inercorp
teknologi yang anda butuhkan

WORKSHOP

MEMBUAT WEB CHATBOT

Berbasis Retrieval Augmented Generation dengan Nextjs, n8n dan Posgre PgVector

Ilham Maulana, M.Kom

Sabtu, 27 Desember 2025

Online

Ilham Maulana, M.Kom

Pekerjaan :

- CTO - PT INER CORP INDONESIA
- Software Engineer - PT PRIMATEK INFOKOM JAYA

Pendidikan

- Magister Ilmu Komputer - Universitas Nusa Mandiri

Aktivitas

- Narsum Workshop Agentic AI di UNM Margonda 2025
- Narsum Workshop & Ujikom Basis Data BSI Bogor 2025
- Dosen Praktisi Universitas Negeri Semarang 2024

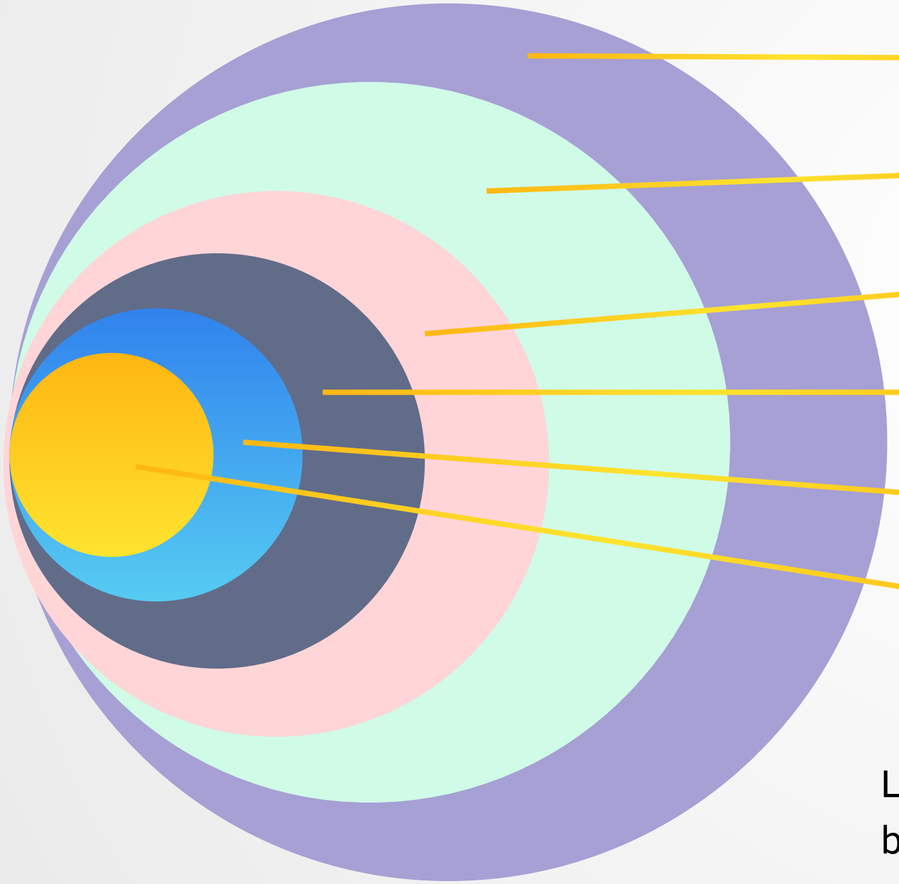
www.ilhammaulana.id

Hasil yang Akan Kamu Bawa Pulang

- Web chatbot tanya PDF
- Source code siap pakai
- Pemahaman alur RAG
- Pengalaman praktik end-to-end (frontend + backend)
- Sertifikat keikutsertaan

https://tanyapdf.k4il.cloud

Large Language Model



- Artificial Intelligence (AI)
- Machine Learning (ML)
- Deep Learning (DL)
- Natural Language Processing (NLP)
- Language Model
- Large Language Model (LLM)

Large Language Model merupakan bagian dari Natural Language Processing yang berada dalam ranah Artificial Intelligence, dan dikembangkan menggunakan pendekatan deep learning berbasis arsitektur Transformer.

LLM Timeline

1950–1990

Rule-Based AI & NLP
AI berbasis aturan (if–then), kamus & grammar manual.

1990–2010

Statistical NLP & Machine Learning
N-gram, Naive Bayes, bahasa dipelajari dari probabilitas.

2013–2017

Word Embedding
Representasi kata sebagai vektor (makna & semantik).

2015–2018

Deep Learning NLP
RNN, LSTM, GRU memahami urutan kata & konteks.

2017

Transformer Era
Attention mechanism, konteks panjang & training paralel.

2018–2020

Pretrained Language Model
Pretrain di data besar, fine-tuning ke task spesifik.

2020–2022

Large Language Model (LLM)
Model skala besar, zero-shot & few-shot learning.

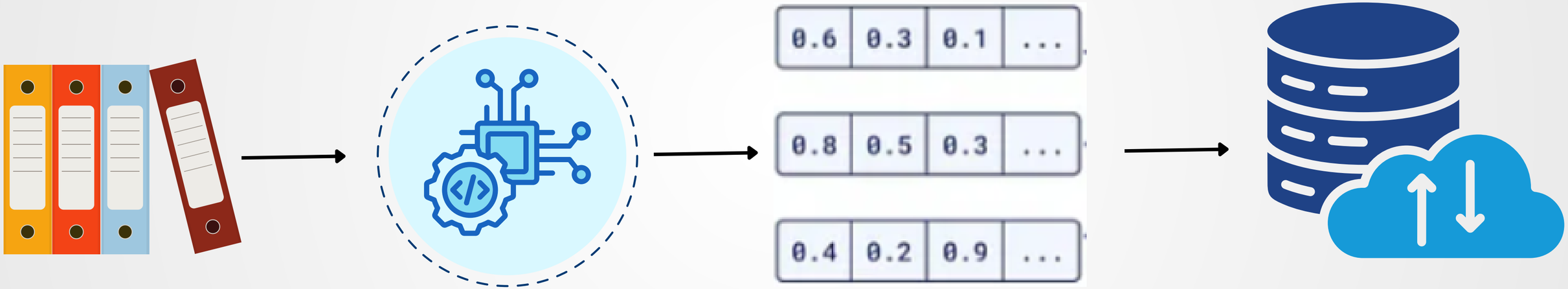
2022–sekarang

Chat & Instruction-tuned LLM
Interaksi natural, reasoning, coding, asisten AI.

2023–sekarang

LLM + RAG & AI Agent
Terhubung ke dokumen, database, API.

Alur Dokumen ke Vector Database



Document

Embedding Model

Chunk + Embedding

Vector Database

Vector Database

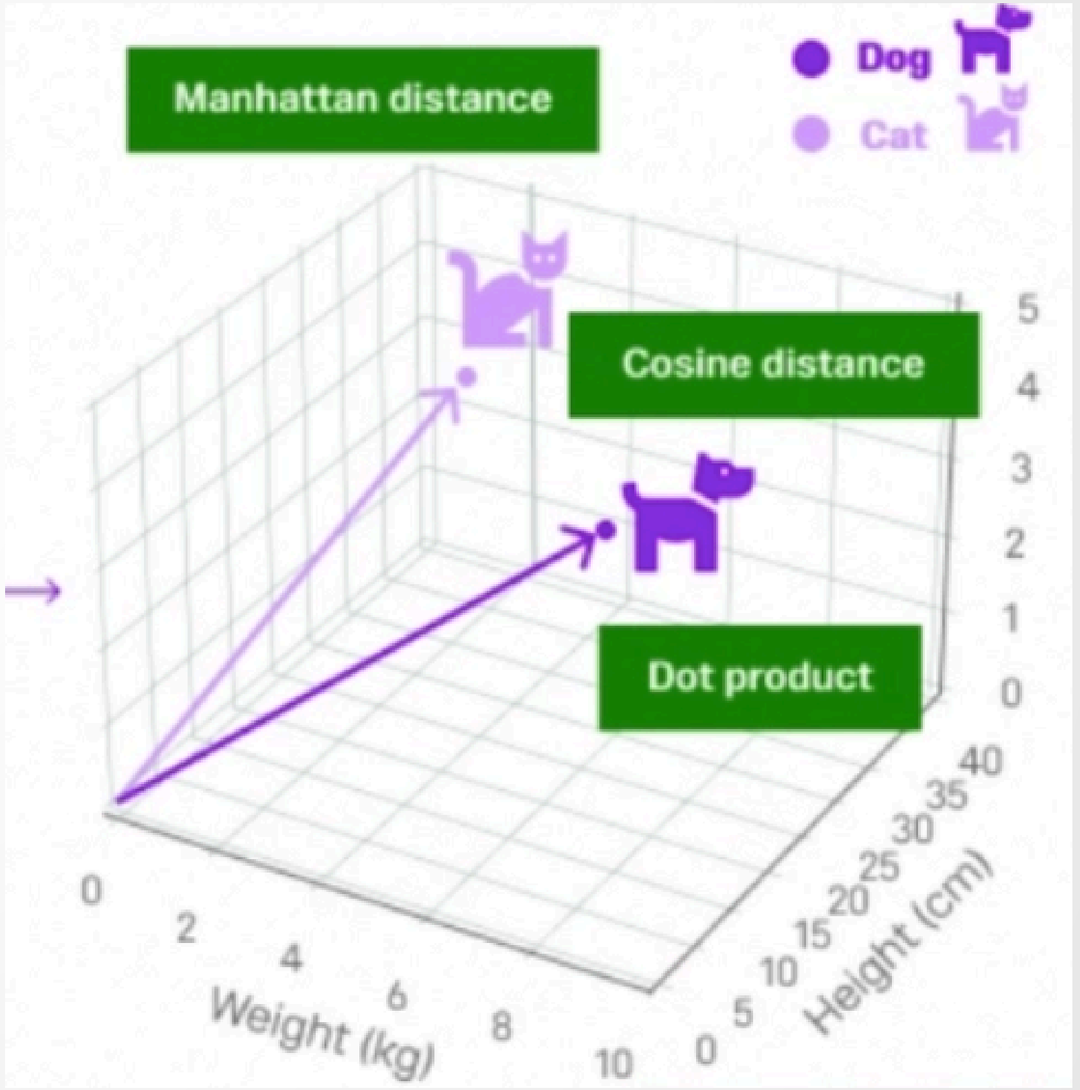
Vector Database adalah database khusus yang menyimpan data dalam bentuk vektor (embedding) dan digunakan untuk mencari kemiripan makna (semantic similarity), bukan sekadar mencocokkan kata.

Pgvector

Qdrant

Chroma

Pinecone



Chunking Dokumen

Chunking dokumen adalah proses memecah dokumen panjang menjadi potongan-potongan kecil (chunk) sebelum di-embedding dan disimpan ke Vector Database.

Kenapa perlu chunking?

- LLM & embedding model punya limit token. Dokumen panjang tidak bisa diproses sekaligus.
- Mengambil bagian yang relevan saja, bukan seluruh dokumen.
- Pencarian lebih cepat, biaya embedding lebih rendah.
- Jawaban lebih fokus

Apa itu embedding

Embedding adalah proses mengubah data (terutama teks) menjadi representasi angka (vektor) agar bisa dipahami dan diproses oleh model AI.

Contoh sederhana Kalimat:

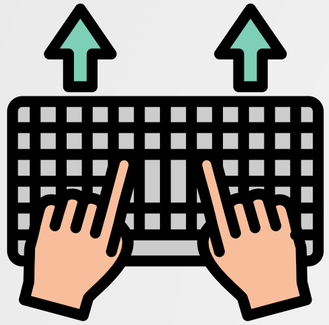
“Saya belajar AI”

“Saya sedang mempelajari kecerdasan buatan”

Walaupun kata-katanya berbeda, embedding kedua kalimat ini akan mirip, karena maknanya sama. Hasil embedding

[0.12, 0.87, 0.33, ...]
[0.11, 0.85, 0.35, ...]

Retrieval Augmented Generation



User Promt

Embedding Model

Cosine Similarity

Top-K Selection

Vector DB

Retrieval

LLM

Augmented Prompt

Generate Response

Retrieval-Augmented Generation adalah pendekatan di mana LLM menghasilkan respons berdasarkan konteks eksternal yang diambil melalui mekanisme retrieval.

Retrieval

Proses mencari dan mengambil informasi yang paling relevan dari sumber data eksternal (Vector Database) berdasarkan pertanyaan pengguna.

- User prompt diubah menjadi embedding (vektor)
- Sistem melakukan similarity search (misalnya cosine similarity)
- Dipilih Top-K dokumen yang paling mirip secara makna

Tujuan:
Menyediakan konteks yang akurat dan relevan untuk menjawab pertanyaan

Augmented

Proses menggabungkan pertanyaan pengguna dengan konteks hasil retrieval menjadi satu prompt yang lebih kaya informasi.

- Prompt asli user
- Potongan dokumen hasil retrieval
- Disusun menjadi Augmented Prompt

Tujuan:
“Memperkaya” prompt agar LLM tidak menebak, tapi menjawab berdasarkan data

Generation

Proses LLM menghasilkan jawaban akhir berdasarkan augmented prompt yang telah berisi konteks relevan.

- LLM membaca augmented prompt
- Memahami konteks + pertanyaan
- Menghasilkan respons natural dalam bahasa manusia

Tujuan:
Memberikan jawaban yang informatif, kontekstual, dan lebih akurat

Kenapa teks diubah jadi vektor

- AI bekerja dengan angka, bukan teks
- Vektor menyimpan MAKNA, bukan sekadar kata
- Vektor memungkinkan perhitungan kemiripan
- Vector Database butuh vektor
- LLM juga memproses vektor

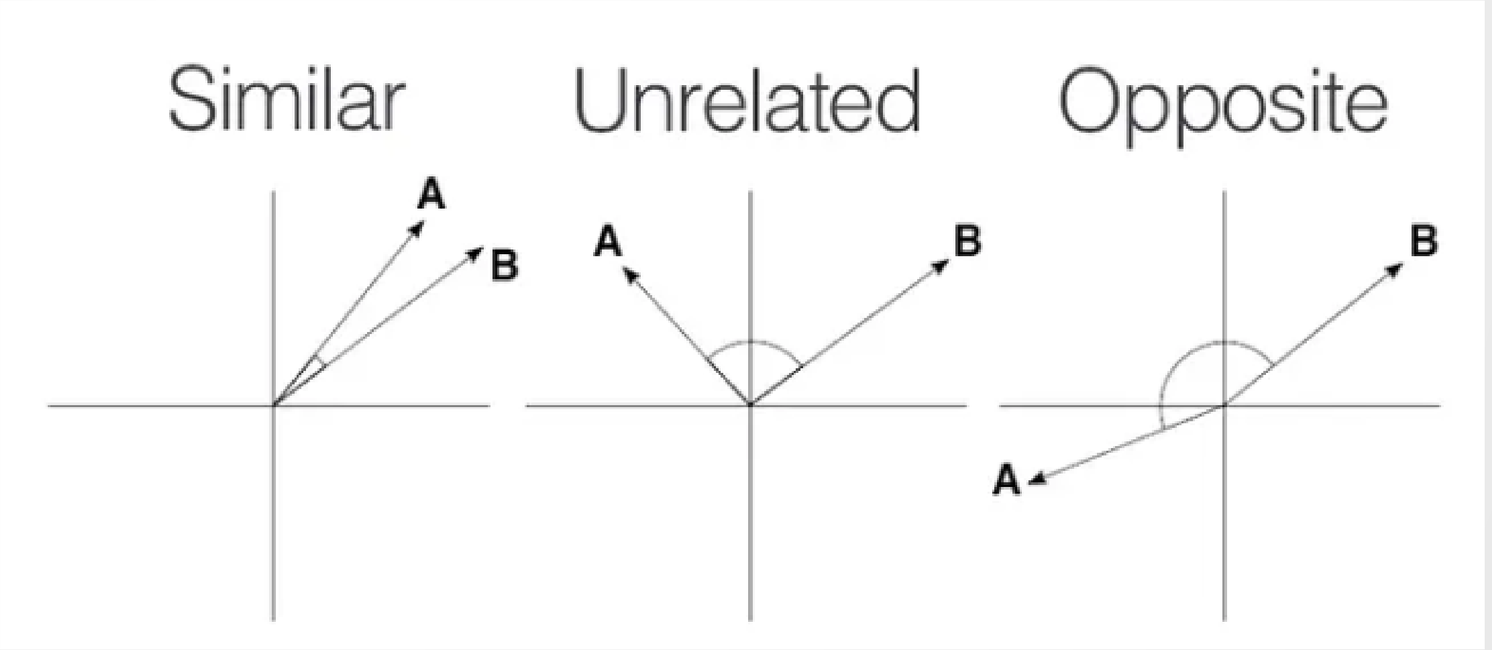
Teks diubah menjadi vektor supaya AI bisa memahami makna, menghitung kemiripan, dan melakukan pencarian serta penalaran berbasis konteks.

Cosine similarity

Cosine similarity adalah metode untuk mengukur tingkat kemiripan dua vektor dengan melihat sudut di antara vektor tersebut, bukan panjangnya.

Cosine similarity mengukur seberapa searah dua vektor makna.

- Sudut kecil = makna mirip
- Sudut besar = makna berbeda



HANDS-ON

Hands-On

Tanya PDF | Upload PDF | Handbook - Build Your First AI Agent.pdf

Document List 164

1. 4 Proses Pembayaran Tools yang dipakai: getReservations makePayment updateReservations A. Cek Status Reservasi Gunakan getReservations: NEW → boleh membuat invoice PENDING → invoice sudah dibuat PAID → pembayaran selesai B. Jika tamu ingin bayar (status = NEW) Panggil: makePayment { "order_id" : nomor_reservasi, "total" : total_biaya } C. Setelah invoice berhasil dibuat Ambil dari hasil makePayment: 64 nomor VA transaction_id (nomor invoice) expired Lalu panggil: updateReservations { status: "PENDING" , va: nomor_va, nomor_invoice: transaction_id, expired: tanggal_expired } 8.15. To Be Continued... Worflow lengkap dapat didownload di https://github.com/k4ilham/Build-Your-First-AI-Agent/tree/main/workflow Kerja bersama terasa ringan, Semua langkah jadi tertata. Workflow selesai, hasil pun memuaskan, Semoga bermanfaat untuk kita semua. 65 66

2. 5. Basic Auth: pilih credential Midtrans yang sudah dibuat sebelumnya Aktifkan Send Query Parameters, kemudian isi: 1. order_id → nilai dari AI (Defined automatically by the model) 2. total → nilai dari AI (Defined automatically by the model) Pada bagian Body Content Type, pilih JSON. Untuk Specify Body, pilih Using JSON, lalu isi dengan payload berikut: { "payment_type": "bank_transfer" , "transaction_details" : { "order_id": "{{ $fromAI('order_id') }}" , "gross_amount": {{ $fromAI('total') }} }, "bank_transfer" : { "bank": "bca" } } Payload ini akan mengirim permintaan pembayaran ke Midtrans menggunakan metode bank transfer BCA, dengan order_id dan total yang berasal dari hasil percakapan AI. Sekarang tambahkan prompt pada bagian System Message di AI Agent. 63

digunakan nanti di dalam workflow. Kembali lagi

Handbook - Build Your First AI Agent 2 / 66 93%

1


2


3

Daftar Isi 2

Sisa chat hari ini: 99 pesan

Halo! Saya asisten PDF Anda. Silakan tanyakan apa saja tentang dokumen ini.

judulnya apa

Judulnya adalah "Handbook - Build Your First AI Agent". Apakah Anda ingin informasi lebih lanjut dari dokumen ini?

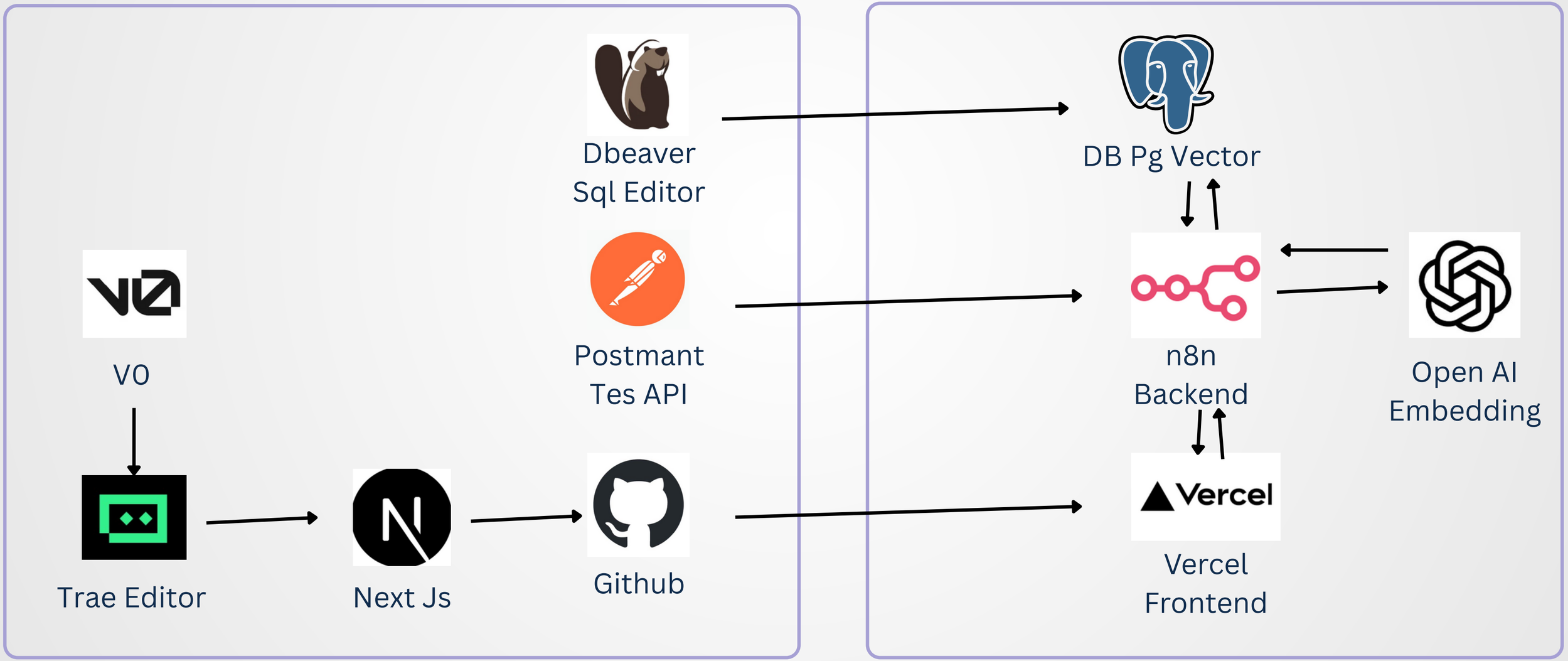
Ketik pesan Anda...

Chatting about: Handbook - Build Your First AI Agent.pdf

Arsitektur

Development

Deployment



POSTGRE

https://github.com/k4ilham/belajar-membuat-chatbot-berbasis-rag/tree/main/postgre

Postgre

Host : 43.157.223.46

Port : 55432

Database : **vectordb**

username : **user_alpha**

password : alpha_!92X

username : **user_beta**

password : beta_#73Q

username : **user_gamma**

password : gamma_$55Z

Tabel Document

```
create table documents (
    id bigserial primary key,
    content text,
    metadata jsonb,
    embedding vector(1536)
);
```

Function Match Document

```
create function match_documents (
  query_embedding vector(1536),
  match_count int default null,
  filter jsonb DEFAULT '{}'
) returns table (
  id bigint,
  content text,
  metadata jsonb,
  similarity float
)
language plpgsql
as $$
#variable_conflict use_column
begin
  return query
  select
    id,
    content,
    metadata,
    1 - (documents.embedding <=> query_embedding) as similarity
  from documents
  where metadata @> filter
  order by documents.embedding <=> query_embedding
  limit match_count;
end;
$$;
```

BACKEND N8N

User

url : https://n8n-ra3tyopx8qvs.n8x.web.id

username : futura21@gmail.com
username : futura22@gmail.com
username : futura23@gmail.com
username : futura24@gmail.com
username : futura25@gmail.com
username : futura26@gmail.com
username : futura27@gmail.com
username : futura28@gmail.com
username : futura29@gmail.com
username : futura30@gmail.com

username : futura31@gmail.com
username : futura32@gmail.com
username : futura33@gmail.com
username : futura34@gmail.com
username : futura35@gmail.com
username : futura36@gmail.com
username : futura37@gmail.com
username : futura38@gmail.com
username : futura39@gmail.com
username : futura40@gmail.com

password : Futura05unm

url : https://n8n-jd8rxks4f82u.uranium.sumopod.my.id

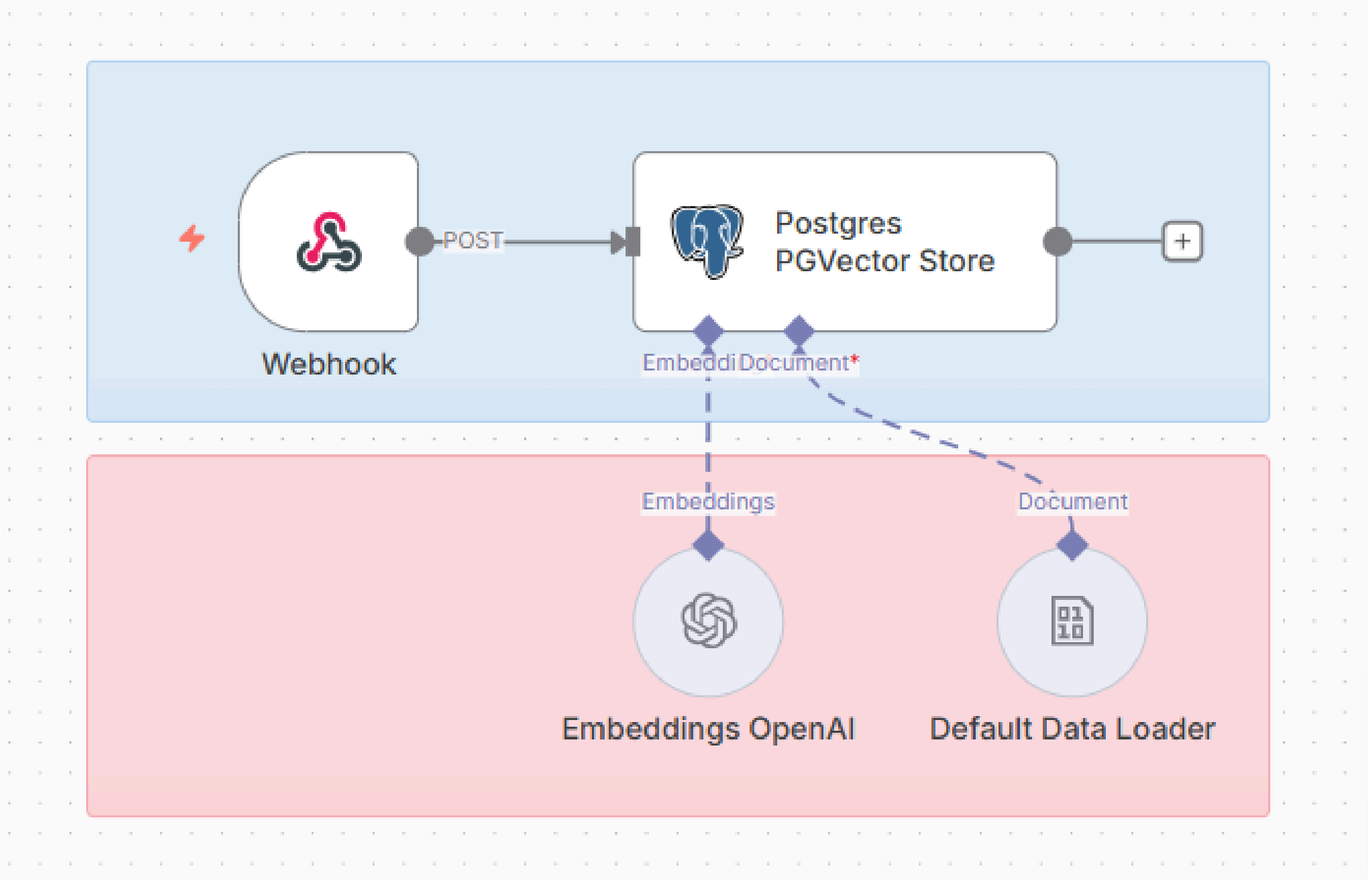
username : futura01@gmail.com
username : futura02@gmail.com
username : futura03@gmail.com
username : futura04@gmail.com
username : futura05@gmail.com
username : futura06@gmail.com
username : futura07@gmail.com
username : futura08@gmail.com
username : futura09@gmail.com
username : futura10@gmail.com

username : futura11@gmail.com
username : futura12@gmail.com
username : futura13@gmail.com
username : futura14@gmail.com
username : futura15@gmail.com
username : futura16@gmail.com
username : futura17@gmail.com
username : futura18@gmail.com
username : futura19@gmail.com
username : futura20@gmail.com

password : Futura05unm

Workflow

Upload Document

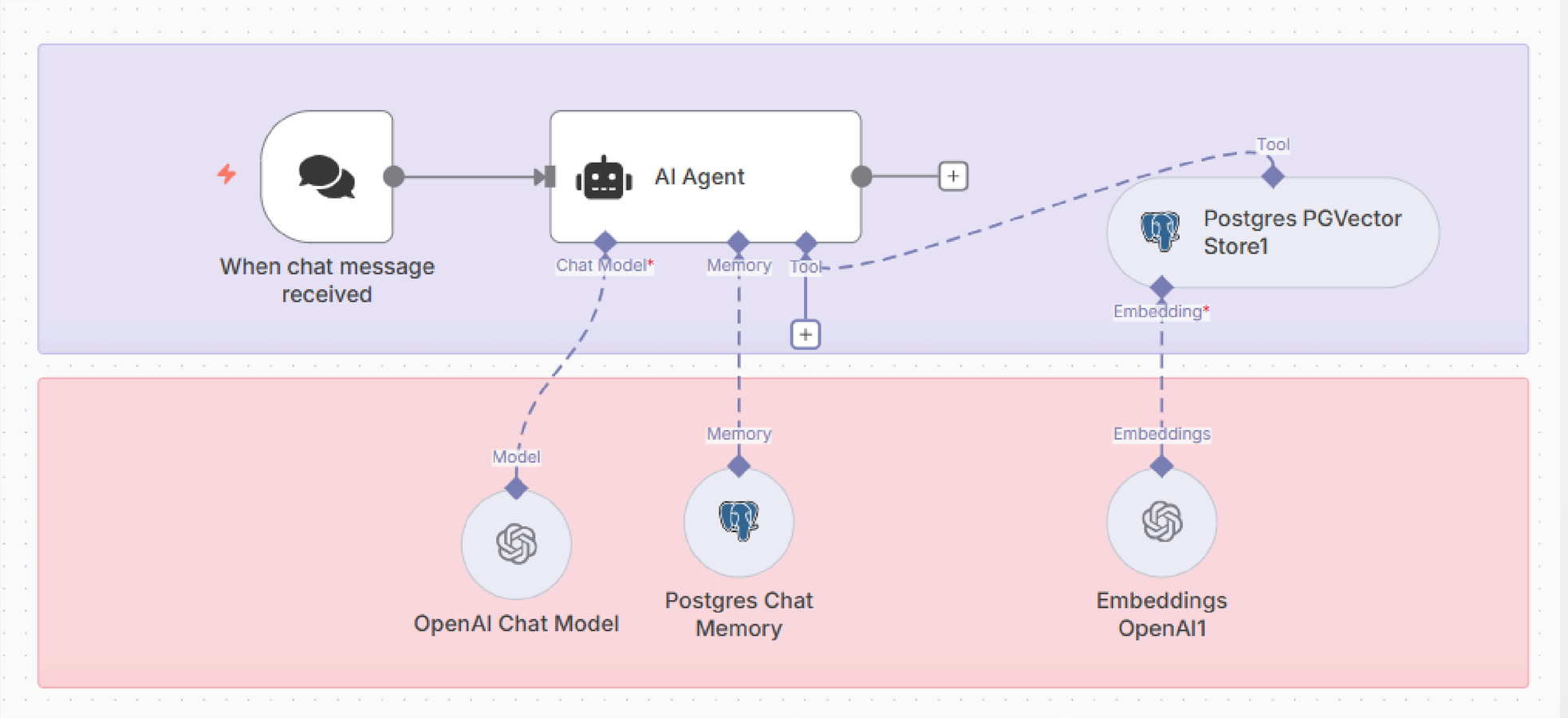


SQL

```
SELECT * FROM documents;

SELECT * FROM documents where id=4;

SELECT embedding FROM documents where id=4 LIMIT 1;

SELECT id, content, 1 - (documents.embedding <=> q.embedding) AS similarity FROM documents
CROSS JOIN ( SELECT embedding FROM documents WHERE id = 1 ) q ORDER BY
documents.embedding <=> q.embedding LIMIT 10;

SELECT id, content, similarity FROM match_documents( (SELECT embedding FROM documents where
id=4 LIMIT 1), 10, '{}'::jsonb );

SELECT id, content, similarity FROM match_documents( ARRAY[ -0.00527411, 0.010059996,
0.0252335, ... ]::vector, 10, '{}'::jsonb );
```

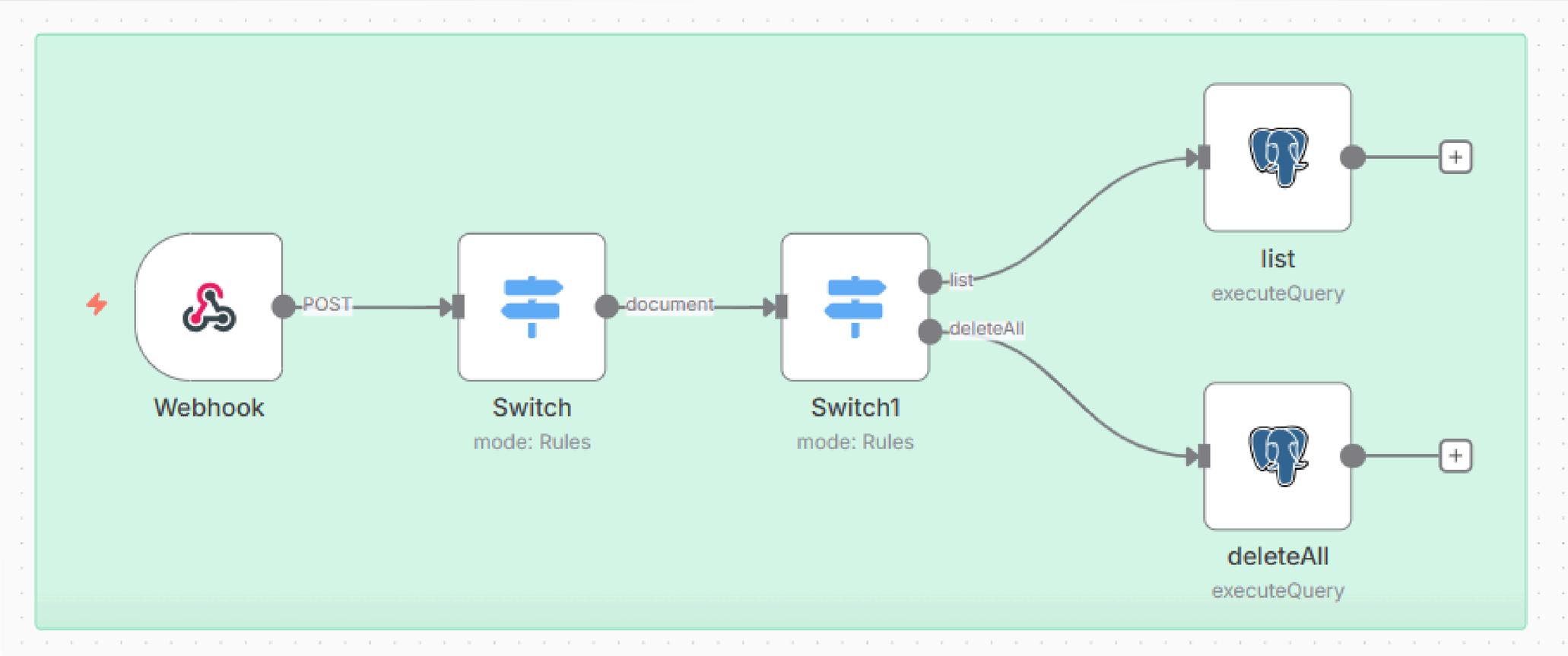
Workflow

Chatbot AI Agent + RAG



Workflow

Rest API



FRONTEND NEXT JS

https://github.com/k4ilham/belajar-membuat-chatbot-berbasis-rag/tree/main/n8n

Pembuatan FrontEnd

- Buat desain di V0
- Koneksikan dengan github
- Publish Project ke Vercel
- Koneksikan Dengan Domain
- Download Code di Local dengan github Desktop
- Buka Code dengan TRAE
- Ubah kode
- Push Ke github
- Otomatis Update Ke Vercel

inercorp
teknologi yang anda butuhkan

TERIMA KASIH