

# Laporan Progres 5

## Integration

Oleh: Mahendra Zidane Rainafa (24523142)

### A. Pendahuluan

Progress terakhir ini merupakan puncak implementasi arsitektur *Retrieval Augmented Generation* (RAG) dengan melakukan integrasi total antara dua saluran komunikasi (Telegram dan WebApp) ke dalam satu sistem pemrosesan pusat. Berbeda dengan pendekatan chatbot biasa, sistem ini menggunakan arsitektur *Chain* (*Question and Answer Chain*) yang secara ketat mewajibkan setiap pertanyaan pengguna diproses melalui pencarian data vektor di Pinecone terlebih dahulu. Hal ini menjamin bahwa setiap jawaban yang dihasilkan oleh AI memiliki landasan referensi yang kuat dari dokumen internal, meminimalisir halusinasi, dan didistribusikan kembali ke platform asal pengguna (Telegram atau WebApp) melalui logika perutean (*routing*) yang cerdas.

### B. Arsitektur Final

Berdasarkan implementasi final, sistem menggunakan alur "Merge & Chain" sebagai berikut:

#### 1. Input Unification (Penyatuan Jalur):

- Trigger Telegram dan Webhook diproses terpisah menggunakan node **Edit Fields** untuk standarisasi format data (memberi label *source*).
- Kedua jalur digabungkan menggunakan node **Merge** (Mode: Append) menjadi satu antrian proses.

#### 2. RAG Processing Core:

- Menggunakan node *Question and Answer Chain* sebagai otak utama.
- Retrieval: Terhubung ke *Vector Store Retriever* → *Pinecone* → *Embeddings Google Gemini* untuk mencari konteks dokumen.
- Generation: Terhubung ke *OpenAI Chat Model* (atau Groq) untuk merangkai jawaban berdasarkan konteks yang ditemukan.

#### 3. Smart Routing (Jalur Pulang):

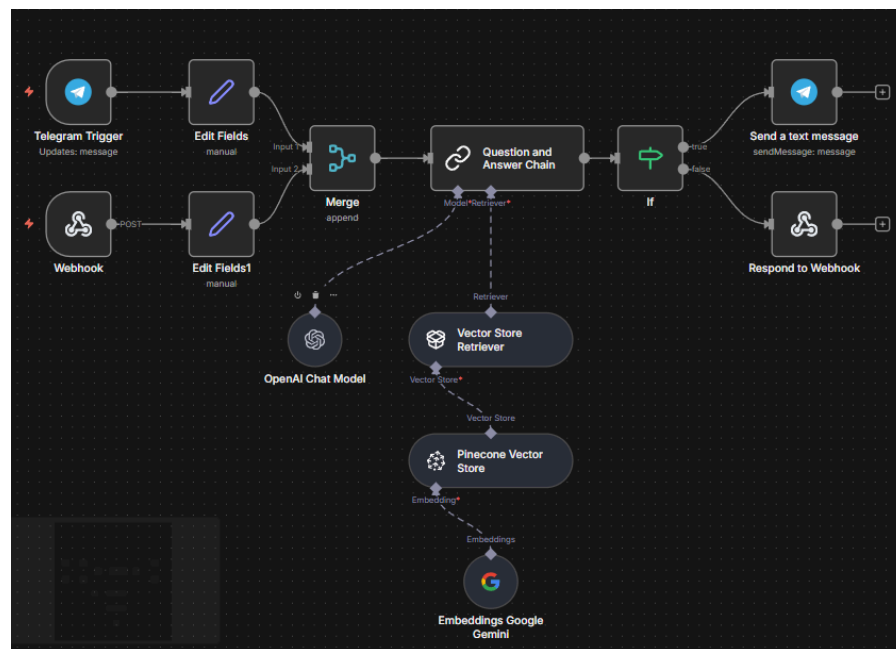
- Menggunakan node **If** untuk memeriksa label *source*.
- Jika *True* → Kirim ke Telegram.
- Jika *False* → Kirim respons ke Webhook (WebApp).

## C. Langkah Pengerjaan

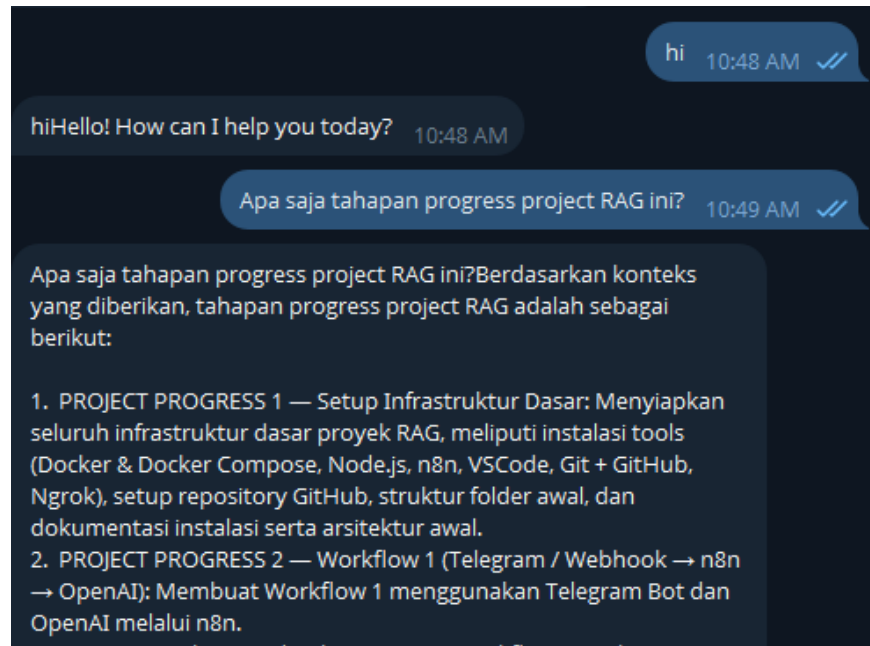
1. **Standarisasi Input:** Menambahkan node **Edit Fields** setelah masing-masing trigger untuk menandai asal pesan (misal: `source: telegram` atau `source: webhook`).
2. **Konfigurasi Chain:** Mengganti logika chatbot standar dengan node **Question and Answer Chain** untuk memaksa sistem membaca database Pinecone sebelum menjawab.
3. **Integrasi Vektor:** Menghubungkan chain dengan **Pinecone Vector Store** yang menggunakan model embedding **Google Gemini** agar sesuai dengan data yang di-upload pada Progress 4.
4. **Logika Percabangan:** Memasang node **If** di akhir proses untuk mengarahkan jawaban AI kembali ke pengirim yang tepat agar tidak salah alamat.

## D. Hasil Implementasi

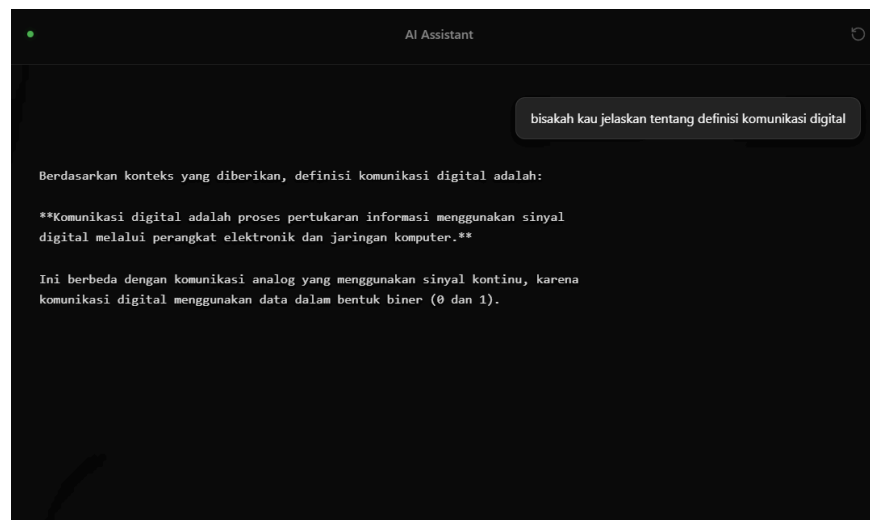
1. Screenshot Workflow Gabungan:



2. Screenshot Telegram Berbasis Data:



3. Screenshot WebApp Berbasis Data:



4. Link workflow:

<https://github.com/kevv01/Project-RAG/blob/main/workflows/workflow-4-integration.json>

5. Link GitHub:

<https://github.com/kevv01/Project-RAG.git>

