HYPERTEAMS 4회차 · 85분

확장하기

두 개의 API 표면 · REST · 모델 갖추기 · 음성과 이미지

발표자 노트 · 01

여는 말

- 여기서 성격이 바뀝니다. 앞은 화면에서 쓰는 법, 여기는 다른 프로그램이 이 컴퓨터를 부르게 하는 법입니다.
- 문을 여는 이야기라 안전이 먼저입니다.

화면에서 쓰는 법이 아니라,
다른 프로그램이 이 컴퓨터를 부르는 법입니다

그래서 편의보다 안전이 먼저입니다 — 두 표면 모두 기본값은 꺼짐입니다

발표자 노트 · 02

성격이 바뀝니다

- 앞의 세 회차는 “내가 쓴다”였고, 이 회차는 “남이 부른다”입니다. 청중에게 이 전환을 명시하세요.
- 켠 적 없는 설치는 401 이 아니라 404 를 냅니다 — API 의 존재 자체를 숨깁니다.
- 이 회차를 듣고 아무것도 안 켜는 것도 정상적인 결론입니다.

표면

바깥으로 열리는 문은 둘뿐

/api/v1

무엇을
업무를 시킨다

인증
Bearer

기본값
꺼짐

단계
켜짐 / 꺼짐

/api/ai

무엇을
✓ 모델을 부른다

인증
✓ Bearer

기본값
✓ 꺼짐

단계
✓ off · inference · full

그 밖의 모든 /api/* 는 대시보드 쿠키 전용 — 바깥에서 못 부릅니다.

발표자 노트 · 03

두 개의 표면

- 켠 적 없는 설치는 401이 아니라 404입니다 — API의 존재 자체를 숨깁니다.
- 모델 표면은 inference 부터. full 은 내 모델 서버를 관리할 권한을 주는 것입니다.

모델 표면

얼마나 열지를 고릅니다

모드

OFF
기본값

INFERENCE
여기서 시작

FULL
관리까지

열리는 것

OFF
✓ 아무것도. 404

INFERENCE
✓ 추론만 — 대화 · 임베딩 · 음성 · 이미지

FULL
✓ 상류 모델 서버의 관리 경로까지

경로가 범위를 강제하지 않습니다. .../api/ai/v1 을 안내했다고 그 아래로만 갈 수 있는 게 아닙니다 — 범위를 정하는 것은 모드입니다.

발표자 노트 · 04

3단계

- inference 가 여는 목록은 정해져 있습니다 — models · chat · completions · embeddings · moderations · audio 셋 · images 셋.
- 그 목록 밖은 inference 에서 404 입니다. 모델을 내려받거나 서버를 재시작하는 것은 full 에서만 열립니다.
- full 은 그 프로그램에게 내 모델 서버 관리 권한을 주는 것입니다. 최소 권한이 여기도 그대로 적용됩니다.
- base 주소가 둘인 것도 여기서 짚으세요 — OpenAI 호환 도구는 …/api/ai/v1, 다른 대시보드는 …/api/ai.

열쇠가 곧 대시보드 비밀번호입니다

그래서 비밀번호가 없으면 켜지지 않고, 값이 복사되는 곳마다 위험이 늘어납니다

발표자 노트 · 05

열쇠

- 편집기 설정 · 스크립트 · 환경변수 · CI — 하나만 새도 대시보드가 통째로 열립니다.
- 켜기 전에 '이 열쇠가 붙을 곳을 셀 수 있나' 를 자문하게 하세요.

켜는 순서

한 곳에서 한 번 성공시키고 멈추세요

1

비밀번호

hyperteams setup — 비밀번호가 없으면 애초에 켤 수 없습니다

2

필요한 표면만

업무를 외부에서 시킬 것 → REST · 모델을 빌려줄 것 → inference

3

열쇠 목록

그 값을 넣을 곳을 종이에 적습니다. 셀 수 없으면 켜지 마세요

4

한 곳만

한 번 성공하면 그다음은 복사입니다

처음부터 여러 곳에 붙이면 어디가 틀렸는지 못 찾습니다. 4번이 이 슬라이드의 핵심입니다.

발표자 노트 · 06

켜는 순서

- 3번을 실제로 시켜보세요 — 편집기 설정 · 스크립트 · 환경변수 · CI 로 금세 넷이 됩니다.
- 명세는 설정 화면에서 확인할 수 있습니다. 외우게 하지 마세요.
- "켜두면 편하니까 둘 다" 가 가장 흔한 유혹입니다. 쓰지 않는 표면은 끄는 것이 기본입니다.

REST

엔드포인트는 여섯 개가 전부입니다

01

GET /workspaces

작업 폴더 목록 — 여기서 번호를 얻습니다

02

GET /tasks

업무 목록

03

POST /tasks

업무 만들기

04

GET /tasks/{id}

업무 하나 보기

05

POST .../follow-up

이어 말하기

06

POST .../stop

중지

전부 /api/v1 아래입니다. 화면에서 하던 것과 같은 함수를 부르므로 결과도 화면에 그대로 보입니다.

발표자 노트 · 07

엔드포인트

- 작업 폴더 번호를 하드코딩하지 말라고 여기서 못 박으세요 — GET /workspaces 로 조회합니다.
- 실습 1단계가 이 조회입니다. 200 이 오면 토큰은 정상이라는 뜻이라 문제를 절반으로 줄여줍니다.

REST

업무를 코드로 시키기

- POST /api/v1/tasks — 화면에서 만든 업무와 같은 함수
- follow-up 으로 이어 말하고 stop 으로 멈춤
- mode 를 빼면 전부 자동입니다
- workspaceId 는 하드코딩 말고 경로로 조회

멱등 키가 없습니다 — 같은 요청을 두 번 보내면 업무가 두 개 생깁니다.

발표자 노트 · 08

REST

- 타임아웃이 곧 실패가 아닙니다. 재시도 전에 목록을 조회하거나 name 에 고유값을 넣으세요.
- 의도된 설계입니다 — 없는 보장을 있는 척하지 않는다는 것.

멱등 키가 없습니다

중복 방지는 부르는 쪽 책임입니다

1
재시도 전에 조회
GET /api/v1/tasks 로 최근 목록을 먼저 확인

2
name 에 고유값
“일일점검-2026-08-15” — 이미 있는지 알아볼 수 있게

3
재시도 상한
무한 재시도 금지

4
파일을 고치는 업무면
재시도를 아예 사람 확인으로

타임아웃이 곧 실패는 아닙니다 — 요청은 도착했는데 응답만 못 받았을 수 있습니다. 그 상태의 재시도가 같은 일을 두 번 실행시킵니다.

발표자 노트 · 09

멱등 키

- 빠뜨린 게 아니라 의도한 설계입니다 — “없는 보장을 있는 척하지 않는다”.
- 2번이 가장 실용적입니다. 날짜나 실행 식별자를 이름에 넣으면 조회 한 번으로 끝납니다.
- 실습 4단계에서 실제로 두 개가 생기는 것을 보게 됩니다. 여기서 미리 예고하세요.

모델

내 컴퓨터를 모델 서버로

- 대화 · 임베딩 · 음성 · 이미지가 같은 규격으로
- 나가면 안 되는 데이터를 로컬에서 처리
- 반복 · 대량 호출은 건당 비용이 없음
- 복제 목소리는 OpenAI 규격 밖 — 동의가 전제입니다

2줄

base 주소와 열쇠

OpenAI SDK 코드는 그대로

발표자 노트 · 10

모델 API

- 분류 · 태깅 · 임베딩처럼 반복 많고 난이도 낮은 것이 로컬에 잘 맞습니다.
- 508 이 나오면 주소가 자기 자신인지 보세요 — 루프 방지 장치입니다.

로컬 적합성

무엇을 내 컴퓨터로 돌릴 것인가

작업	판단
높음 분류 · 태깅 · 임베딩	높음 ✓ 반복 많고 난이도 낮음 · 대량 처리
높음 음성 인식 · 합성	높음 ✓ 건당 비용 없이 길이만큼
중간 긴 문서 추론	중간 ✓ 모델과 하드웨어에 달림
낮음 복잡한 코드 작성	낮음 ✓ 상용 모델이 앞섭니다

모델 라우팅의 로컬판입니다 — 난이도에 맞춰 고르되, 선택지에 “내 컴퓨터”가 하나 더 생긴 것입니다.

발표자 노트 · 11

무엇을 로컬로

- “그럼 상용 API 는 이제 필요 없나요”가 반드시 나옵니다. 이 표가 답입니다.
- 나가면 안 되는 데이터는 적합성과 별개로 로컬이 답입니다 — 그때는 느려도 그쪽입니다.
- 벤치마크 목적으로도 좋습니다. 같은 코드로 base 만 바꿔 비교할 수 있습니다.

붙일 때

실패하는 자리는 정해져 있습니다

01

모델 이름이 다릅니다

상용 API 이름을 그대로 쓰면 없습니다. /v1/models 로 먼저 목록 확인

02

응답 시간이 다릅니다

상용 기준 타임아웃이 그대로면 실패합니다. 한 번 재보고 여유를

03

508 이 나옵니다

주소가 자기 자신입니다. 버그가 아니라 무한 루프를 막는 장치

04

inference 인데 404

관리 경로를 부른 것입니다. 정말 필요하면 full 이지만 권한이 커집니다

네 가지 모두 열쇠 문제가 아닙니다. 401 이 아닌 오류를 인증 문제로 오해하는 데서 시간이 가장 많이 샙니다.

발표자 노트 · 12

실패하는 곳

- 이 네 줄을 실습 전에 띄워두면 질문의 대부분이 여기서 해결됩니다.
- 508 은 홉마다 소켓이 쌓이는 것을 막는 장치입니다. 안 끊으면 컴퓨터가 멈춥니다.

모델 갖추기

받기 전에 이 PC에서 되는지부터

01

서버 — 둘 중 하나

이 컴퓨터에서 돌리기 / 이미 도는 주소에 붙기

02

프리셋 다섯

Whisper(음성→글) · Fish Speech(글→음성) · 대화 둘 · 이미지 하나

03

프리플라이트

메모리와 디스크를 따로 봅니다. "빠듯합니다"를 가볍게 보지 마세요

04

컨텍스트 = 메모리

8k는 1.3GiB, 128k는 21GiB. 무조건 크게가 아닙니다

엔진은 직접 고르지 않습니다. 설치 시점에 하드웨어에 맞춰 정해지므로 CPU 경고에는 '수리 다시 실행'입니다.

로컬 모델

- 목록이 짧은 것은 의도된 것입니다 — 모르는 사람에게 1600개를 보여 주는 것은 도움이 아닙니다.
- 무엇부터 받나 — 앞뒤 장이 실제로 쓰는 것부터입니다. 회의 녹음이면 Whisper(1.6GB)가 가장 작습니다.
- "동작하지만 여유가 빠듯합니다"는 여유가 필요량의 1.25배 아래라는 뜻입니다. OS와 브라우저가 이미 먹고 있습니다.
- 컨텍스트는 서버 설정 하나로 넘기고, 바꾼 값은 다음 기동부터 적용됩니다.

음성 · 이미지

오래 걸리고, 실패하면 부분만 다시 만듭니다

- 긴 원고는 문단으로 나눠 만드세요 — 실패한 문단만 다시 만들 수 있습니다
- 이미지 프롬프트는 파일로 — 재현 · 수정 · 일관성 셋이 여기서 나옵니다
- 낮은 해상도로 여러 장 → 고른 것만 확대(upscale)가 시간과 비용에 맞습니다
- 입력과 출력을 폴더로 나누고, 출력은 git에서 제외하세요

복제 목소리는 OpenAI 규격 밖입니다. 목소리는 개인정보이고 동의가 전제입니다 — 내부 자료여도 마찬가지입니다.

음성과 이미지

- `--output`을 빠뜨리면 터미널에 이진 데이터가 쏟아집니다. 흔한 실수입니다.
- 음성→글은 회의 녹음을 회의록으로 만드는 파이프라인의 첫 단계입니다 — 모델 API와 업무 API를 잇는 형태가 실무에서 가장 잦습니다.
- 이미지 수십 장은 십 분 이상 걸립니다. 예약 주기를 잡을 때 이 감각이 필요합니다.
- 파일명에 순번·날짜를 넣으세요. 순서가 중요한 파이프라인에서 필수입니다.

실습 · 40분

REST로 업무를 걸어보기

1
읽기부터
조회가 200 이면 토큰은 정상.
인증과 요청 문제를 분리합니다

2
읽기만 하는 업무
화면에도 같은 업무가 보이는지 —
두 표면이 같은 것을 봅니다

3
진행 읽기
조회 간격을 적어두세요. 자동화할
때 그대로 설계값이 됩니다

4
두 번 보내기
업무가 두 개 생깁니다 — 멱등
키가 없습니다

타임아웃은 실패가 아니라 응답이 늦은 것일 수 있습니다. 재시도는 실패가 확인됐을 때만 — 이게 가장 자주 사고를 냅니다.

발표자 노트 · 15

실습 진행

- 4단계를 반드시 시키세요. 읽기 업무라 지금은 무해하지만 메일 발송이었으면 두 통이 나갑니다.
- 중복 방지는 호출하는 쪽 책임이라는 것을 못 박으세요.
- 토큰이 쉘 히스토리에 남지 않았는지 정리 단계에서 확인시키세요.

정리

가져갈 것

1. 문은 둘, 기본은 꺼짐, 열쇠는 대시보드 비밀번호
2. REST 에 멱등 키가 없다 — 재시도는 호출자 책임
3. OpenAI 호환이라 base 주소 한 줄로 갈아탄다
4. 모델은 프리플라이트를 보고 작은 것부터 받는다
5. 컨텍스트를 최대로 올리는 것은 메모리를 태우는 일이다
6. 음성·이미지는 조각으로 만들고 프롬프트를 파일로 남긴다

발표자 노트 · 16

닫는 말

- 여기까지가 “바깥으로 여는 문”과 “내 컴퓨터를 모델 서버로”입니다.
- 다음 회차는 이것들을 예약으로 엮어 실제 파이프라인을 만드는 실전 자동화입니다.