AI로 만들었는데 서버가 죽었습니다

바이브 코더를 위한 스케일링 생존 가이드

강의 핵심

1. AI가 앱을 만드는 시대
2. 하지만 서비스는 트래픽, 비용, 장애를 만난다
3. 이 강의의 목표: 버티는 구조를 읽는 눈

작동하는 것과 버티는 것은 다르다

데모는 동작을 증명하고, 서비스는 지속성을 증명한다.

강의 핵심

1. 작동은 기능의 문제
2. 버팀은 구조와 운영의 문제
3. 처음부터 둘을 구분해야 한다

개발의 출발선이 달라졌다

AI는 만들기의 문턱을 낮췄지만, 운영의 비용을 없애지는 않았다.

강의 핵심

1. AI는 시작 속도를 바꿨다
2. 배포가 쉬워지면 검토도 쉽게 생략된다
3. 운영 감각은 별도로 배워야 한다

나는 지금 무엇을 만들었나?

화면, 앱, 서비스, AI 앱, 에이전트, 플랫폼

강의 핵심

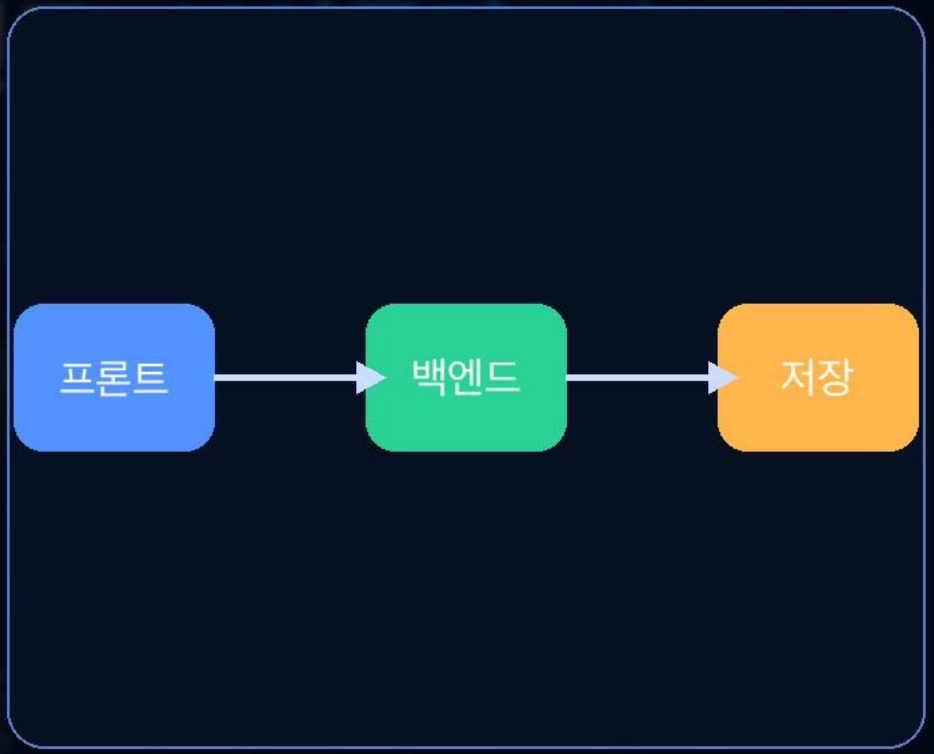
1. 화면은 보이는 층
2. 앱은 입력과 상태의 단위
3. 서비스는 저장, 인증, 배포, 운영까지 포함한다

반드시 보여야 하는 세 층

누가 보여주고, 누가 판단하고, 누가 기억하는가?

강의 핵심

1. 프론트엔드: 사용자가 경험하는 층
2. 백엔드: 규칙과 판단이 모이는 층
3. 저장 계층: 서비스의 기억



클릭 한 번은 여러 사건이다

버튼 하나가 시스템 전체를 통과한다.

강의 핵심

1. 하나의 클릭은 여러 계층을 지난다
2. 인증, 저장, 외부 API가 함께 움직인다
3. 느린 곳을 찾으려면 흐름을 그려야 한다

작은 성공이 숨은 부채를 드러낸다

어제 괜찮았던 이유는 어제가 작았기 때문이다.

강의 핵심

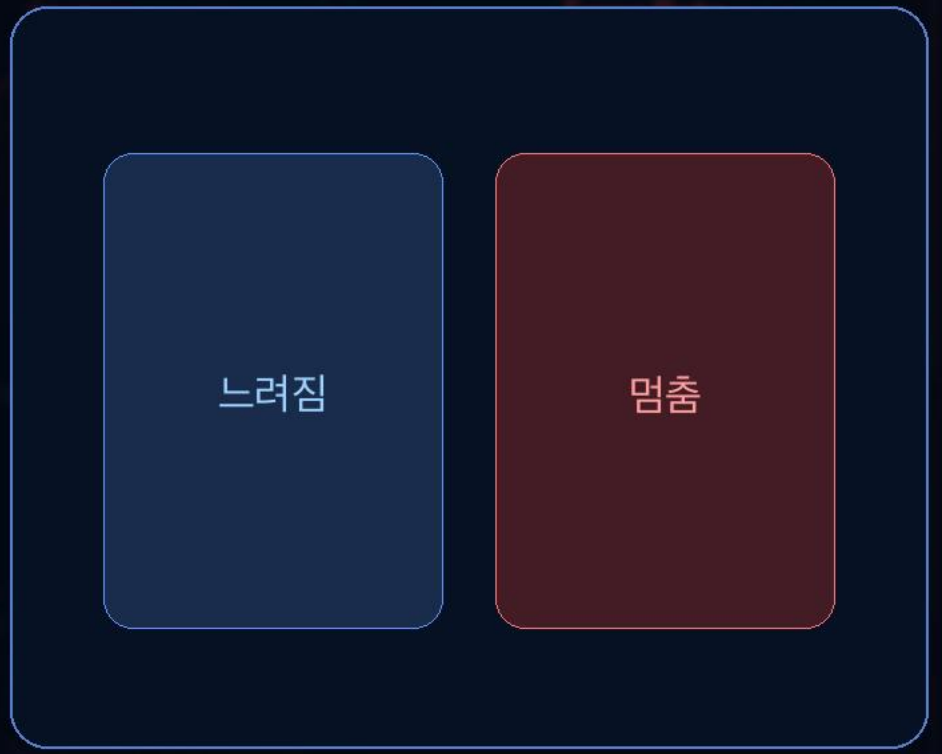
1. 작을 때는 비효율이 숨어 있다
2. 바이럴은 축복이자 부하 테스트
3. 성공이 구조의 한계를 드러낸다

느려짐과 멈춤은 다른 사건이다

지연, 한계, 장애, 배포 문제는 대응이 다르다.

강의 핵심

1. 느려짐은 병목을 찾는 문제
2. 멈춤은 한계치와 오류를 찾는 문제
3. 증상을 먼저 분류해야 대응이 보인다



대시보드는 조기 경보 시스템이다

사용자가 실패를 읽기 전에 운영자가 지표를 읽어야 한다.

강의 핵심

1. 대시보드는 장애 후 확인 도구가 아니다
2. 응답 시간, 오류율, 읽기 수, 비용 추이를 본다
3. 숫자는 두려움이 아니라 정보다

프론트엔드는 첫 경험이다

React, Vite, Next.js, CSR, SSR, SEO는 사용자 경험의 선택이다.

강의 핵심

1. 렌더링 방식은 사용자 경험에 직접 닿는다
2. SSR, CSR, SEO는 유행어가 아니라 선택지
3. 첫 경험이 서비스 신뢰를 만든다

첫 화면 속도는 제품 결정이다

무엇을 먼저 보여주느냐가 신뢰를 만든다.

강의 핵심

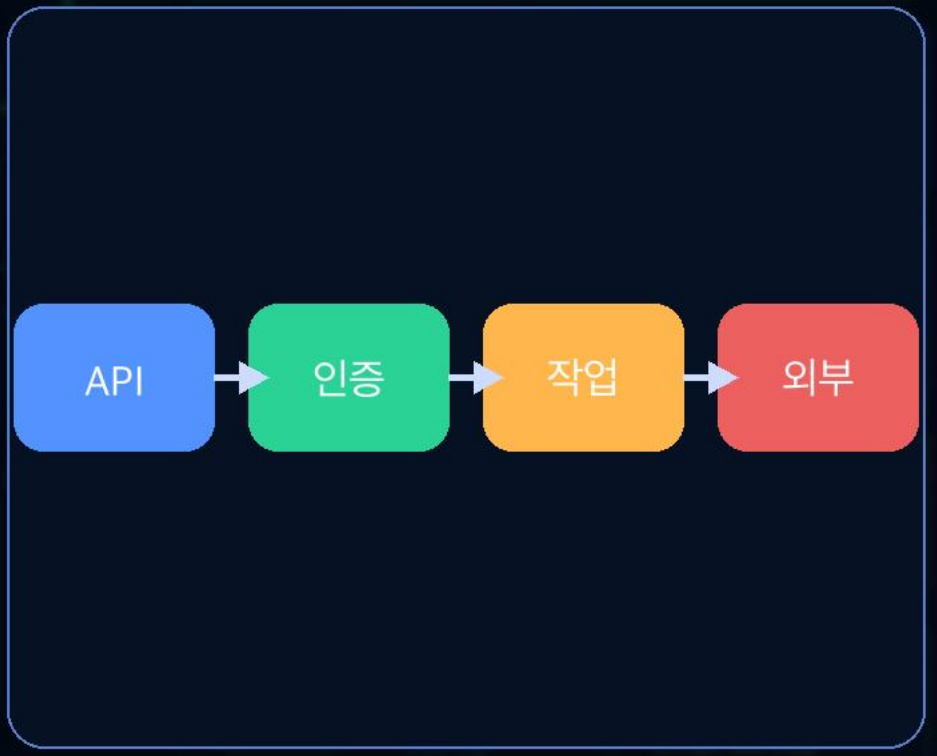
1. 사용자는 빈 화면을 신뢰하지 않는다
2. 먼저 보여줄 것을 설계해야 한다
3. 속도는 기술이면서 제품 판단이다

백엔드는 규칙이 사는 곳이다

API, 웹훅, 인증, 작업, 외부 서비스가 행동을 정의한다.

강의 핵심

1. 백엔드는 데이터 전달 통로가 아니다
2. 권한, 검증, 외부 연동, 실패 처리를 맡는다
3. 서비스의 행동은 백엔드에서 결정된다

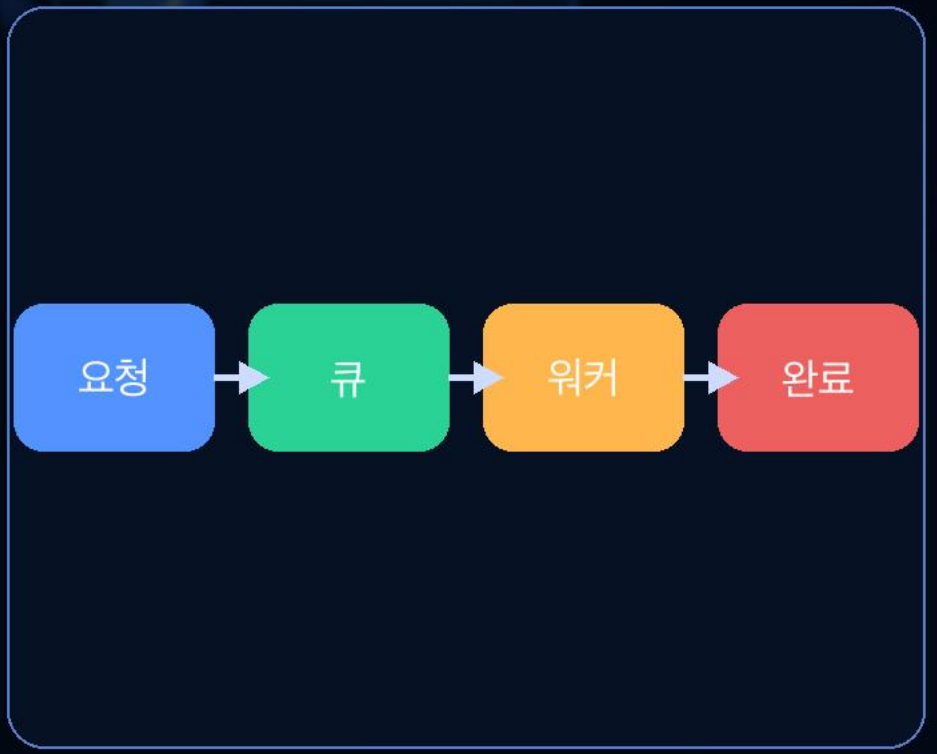


오래 걸리는 일은 요청에서 빼야 한다

지금은 빠르게 응답하고, 나중에 백그라운드에서 처리한다.

강의 핵심

1. 긴 작업은 요청 시간을 붙잡지 않는다
2. 응답과 처리를 분리한다
3. 큐와 워커는 사용자 경험을 지킨다

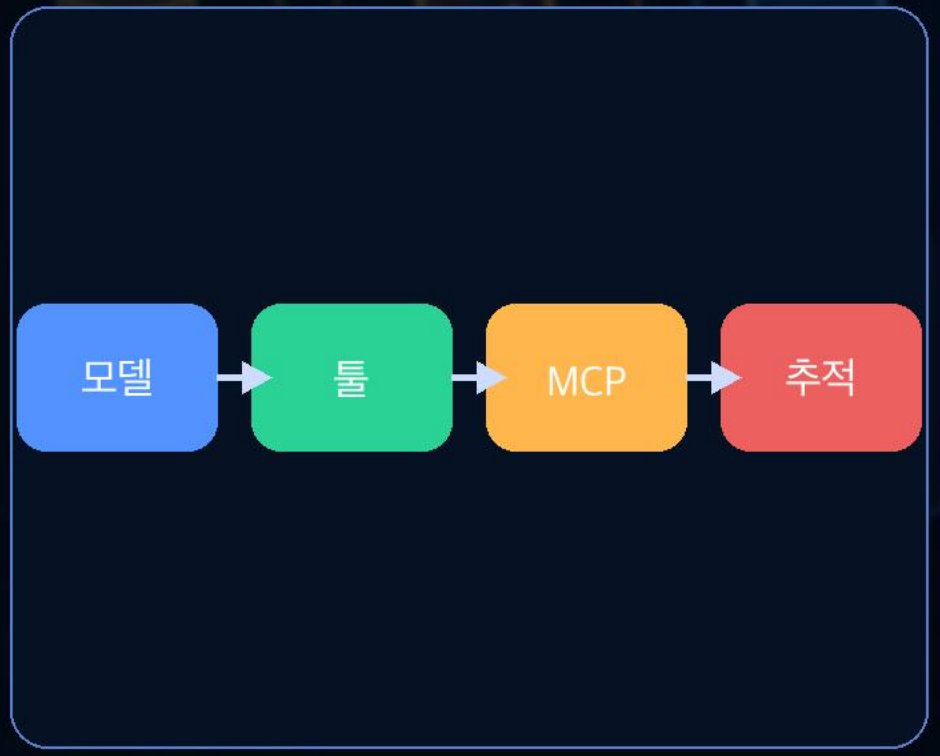


AI 백엔드는 새로운 층을 더한다

모델 호출, 툴, MCP, 메모리, 승인, 추적

강의 핵심

1. AI 백엔드는 모델 호출만이 아니다
2. 툴, MCP, 메모리, 승인, 추적이 필요하다
3. 에이전트가 될수록 운영면이 넓어진다



에이전트는 챗봇이 아니다

목표를 향해 단계, 도구, 상태를 이어간다.

강의 핵심

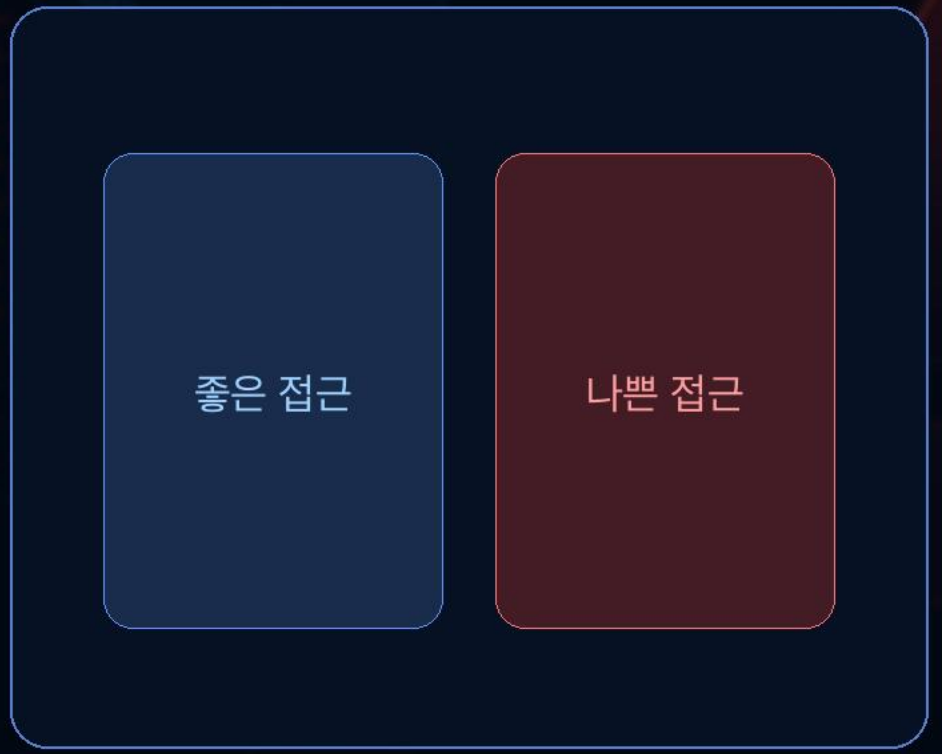
1. 챗봇은 답변하고 끝난다
2. 에이전트는 목표를 위해 행동을 이어간다
3. 상태와 도구 사용이 핵심 차이다

대부분의 병목은 저장소에서 시작된다

데이터가 적을 때는 나쁜 접근 패턴이 숨어 있다.

강의 핵심

1. 데이터가 적을 때는 나쁜 쿼리도 빨라 보인다
2. N+1, 전체 스캔, 읽기 과다가 먼저 드러난다
3. 저장소는 성능과 비용의 공통 원인이다

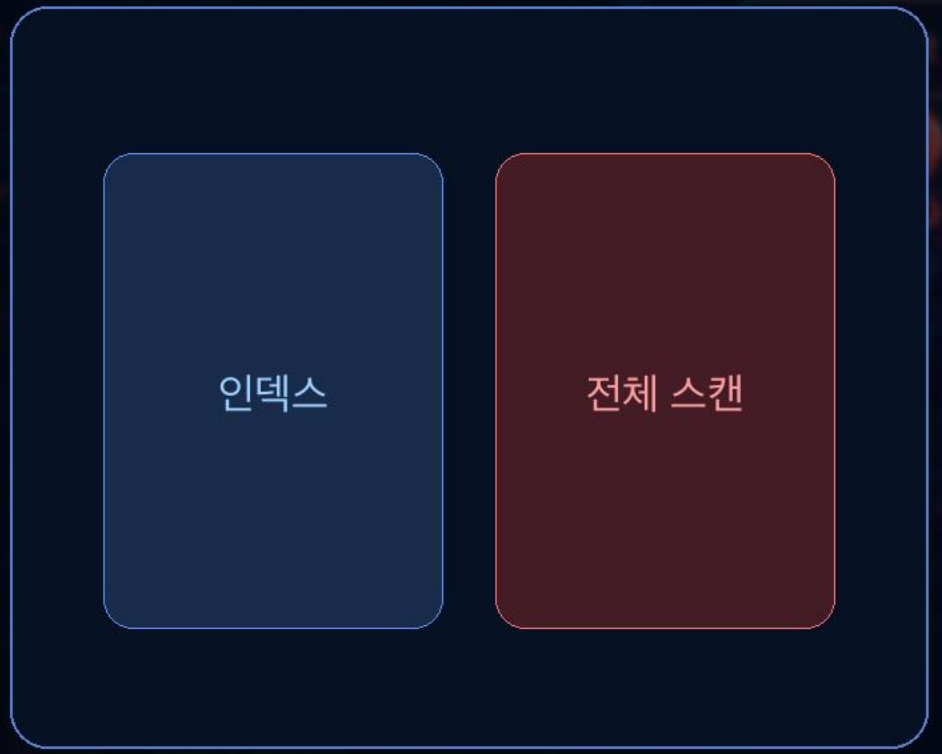


인덱스는 데이터 속의 길이다

길이 없으면 쿼리는 도시 전체를 걸어야 한다.

강의 핵심

1. 인덱스는 미리 깔아둔 길이다
2. 길이 없으면 데이터 전체를 훑는다
3. 실행 계획은 DB의 속마음을 보여준다

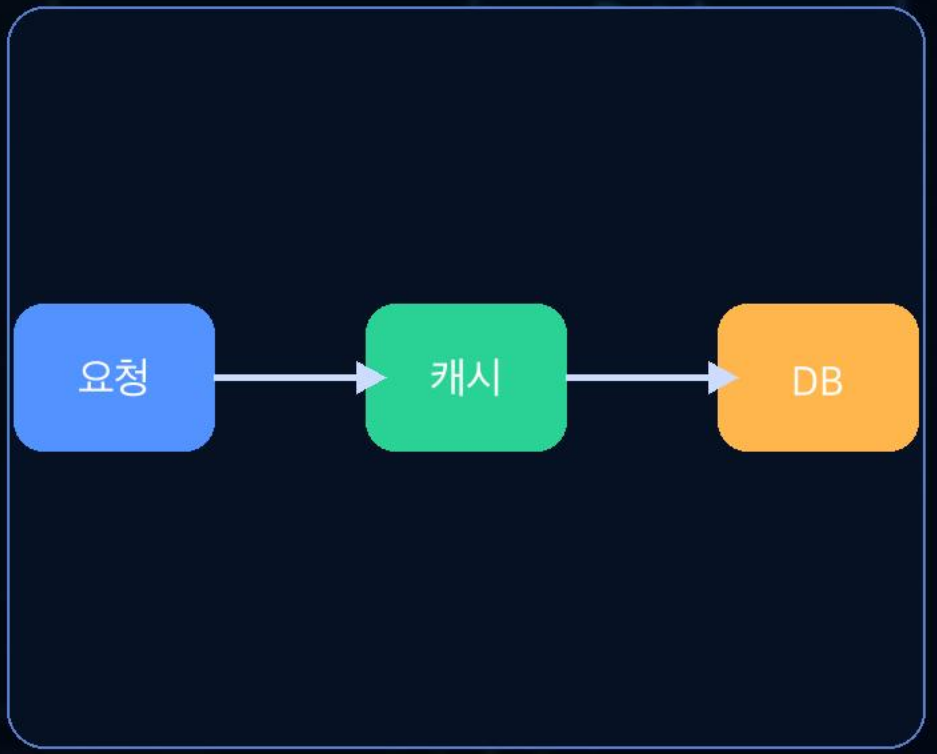


캐시는 덜 묻는 기술이다

반복이 사라지기 때문에 빨라진다.

강의 핵심

1. 캐시는 같은 질문을 덜 하게 만든다
2. 속도와 비용을 동시에 바꾼다
3. 모든 것을 캐시하면 안 된다

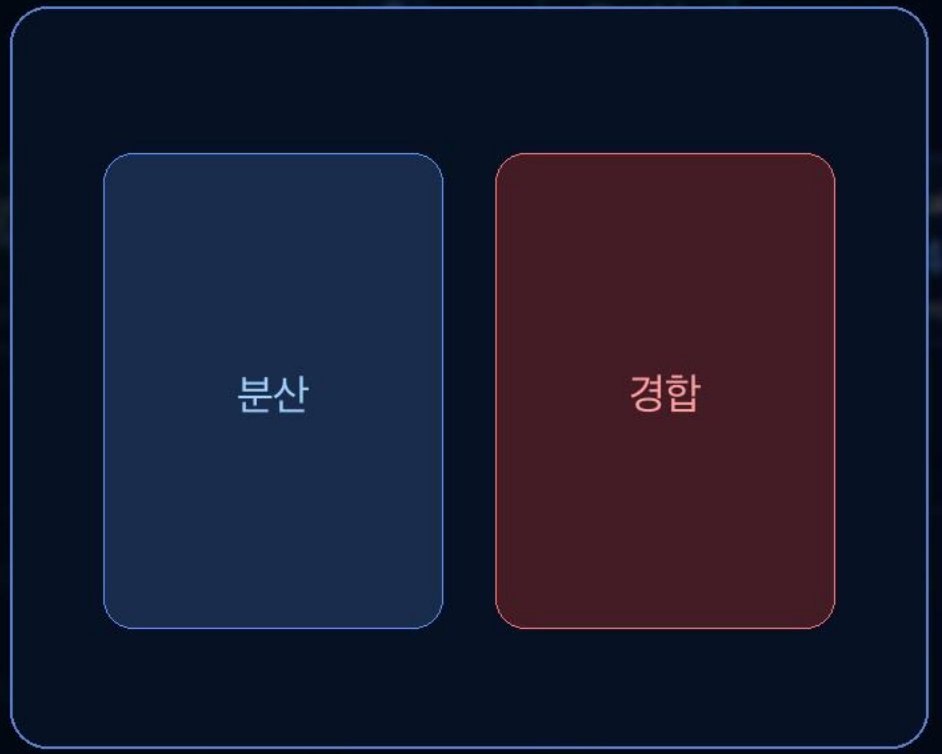


쓰기는 다르게 무너진다

많은 사용자가 하나의 값을 바꾸면 정확성이 흔들린다.

강의 핵심

1. 읽기 문제와 쓰기 문제는 다르다
2. 하나의 값에 쓰기가 몰리면 정확성이 흔들린다
3. 큐, 샤딩, 분산 카운터를 고려한다



인프라는 선택지의 묶음이다

VPS, 서버리스, 컨테이너, 엣지는 서로 다른 문제를 푼다.

강의 핵심

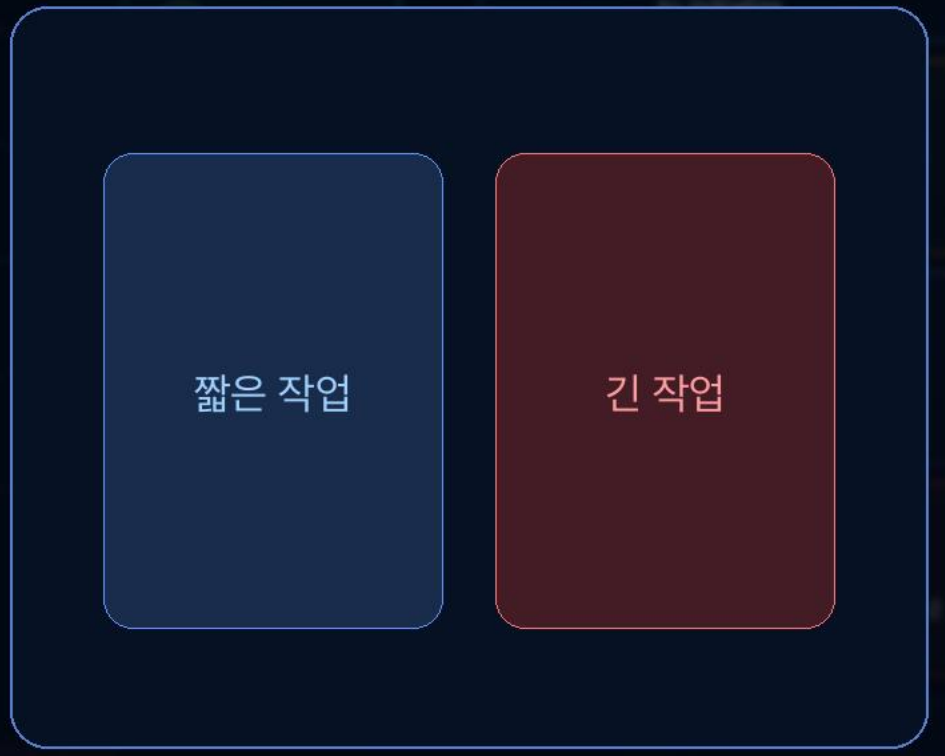
1. 서버는 장소보다 방식에 가깝다
2. VPS, 서버리스, 컨테이너, 엣지는 각자 강점이 다르다
3. 하나만 고르는 문제가 아니다

서버리스는 기본값이지 신앙이 아니다

짧은 폭주에는 강하지만, 긴 작업과 숨은 한계에는 약하다.

강의 핵심

1. 서버리스는 웹앱의 좋은 출발점
2. 하지만 타임아웃과 동시성 한계가 있다
3. 긴 작업과 상태 유지에는 다른 구조가 필요하다

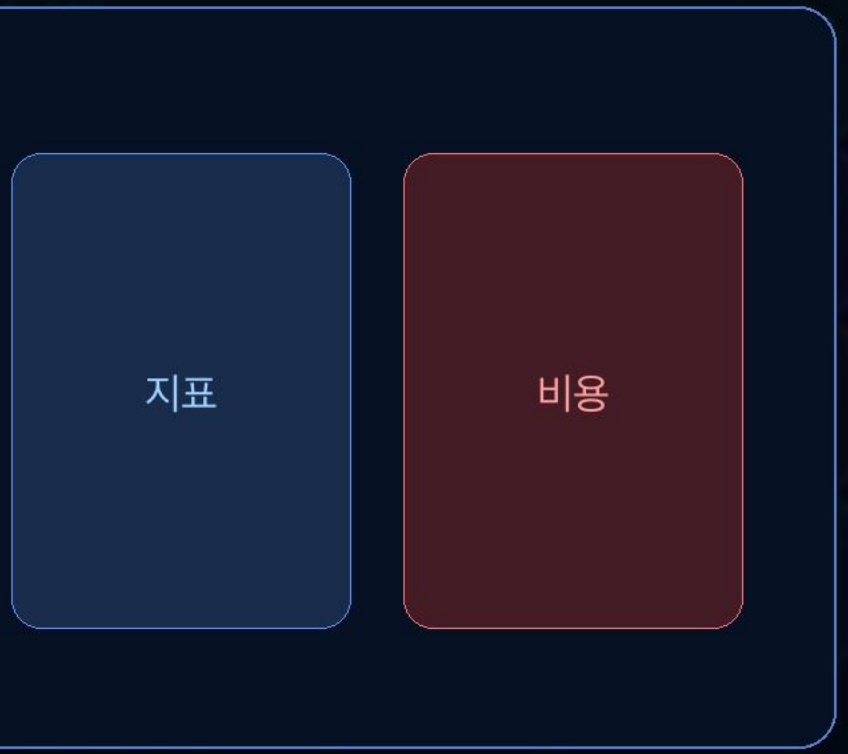


청구서는 또 하나의 로그다

비용은 구조가 무엇을 반복하는지 말해준다.

강의 핵심

1. 청구서는 구조의 낭비를 보여준다
2. 읽기, 대역폭, 함수 시간, API 호출을 본다
3. 비용 문제는 대개 성능 문제와 같은 뿌리다

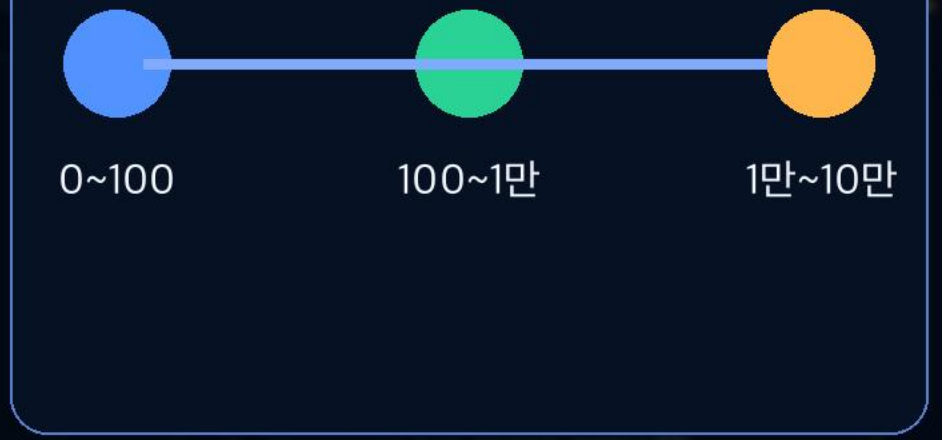


성장하면 질문이 달라진다

0~100명, 100~1만 명, 1만~10만 명은 판단 기준이 다르다.

강의 핵심

1. 0~100명: 먼저 존재하는 서비스
2. 100~1만 명: 첫 병목을 읽는 시기
3. 1만~10만 명: 경계를 다시 묻는 시기



망가지기 전에 더 잘 물어라

증상, 플랫폼, 트래픽, 지표, 최근 변경을 함께 설명하라.

강의 핵심

1. 증상과 플랫폼을 함께 설명한다
2. 트래픽, 지표, 최근 변경을 같이 제공한다
3. AI에게도 좋은 진단 재료가 필요하다