Py Reivew App

최재원
부산대학교 정보컴퓨터공학부 인공지능전공

Background

- 동기들과의 Python 스터디에서 코드 리뷰를 진행하며 '빠른 피드백'의 중요성을 체감
- 정해진 시간에만 리뷰가 가능하여 발생하는 **피드백 지연** 문제를 해결하고자 함
- LLM을 활용하여 코드 리뷰를 자동화하는 웹 앱 개발을 구상

Problem Specification

- **입력 (Input)**:
  - 리뷰가 필요한 문제 설명
  - 사용자가 작성한 Python 코드

- **출력 (Output)**:
  - Linter, Formatter, LLM 등을 활용한 자동화된 코드 리뷰 결과

Design

- **웹 서버**: FastAPI
- **데이터베이스**: TinyDB
  - 유저 정보, 리뷰 로그 저장을 위한 간단한 JSON 기반 NoSQL
- **로그인 및 회원가입**:
  - 세션(Session) 기반으로 로그인 상태 유지
  - 회원가입 시, `pbkdf2_sha256` 알고리즘으로 패스워드를 **해싱** 후 저장

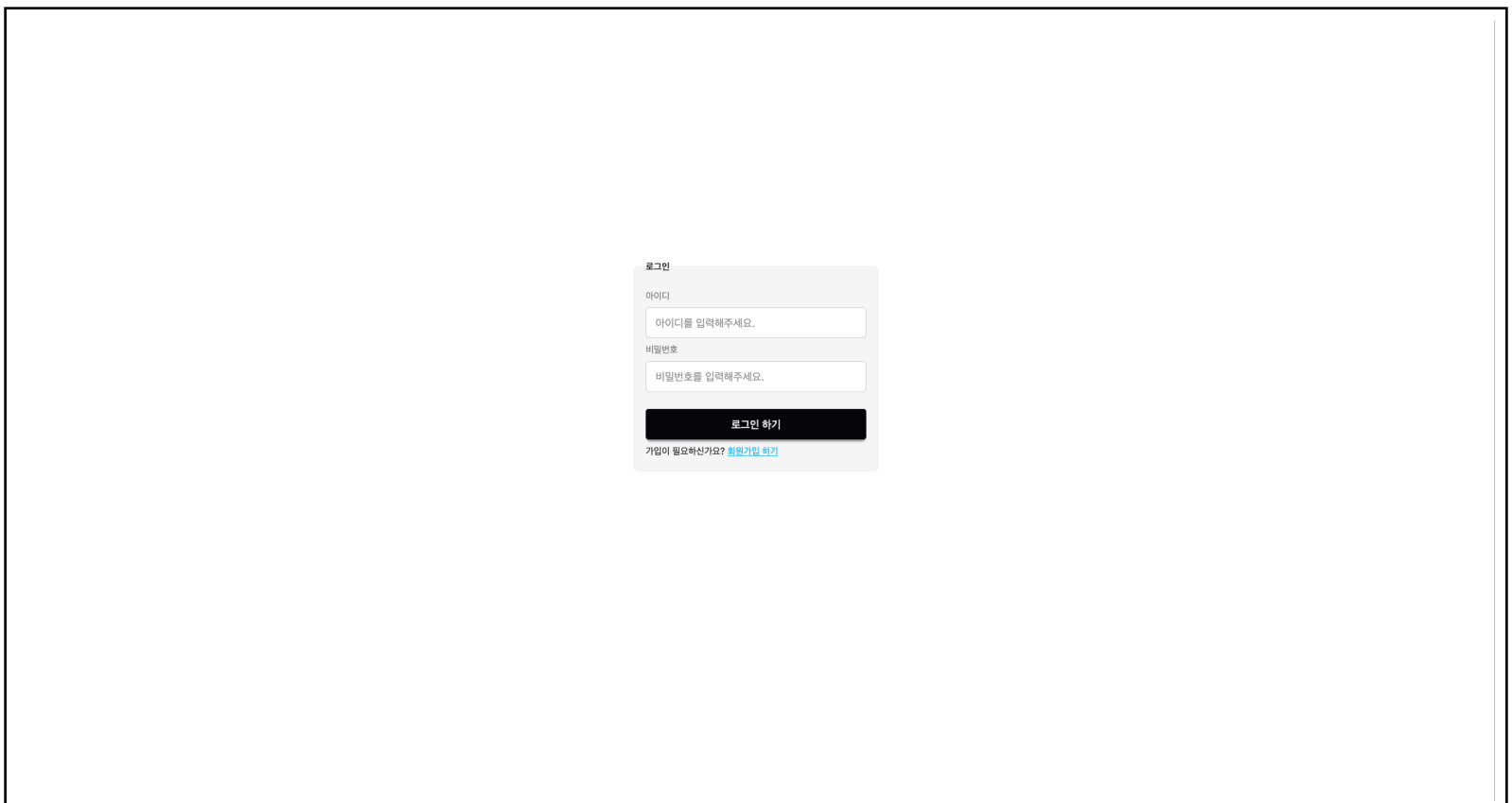
Design

- **코드 리뷰 기능**:
  - Linter/Formatter: Ruff (`subprocess`로 실행)
  - AI 리뷰: OpenAI o1-mini (자체 프롬프트와 조합)
- **프론트엔드**: TailwindCSS, DaisyUI
  - '페이지 당 단일 HTML' 원칙으로 단순성 추구

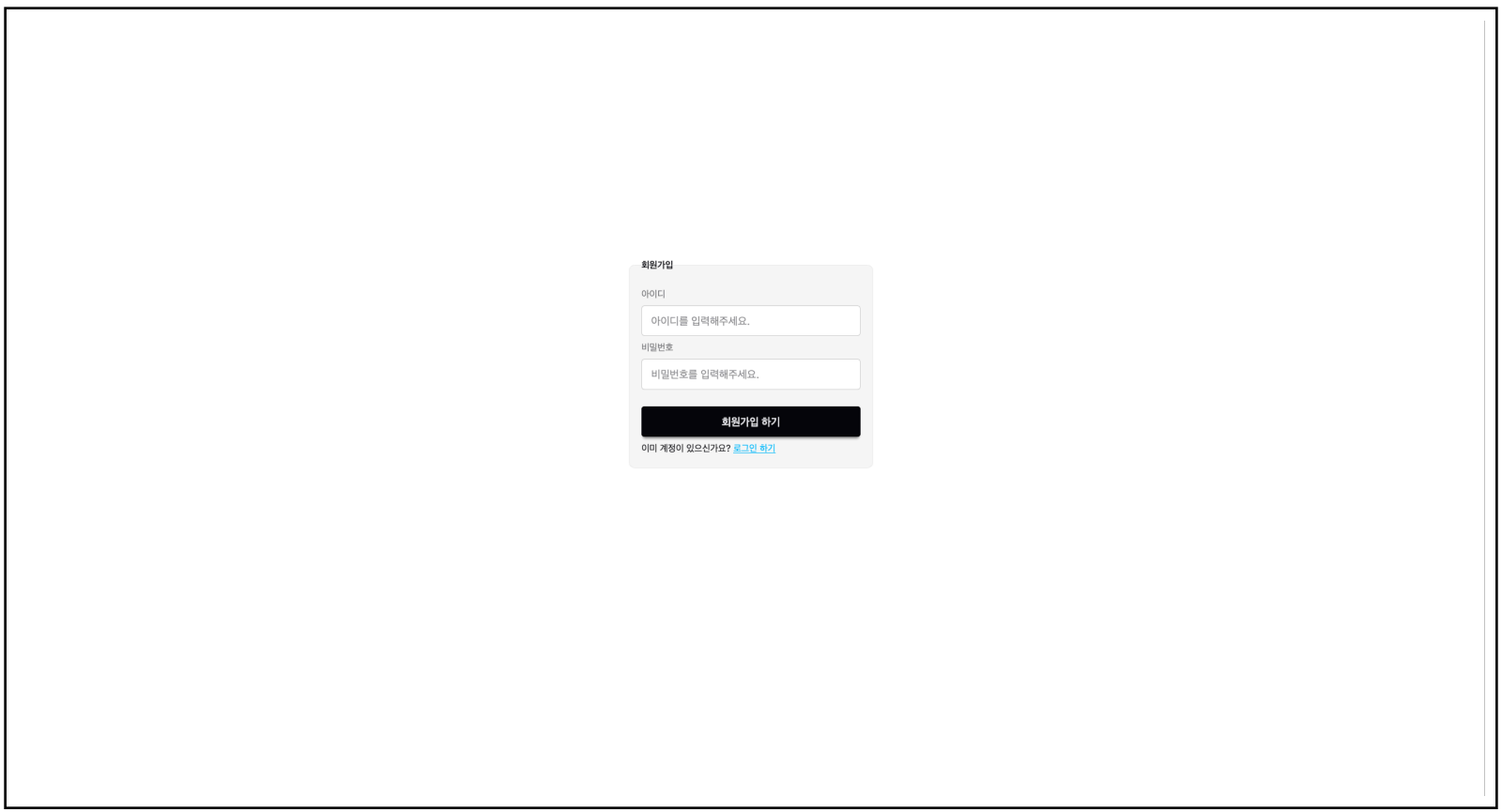
Implementation

- **디렉터리 구조**:
  - `/app`: FastAPI 서버 로직 (Python 코드)
  - `/frontend`: HTML 등 프론트엔드 리소스
- **계층적 설계**:
  - `main.py`의 라우터가 요청을 받고, `auth_helper`, `db_helper` 등 기능별로 분리된 모듈을 호출하여 로직을 처리

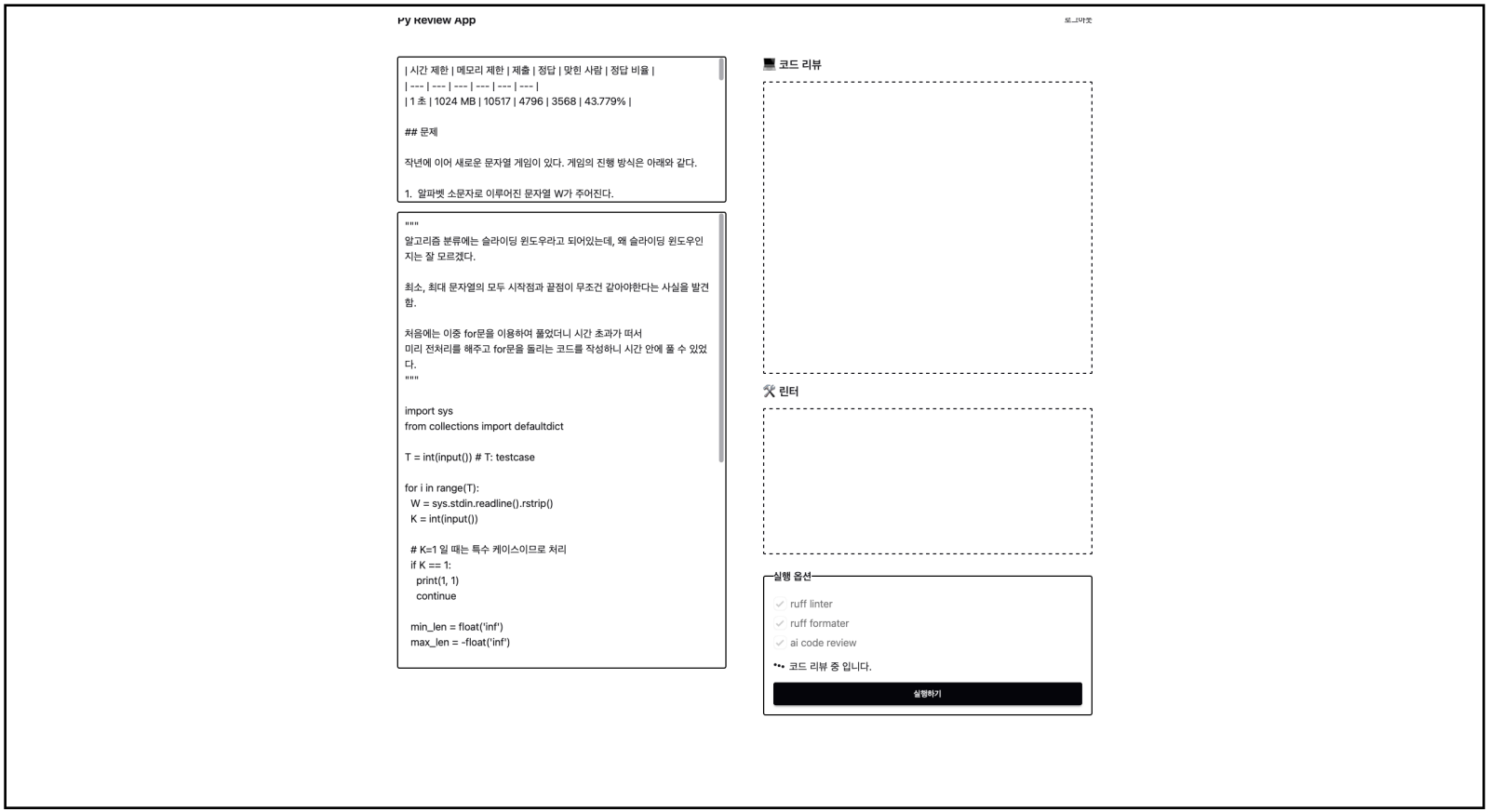
Implementation



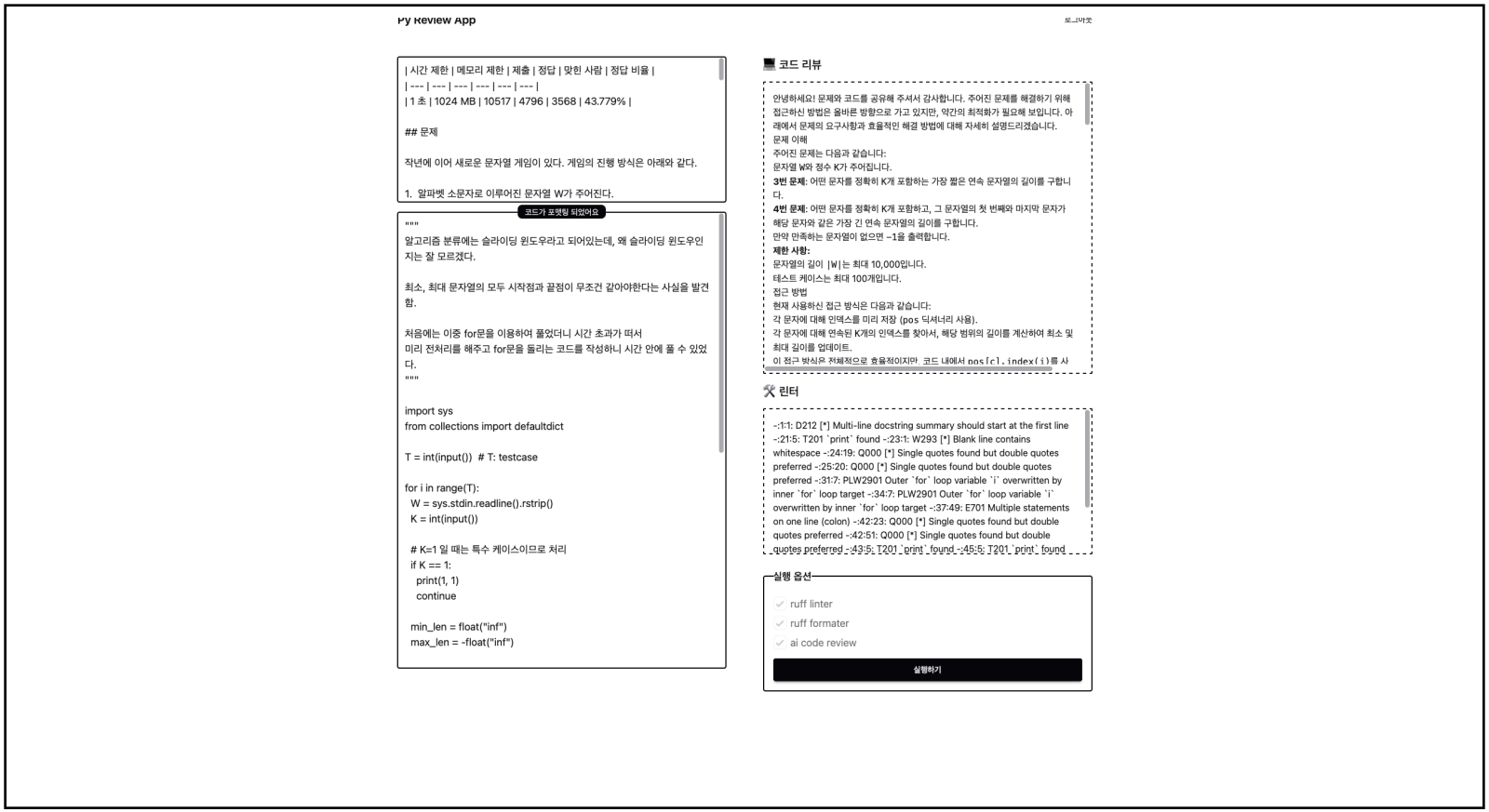
Implementation



Implementation



Implementation



Analysis & Discussion

- **데이터베이스와 ORM**:
  - **고민**: ORM 도입을 고려했으나, 프로젝트 규모에 비해 학습/설정 비용이 크다고 판단.
  - **해결**: `TinyDB` (NoSQL) + `Pydantic` (데이터 검증) 조합을 선택.
  - **구조**: Service <-> Pydantic <-> NoSQL 데이터 흐름을 만들어 ORM의 장점(데이터 유효성 검증)과 NoSQL의 단순함을 모두 취함.
  - **한계**: `TinyDB`는 단일 스레드 기반이므로, 멀티 프로세스 환경에서는 사용 불가.

Analysis & Discussion

- **프론트엔드**:
  - **목표**: 백엔드 로직에 집중하기 위해 프론트엔드는 최대한 단순하게 유지.
  - **구현**: React 같은 무거운 도구 대신, 페이지 당 단일 HTML 파일과 CSS 라이브러리(Tailwind, DaisyUI)를 활용하여 개발 속도 향상.

Analysis & Discussion

- **인증 구현 상세**:
  - **패스워드 보안**: 회원가입 시, 각 패스워드마다 `salt` 값을 부여하고 `pbkdf2_sha256`로 해싱. (레인보우 테이블 공격 방어 및 최신 GPU 환경에서 `bcrypt`보다 안전)
  - **성능**: 해싱은 CPU-bound 작업으로, 사용자 증가 시 서버 병목이 될 수 있음. 대규모 서비스에서는 별도 스레드에서 비동기 처리 필요. 현 프로젝트에선 미적용.
  - **세션 관리**: 메모리에 세션을 저장하므로 멀티 프로세스/스레드 환경에서는 세션 공유 불가. 확장 시, `Redis` 같은 외부 인메모리 DB를 세션 스토어로 사용해야 함.

Analysis & Discussion

- **LLM 비용 관리**:
  - 토큰 당 비용 문제를 해결하기 위해 서비스에 제한을 적용.
  - (1)관리자 인증
  - (2)하루 5회
  - (3)문제/코드 각 1500자 이하.

Conclusion

- Python 코드 리뷰 웹앱을 성공적으로 구현 완료.
- FastAPI, Pydantic 등을 활용하며 Python 웹 생태계의 견고함과 매력을 느낌.
- 단순한 앱이라도 모듈 분리 등 좋은 설계의 중요성을 체감.
- 알고리즘 문제 풀이를 넘어, 실제 서비스를 만들며 언어에 대한 이해도를 높이는 계기가 됨.

Source Code: https://github.com/cjaewon/pyreview-app