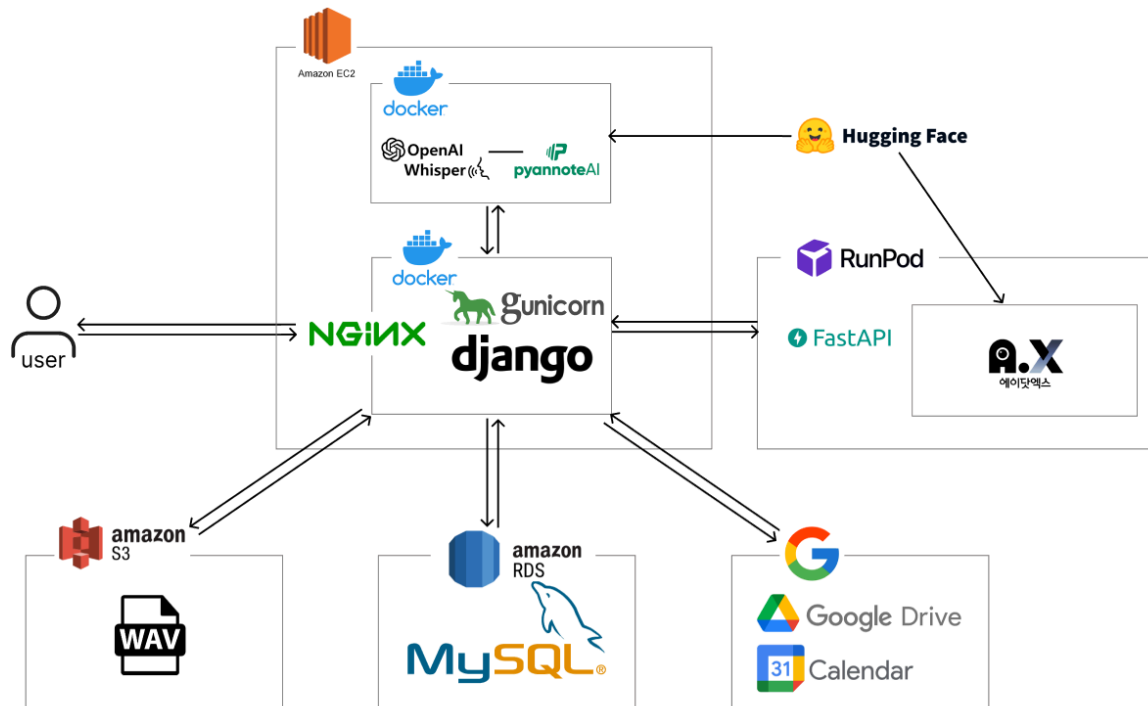


시스템 아키텍처

2025년 11월 17일

1. 시스템 아키텍처



2. 시스템 설계 목표 및 전략

- 시스템 설계 목표:

1. 확장성 있는 시스템 설계가 가능하도록 함
2. 데이터를 안전하게 저장하고 관리할 수 있도록 함
3. 스케줄 및 일정 관리가 가능하도록 함

- 시스템 설계 전략:

1. 확장성 있는 시스템 설계가 가능하도록 함
 - a. Amazon EC2를 활용하여 서버를 호스팅하고, Docker를 통해 시스템의 구성요소들을 컨테이너화하여 배포 및 관리 효율성을 극대화
 - b. Django를 사용하여 사용자 인터페이스를 구축하고, Gunicorn을 통해 웹 서버의 성능을 최적화한다.
 - c. Nginx를 통해 서버의 트래픽을 효율적으로 분배한다.
2. 데이터를 안전하게 저장하고 관리할 수 있도록 함

- a. Amazon S3를 사용해 음성 파일을 안전하게 저장하고 관리할 수 있도록 한다.
- b. Amazon RDS(MySQL)를 사용하여 음성 파일과 회의 관련 메타데이터를 중앙에서 관리하고, 파일에 대한 검색 및 관리 작업을 빠르게 처리할 수 있도록 한다.
- 3. 스케줄 및 일정 관리가 가능하도록 함
 - a. Google Calendar API와 Google Drive API를 통해 추출한 태스크를 구글캘린더에 반영하고, 요약본을 드라이브에 자동저장되도록 한다.

3. 구성

- 사용한 모델

1. openai/whisper-medium
2. pyannote/speaker-diarization-3.1
3. skt/A.X-4.0-Light

- 구성 요소

1. 클라이언트(Client):
 - 컴포넌트:
 - Chrome 브라우저
 - Docs
 - Chrome Extensions
 - 인터페이스:
 - Google Cloud OAuth 2.0 (인증)
 - JSON 데이터 전송 (HTTPS)
2. AWS Cloud:
 - EC2 인스턴스:
 - Nginx (리버스 프록시 및 정적 파일 제공)
 - Unicorn (WSGI 서버)
 - FastAPI (백엔드 애플리케이션)
 - S3 버킷
3. 데이터베이스:
 - Amazon RDS (MySQL)
4. 외부 서비스:
 - RunPod (sLLM 실행 환경)

- 설명

- 클라이언트는 Google Cloud OAuth 2.0을 통해 인증을 수행하고, JSON 데이터를 HTTPS를 통해 AWS Cloud로 전송.
- AWS Cloud의 EC2 인스턴스는 요청을 처리하며, Nginx와 Unicorn을 통해 FastAPI 애플리케이션이 실행.
- Amazon RDS는 음성 파일의 메타데이터를 저장하고 관리하며, RunPod는 AI 모델 실행을 통해 텍스트 분석을 처리.
- 최종적으로 처리된 데이터는 클라이언트로 반환.