

Long-term Memory MCP

사내 솔루션 기반의 개인화된 장기 기억 메모리 MCP 개발 사례

Sung

Who is Speaker?



Kim Sungyeon
AI Research Engineer

I Braincrew Inc. RAG 팀 Member

Multi-Domain의 RAG Project 수행

- Education
- Shipping
- Construction
- Financial
- Lifelog



LinkedIn



Github

CONTENTS

01 도입

02 시스템 설계

03 핵심 기능 & 적용 사례

04 개발 회고

PART 01

도입

문제 인식: AI의 기억상실

01

AI의 기억상실 문제

현재 대부분의 AI 시스템이 겪는 문제



최근 했던 프로젝트가 무엇인가요?

저는 BrainCrew의 김성연입니다. 최근에 Longterm Memory 프로젝트를 수행했어요.

최근 했던 프로젝트 회의록 액션아이템 알려줘



AI는 이전 대화의 모든 맥락을 잊어버립니다

기억 상실로 인한 3가지 문제

반복적인 설명

매번 같은 배경 정보를 설명해야 함

개인화 불가

사용자의 역할, 선호도,
맥락에 맞는 답변 제공 불가

낮은 생산성

맥락 설정에 시간 낭비

프로젝트 목표

AI Utility 서비스의 핵심 목표 3가지

사용자 프로필 기억

이름, 부서, 기술 스택 등
사용자 정보를 기억하고 활용

문서 시맨틱 검색

보고서, 이메일, 회의록을 저장하고
의미 기반으로 검색

선호도 기반 응답

대화 톤, 문서 형식 등
개인 선호도를 반영한 맞춤형 응답



사용자별 개인화된 AI어시스턴트 구현

| PART 02

시스템 설계

왜 MCP를 선택했는가

02

MCP 선택의 핵심 이유

플랫폼 종속성 탈피

LangChain Memory

Custom API

VS

MCP

메모리 시스템 재개발 필요

수정없이 그대로 사용

MCP의 기술적 장점 4가지

마이크로서비스 친화적

표준 프로토콜의 힘

네이티브 멀티테넌트

실시간 양방향 통신

독립적 배포, 스케일링,
업데이트 가능

MCP 스펙만 따르면
새 도구 추가 즉시 가능

프로토콜 레벨에서
데이터 격리 지원

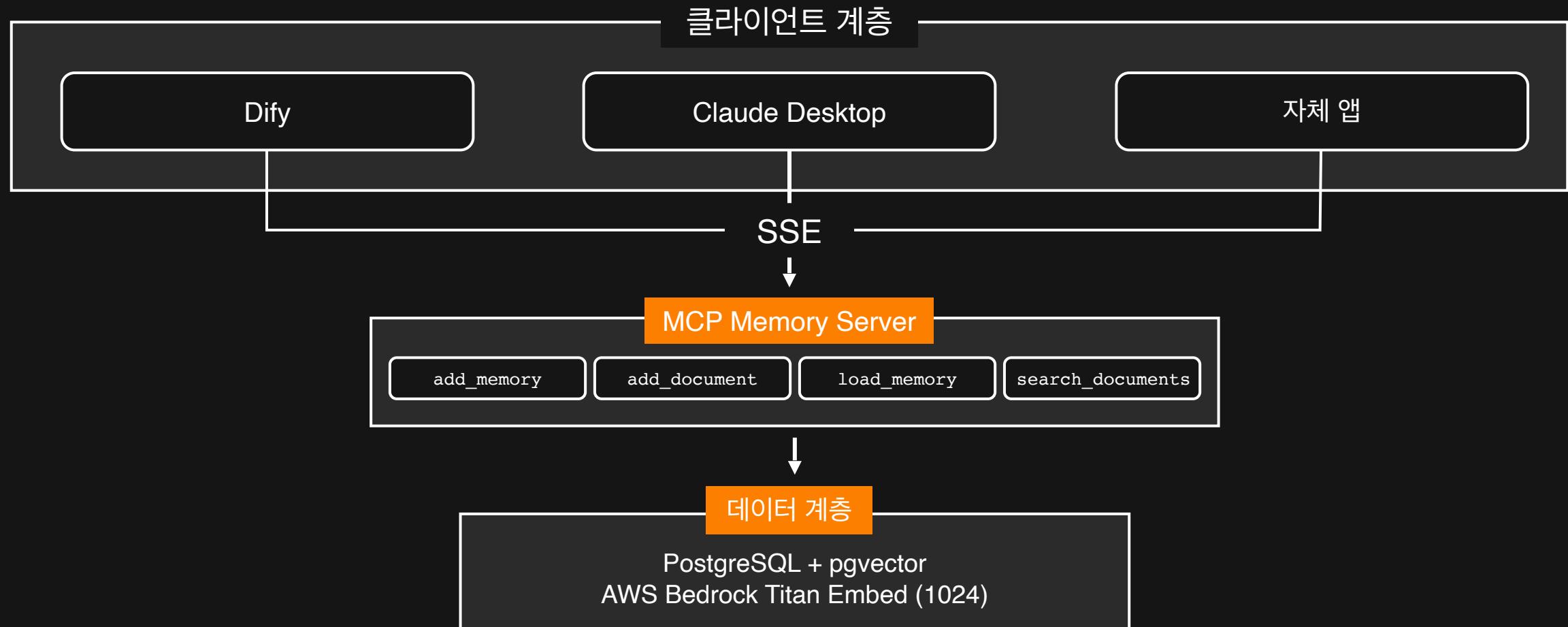
서버가 클라이언트에
능동적 정보 푸시 가능



지금 잘 동작하는 시스템이 아니라
앞으로도 계속 확장할 수 있는 시스템

확장 가능한 아키텍처

3계층 구조



아키텍처의 확장성 이점

플랫폼 교체 비용 0

독립적 스케일링

무중단 업데이트

기능 확장 용이

Dify
→ Claude Desktop
전환 시 Memory
서버 코드 수정 불필요

사용자 1,000명
→ 10,000명: MCP
서버 인스턴스만 추가

새 기능 추가해도
클라이언트에 영향 없음

일정 관리, 알림 등
새 MCP 도구
추가만 하면 끝

데이터베이스 설계

GraphDB vs PostgresDB

GraphDB

VS

PostgresDB

유연함

관계 표현

테이블 기반

높음 (Neo4j)

라이센스 비용

낮음

추가 인프라 필요

운영 복잡도

기존 인프라 활용

MCP 내 LLM 호출 필요

엔티티 추출

불필요

GraphDB는 엔티티 추출을 위한 추가 LLM 호출 비용 발생

고객사에서 이런 숨은 비용을 원하지 않음

간결한 JSON 포맷 설계

Tool Calling 시 저장할 값이 그대로 JSON으로 전달



추가 LLM 호출 없음

간단한 JSON 구조

비용 절감

Tool Calling 한 번으로
기억 저장 완료

명확한 필드명,
단순한 데이터 구조

엔티티 추출
비용 0원

데이터베이스 구조: 5개 테이블

테이블	용도	특징
personal_info	사용자 프로필 (이름, 부서, 직책 등)	FK 기준점
project_info	프로젝트명, 상태, 기간, 팀원	다건 저장
work_pattern	업무 패턴 유형, 빈도, 시간대	다건 저장
preferences	사용자 선호 정보 (대화 톤, 문서 형식 선호도)	Upsert
documents	기억해야 할 업무 문서 (보고서, 메일, 회의록)	HNSW 인덱스

확장 용이: 새 기억 유형 필요 시 테이블 하나만 추가

데이터베이스 구조 : 확장성 측면의 핵심 이점

제네릭 Repository 패턴 기반의 시스템 구조

새 테이블 추가

スキマ 정의

+

Repository 클래스
2개만 추가

(BaseRepository 모든
CRUD 즉시 사용 가능)

새 컬럼 추가

Pydantic 필드 1줄

+

DB ALTER

(동적 upsert로
기존 코드 수정 불필요)

사용자 건강 정보
테이블 추가

スキ마 1개

+

Repository
1개

임베딩 기반 검색이
필요한 새 테이블

DocumentRepository 패턴 복사

personal_info에
SNS 계정 필드 추가

スキ마
필드 1줄

DB ALTER

새. 문서 카테고리 추가
(예: "contract")

Enum에 1줄 추가

| PART 03

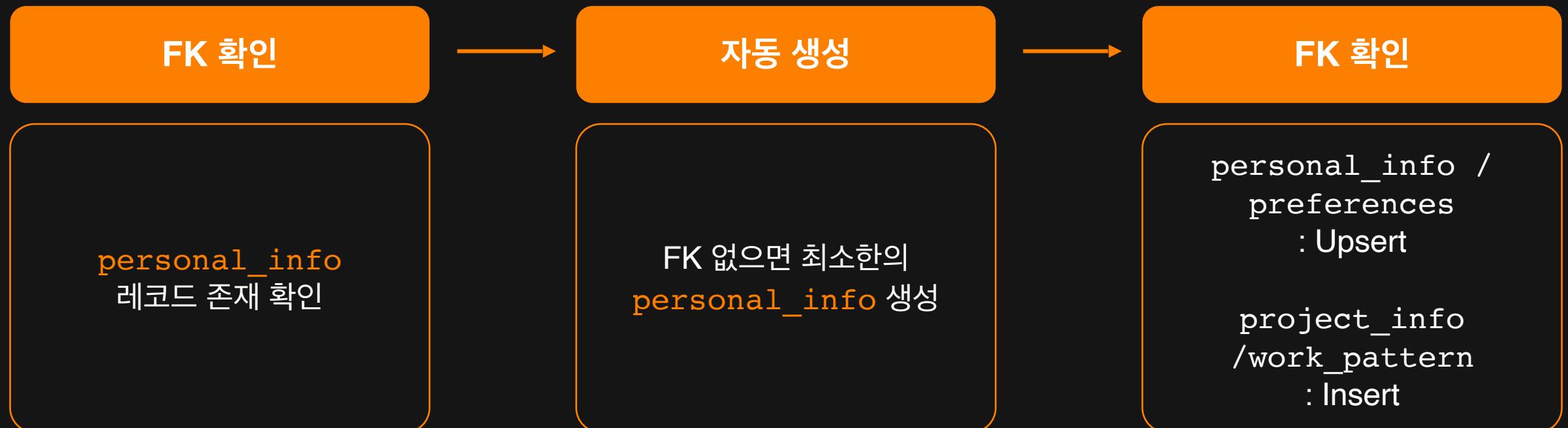
핵심 기능 & 적용 사례

MCP 도구와 실제 시나리오

03

add_memory: 기억 저장

처리 흐름



- `user_id` : 사용자 식별자
- `type` : `personal_info / project_info / work_pattern / preferences`

load_memory: 기억 조회

type = "report" / "email" / "meeting_notes"

```
# 4개 테이블 병렬 조회 (asyncio.gather)
results = await asyncio.gather(
    get_personal_info(user_id),
    get_project_info(user_id),
    get_work_pattern(user_id),
    get_preferences(user_id)
)
# 0.5초 이내 전체 프로필 조회
```

해당 유형의 선호도만 필터링해서 반환

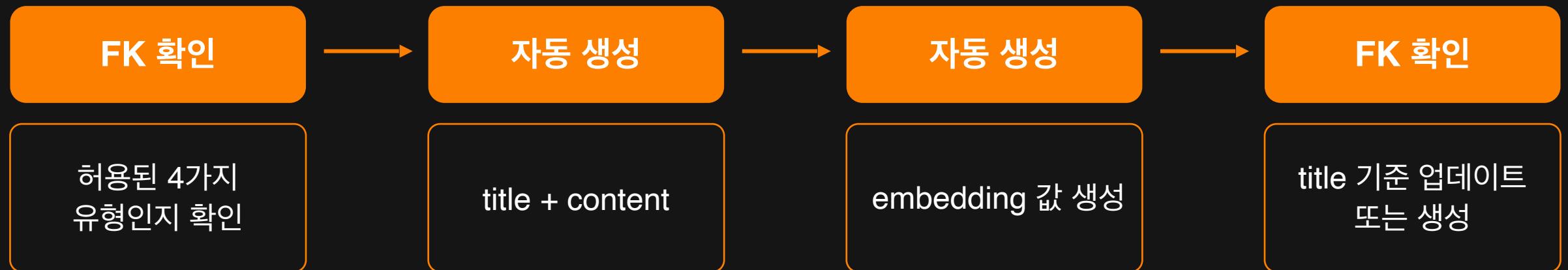
적용 사례 1: 사용자 프로필 기억

시나리오 1: 사용자 프로필 학습



add_document: 문서 벡터화 저장

처리 흐름



- user_id, title(선택), content, category
- Category: report / email / meeting_notes / general

문서 저장 2초 이내 완료

search_documents: 시맨틱 검색

query가 있는 경우: 시맨틱 검색

```
SELECT * FROM documents
WHERE user_id = $1
AND 1 - (embedding <=> $2) > 0.3 -- 유사도 임계값
ORDER BY embedding <=> $2
LIMIT $3;
```

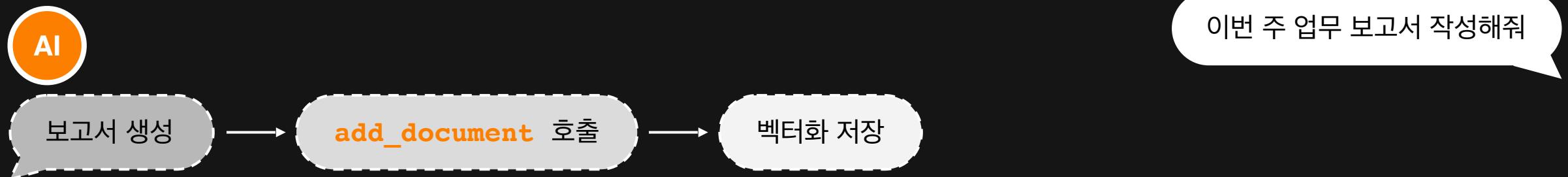
query가 없는 경우: 최신 문서 조회

created_at 기준 내림차순 정렬, top_k개 반환

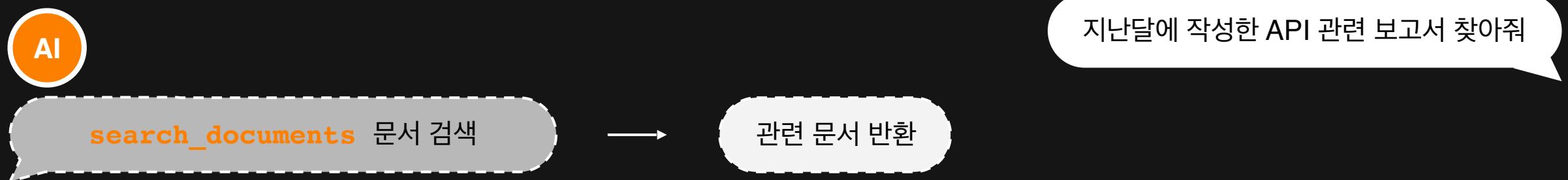
HNSW 인덱스로 검색 1초 이내 완료

적용 사례 2: 업무 문서 기억

저장 단계



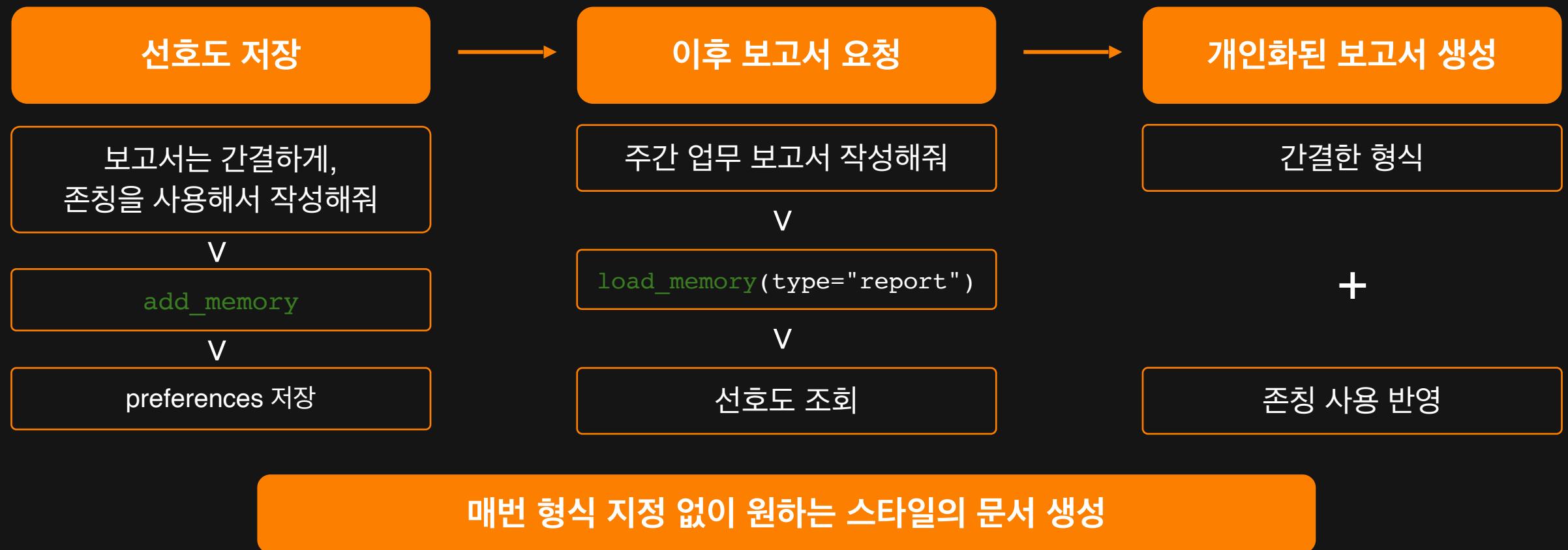
검색 단계



키워드 일치가 아닌 의미 기반 검색 - "API 개발 현황" 제목도 검색됨

적용 사례 3: 선호도 기반 응답

처리 흐름



| PART 04

개발 회고

개발 회고 & Lessons Learned

04

좋았던 점 & 기술적 Lessons Learned

경험

AI 장기 기억 연구

학문적 개념을
실제 서비스에 적용

고객사 대응 방법

요구사항 수집
및 커뮤니케이션 학습

MCP & AWS 배포

최신 프로토콜 구현
및 클라우드 배포

기술

벡터DB 구축

PostgreSQL + pgvector로
관리 부담 감소

기억 유형 분리

4가지 분리로
로직 단순화

확장성을 고려한 개발

실무 및 운영관점에서
확장성 있는 구조로 개발

핵심 교훈

단순하게 시작하라

MVP를 먼저 만들고,
피드백을 받아
확장하는 게 효율적

비용과 인프라를
함께 고려하라

기능 구현뿐만 아니라
비용, 인프라,
운영 복잡도를 함께 고려

임베딩 모델, DB 구성,
서버 스케일링 등
모든 결정에 비용이 따름

고객별 요구사항
우선순위를 파악하라

고객마다 중요하게
생각하는 기능이 다름

어떤 고객은 **검색 정확도**,
어떤 고객은 **응답 속도**를
더 중시

인프라 학습을 병행하라

좋은 코드도
배포가 안 되면
의미가 없음

Q&A

Q

임베딩 모델로 Titan을 선택한 이유는?

A

고객사 AWS 인프라와 통합 용이,
1024차원으로 충분한 표현력 + 합리적 비용

기억 데이터의 보존 기간은?

현재 프로필 정보는 영구 보존. 문서는 6개월 단위로 보
존 하며 사용자별 보존 정책 설정 기능 추가 예정

다른 플랫폼에도 적용 가능한가?

MCP 표준을 따르므로
MCP 지원 플랫폼 어디든 연결 가능

Thank You

