을 자동으로 분석하고, 텍스트, 이미지, 오디오 등의 데이터를 분석하여 임베딩, 벡터DB에 적재하고, 이를 기반으로 질의응답 시스템을 구축합니다. 이를 통해 사용자는 다양한 파일 유형을 입력하여 빠르게 정보를 얻을 수 있습니다. 또한, 모델은 이미지 분석, 텍스트 청크 분할, 임베딩/벡터DB 적재, 질의응답 및 대화 히스토리 기능을 통해 더욱 강력한 분석과 응답을 제공합니다.