< UI >

**1. 개요**

**1-1. 소프트웨어 목적**

환자 DB 연결해 모델링한 최종 결과를 관리자에게 보여 줄 수 있는 웹사이트 생성

**1-2. 소프트웨어 개발 환경 및 실행 환경**

**1-2-1. 개발 환경**

• **주요 플랫폼:** macOS(Apple Silicon), Windows 지원

• **권장 운영체제:** macOS 11.0 이상, Windows 10 이상

• **가상환경:** Anaconda 설치 필요

• 초기설정: yaml파일로 자동 설치됨

텍스트, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

**• 프로그래밍 언어:** python 3.10

**• 데이터베이스:** MySQL

**• 파일 경로 사용자 설정:** base\_path

**• 웹프레임워크:** Streamlit 1.39.0

**• 성능최적화**

• Tensorflow: CPU 전용 모드, 스레드 수 제한

• PyTorch: MPS 비활성화(Apple Silicon 호환성 위함)

• 병렬처리: TF\_NUM\_INTEROP\_THREADS=1, TF\_NUM\_INTRAOP\_THREADS=1

**1-2-1. 주 라이브러리**

| **범주** | **라이브러리** | **버전** |
| --- | --- | --- |
| 데이터 처리 | numpy | 2.0.2 |
| pandas | 2.3.1 |
| 머신러닝 | scikit-learn | 최신버전 |
| 딥러닝 | tensorflow | 2.19.0 |
| tensorflow-macos (macOS 전용) | 2.15.1 |
| torch | 2.8.0 |
| transformers | 4.55.2 |
| 오디오 처리 | librosa | 0.11.0 |
| torchaudio | 2.8.0 |
| soundfile | 0.12.1 |
| pydub | 0.25.1 |
| 모델허브 | tensorflow-hub | 0.16.1 |
| openai-whisper | 20250625 |
| 시각화 | matplotlib | 3.10.0 |
| plotly | 5.24.1 |
| 수치 계산 | scipy | 1.14.1 |
| sympy | 1.13.3 |
| 웹 인터페이스 | streamlit | 1.39.0 |
| 환경 변수 관리 | python-dotenv | 1.1.1 |
| 데이터베이스 연결 | mysql-connector-python | 9.1.0 |

- 프로젝트 구조

ui/: 웹 인터페이스(Streamlit)

models/: 추론 모델에 대한 코드 및 파일

db/: 데이터베이스 스키마 및 연결용

files/: 데이터 파일 저장용

.env: 사용자 환경설정(sql, 파일경로)

.environment.yaml: 환경설정

Readme.txt: 초기 설명

**2. 소프트웨어 구현 사항**

**2-1. 소프트웨어 구성 모듈**

**2-1-1. ui 폴더**

**2-1-1-1. db\_service.py:** 데이터베이스 연동

save\_scores\_to\_db() – 모델링으로 나온 결과(dictonary)를 DB에 저장: QUESTION\_NO와 QUESTION\_MINOR\_NO 재정의 필요

get\_reports() – 리포트 메인 화면에 필요한 DB 호출

get\_db\_modules() – db 폴더에 있는 함수 호출(메모리 절약 및 보안유지)

**2-1-1-2. upload\_service.py**: 파일 업로드 및 데이터베이스 저장

Zip\_upload() – SQL에 DB 경로 저장, order\_num 갱신

**2-1-1-3. env\_utils.py**: 가상환경 설치

find\_conda\_base() – 현재 실행 중인 가상환경 조회

create\_environment() – environment.yaml을 기반으로 새로운 가상환경 생성

activate\_conda\_environment() – 가상환경 실행

delete\_conda\_environment() – 가상환경 삭제

                2-1-1-4. style\_utils.py: CSS 관리용

                2-1-1-5. report\_view.py: 리포트 관련 화면 인터페이스

show\_main\_interface() – 메인화면 초기값 설정, 사이드바

model\_process() – Models 폴더에 있는 API들 호출

show\_report\_page() – 리포트 메인화면

show\_detail\_common() – 검사유형별 리포트 상세 화면 중 상단(고객 정보)

show\_detail() - 검사유형별 리포트 상세화면 중 하단(검사 결과)

show\_graph() – 검사결과 그래프(A유형 한정)

                2-1-1-6. app.py: 메인 애플리케이션 엔트리

Main() – 상태 초기화, 로그인화면, 환자정보 등록화면

        2-1-2. models 폴더: API

                2-1-2-1. ah\_sound.py: ‘아 소리내기’ 점수 계산 API

                2-1-2-2. guess\_end.py: ‘끝말 맞추기’ 점수 계산 API

                2-1-2-3. ltn\_rpt.py: ‘듣고 따라 말하기’ 점수 계산 API

                2-1-2-4. ptk\_sound.py: ‘퍼터커 반복하기’ 점수 계산 API

                2-1-2-5. say\_ani.py: ‘동물 이름 말하기’ 점수 계산 API

                2-1-2-6. say\_obj.py: ‘물건 이름 말하기’ 점수 계산 API

        2-1-3. db 폴더

                2-1-3-1. data: SQL 초기 설정

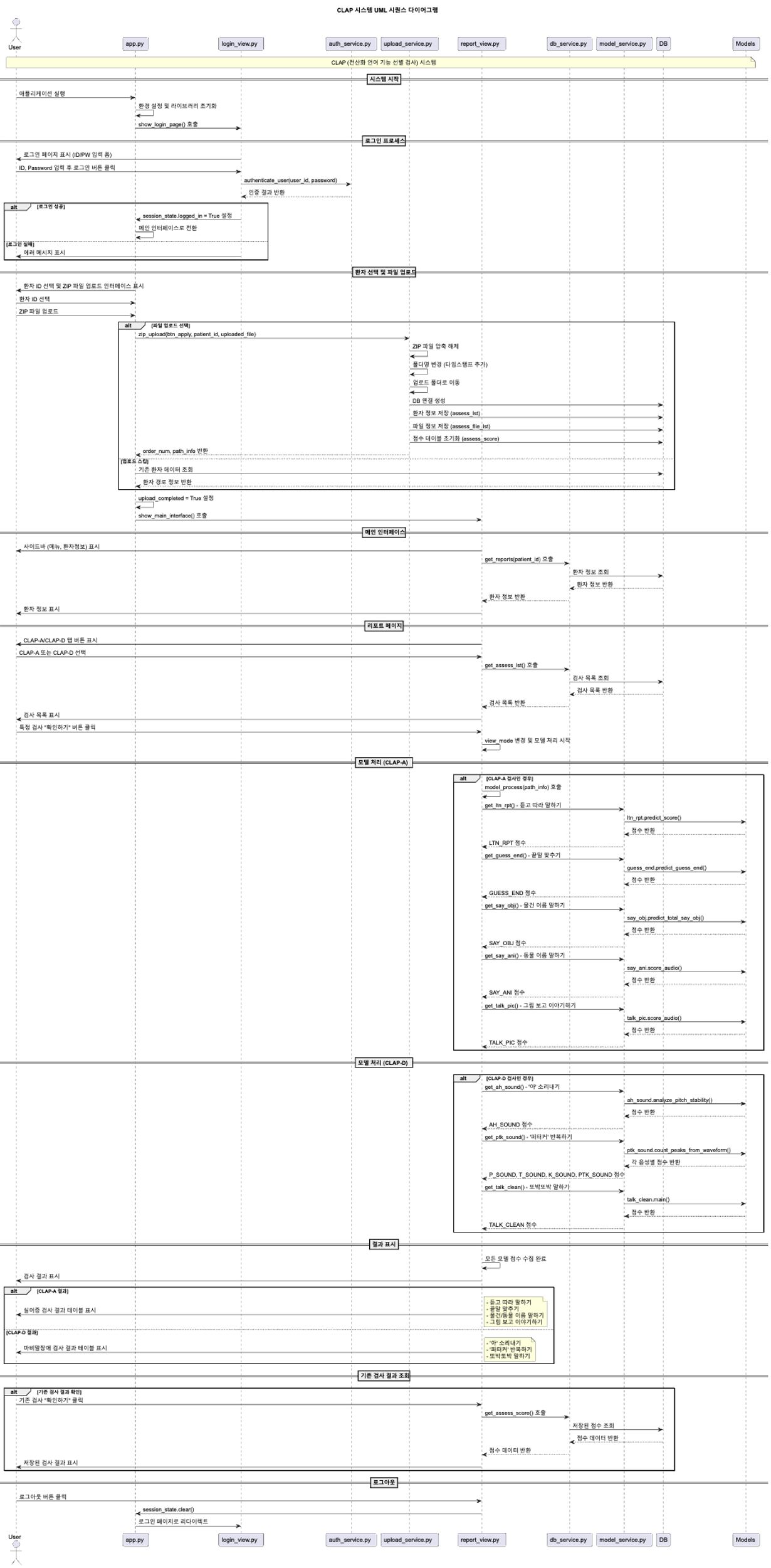
                2-1-3-2. src

                        2-1-3-2-1. model\_comm.py: 모델링된 검사 점수 관리

                        2-1-3-2-2. report\_main.py: 리포트용 DB 호출

        2-1-4. files: 업로드한 환자 정보 저장

2-2. 소프트웨어 구성 모듈별 설명



2-2-1. app.py

  - Streamlit 기반 웹 애플리케이션의 진입점

  - 환경 설정 및 라이브러리 초기화 (TensorFlow, PyTorch 설정)

  - 세션 상태 관리 및 사용자 인증 흐름 제어

  - 환자 ID 선택 및 파일 업로드 인터페이스 제공

스크린샷, 텍스트, 운영 체제, 소프트웨어이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

2-2-2. upload\_service.py

  - ZIP 파일 업로드 및 압축 해제

  - CLAP\_A, CLAP\_D 오디오 파일 분류 및 데이터베이스에 경로 저장

  - 환자 정보(CSV) 파싱 및 assess\_lst 테이블 저장

  - 중복 데이터 처리 및 파일 메타데이터 관리

스크린샷, 텍스트, 소프트웨어, 운영 체제이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

2-2-3. report\_view.py

  -  show\_main\_interface(): 메인화면 사이드바, 환자 정보, 메뉴 구성(최신순)

텍스트, 스크린샷, 소프트웨어, 운영 체제이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

  -  model\_process(): 모델 실행 및 점수 계산(1차 실행 후 DB에 결과를 저장하여 재사용)

        - 각 모델은 해당하는 파일이 있는 경우에만 lazy loading으로 실행해 메모리 절약

  -  show\_detail(): 특정 환자, 특정 검사의 결과(CLAP-A, CLAP-D) 테이블 렌더링

텍스트, 소프트웨어, 번호, 컴퓨터 아이콘이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

텍스트, 소프트웨어, 번호, 스크린샷이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

2-2-4. db\_service.py

  - 데이터베이스와 즉시 연결하지 않고 필요한 모듈만 호출할 수 있도록 구성

  - get\_db\_modules(): db.src에서 model\_comm, report\_main 모듈 import

2-2-5. model\_service.py (lazy loading 관리)

  - API 모델을 개별 로드할 수 있도록 각각 정의

2-3. 저장 가능한 세션 목록

logged\_in: 로그인여부

user\_info: 사용자 ID

patient\_id: 환자번호(PK)

order\_num: 검사 등록 순서(PK) – 1부터 시작하며 파일 업로드 시 부여됨

current\_page: 사이드바에서 선택된 메뉴 - 평가, 재활, 리포트

view\_mode: 상세보기할 리포트 선택 – list, clap\_a\_detail, clap\_d\_detail

selected\_filter: 리포트 구분용 – CLAP\_A, CLAP\_D

upload\_completed: 파일 등록 완료 여부

path\_info: 환자번호, order\_num, 검사유형, 문제이름, 문제번호, 파일경로 최상단(유일성 부여하기 위해 현재시각 추가), 하단 파일경로(SUB\_PATH = 검사별 고유번호), 음성파일



텍스트, 스크린샷, 폰트, 흑백이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

**그림 2 SUB\_PATH, QUESTION\_CD, MAIN\_PATH 예**

: API 모델링 시 Input 값으로 활용

Skip: ‘업로드 스킵’ 버튼 클릭 여부

model\_completed: 최초 모델링 완료 여부 – 등록되어 있으면 모델링하지 않고 SQL로 결과 호출(order num을 key로 활용)

3. 소프트웨어 개선 사항

3-1. 예외 사항 처리

* 현재는 제공받은 입력데이터가 환자번호와 검사결과뿐이지만, 각 환자에 대한 patient\_info(이름, 나이, 성별, 과거 병력 등)도 주어지면 반영 가능(현재는 등록되지 않은 정보가 요청되면 None 처리함)
* 업로드 스킵 버튼: 해당 환자 정보가 SQL에 없으면 에러

3-2. 기능 구현

* 메인화면의 평가, 재활 버튼은 주어진 UI 샘플이 없어 구현하지 않음 – 스토리보드 필요
* 모델 로딩 시간 단축
* 로그인: 데모화면만 있고 연결된 DB나 회원가입 기능 없음
* 사용자 친화적으로 UI/UX 개선
* QUESTION\_NO와 QUESTION\_MINOR\_NO 재정의 필요(2-1-1-1)
* 세션(st.session\_state) 정리: 중복, 미사용, 저장 문제 등