



LLM-Based Tools and Gemini API Integration for Data Scientists

+ Partnerships Hacktiv8

This Program is part of the AI Opportunity Fund: Asia Pacific, in collaboration with AVPN and supported by [Google.org](#) and the Asian Development Bank.



Retrieval Augmented Generation with LangChain

Session 4

Learning Session

Introduction to AI & Generative AI with Gemini

- Pengenalan AI, Generative AI dan LLM
- AI Ethics
- Pengenalan Chatbot
- Hallucinations & Prompting Techniques
- Pengenalan Gemini
- Hands-On

Session 2

Implementing Generative AI with Gemini

- Teknik Prompting Lanjutan
- Konfigurasi Google Gemini
- Pemanggilan Fungsi

Session 3

RAG with LangChain

- Pengenalan RAG
- Vector Database
- Pengenalan LangChain
- Pengenalan Llama
- Hands-On

Session 5

Building Applications with LLM

- Pengenalan Streamlit
- Pembuatan ChatBot LLM dengan Streamlit
- Pengenalan AI Agents
- Mendeploy Aplikasi Streamlit ke Cloud

Session 4

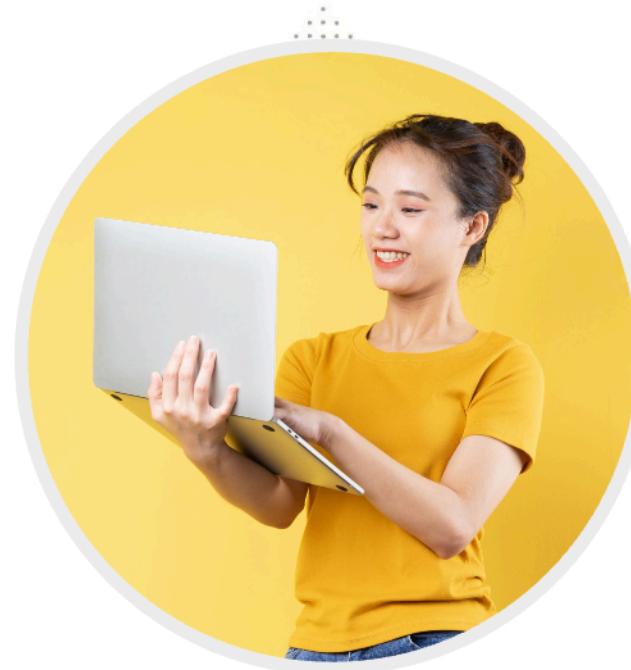
Training Guidelines

Untuk mengikuti materi pelatihan berikut ada beberapa hal yang harus diperhatikan oleh peserta:

- Peserta diharapkan sudah memiliki pengetahuan dasar tentang **Python dan Generative AI**.
- Peserta sudah membuat akun pada <https://console.groq.com/>.
- Peserta sudah menyiapkan 1 dokumen PDF (2-5 halaman) yang akan digunakan untuk membuat chatbot.
- Instruktur diharapkan dapat menjelaskan materi dengan jelas dan memastikan peserta pelatihan dapat memahami topik yang dibahas.
- Materi yang diajarkan sebagian besar adalah praktik atau menggunakan Google Colab (Jupyter Notebook).

List of Contents

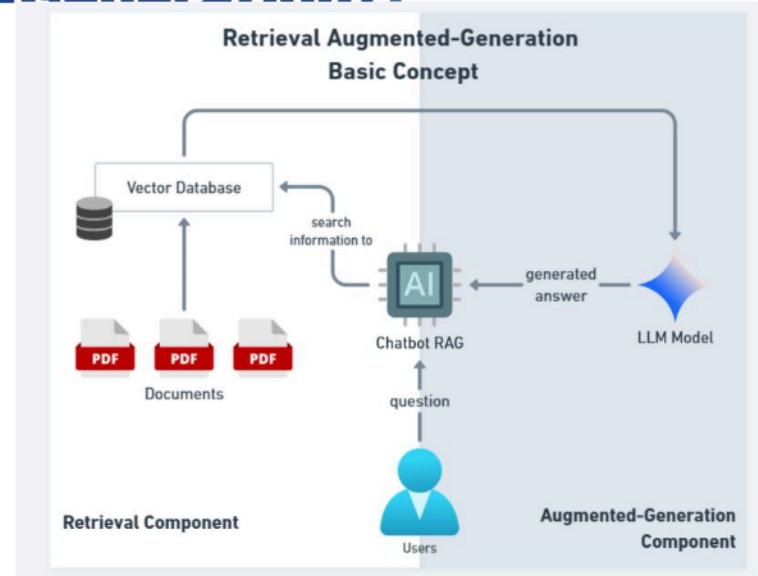
- Pengenalan RAG
- Vector Database
- Pengenalan LangChain
- Pengenalan Llama
- Exercises





Pengenalan RAG

Apa Itu RAG (Retrieval Augmented-Generation)?

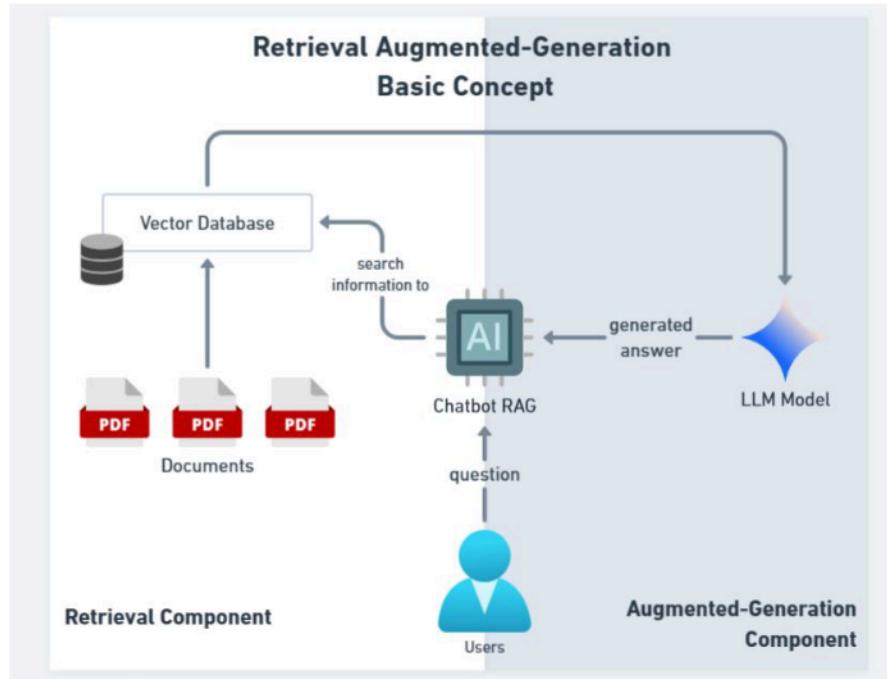


RAG adalah sebuah teknik yang menggabungkan retrieval (pencarian informasi eksternal) dan generation (pembuatan teks oleh LLM). Teknik ini memungkinkan pengimplementasian LLM hanya untuk tanya jawab pada topik tertentu sesuai dengan dokumen yang diberikan.

Komponen pada RAG

Retrieval Component

Berfokus pada pencarian informasi yang relevan dari berbagai macam dataset, seperti artikel, dokumen, dan database, yang telah di index ke Retrieval Tool. Retrieval tool akan mengembalikan informasi yang relevan dengan pertanyaan pengguna.



Augmented-Generation Component

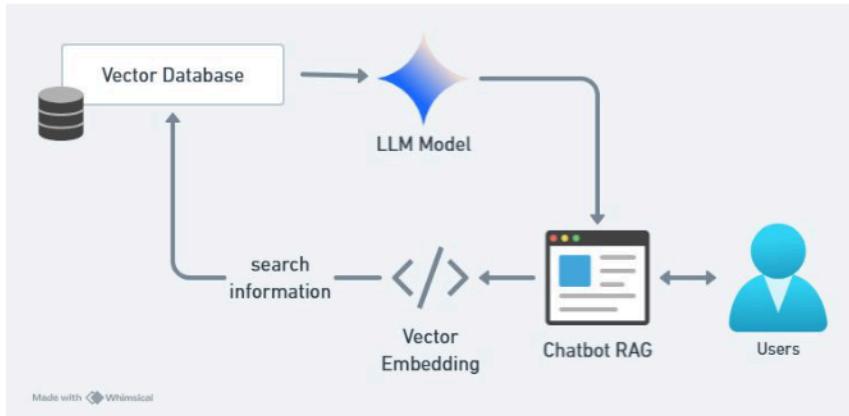
Berfokus pada memberikan response yang comprehensive sesuai dengan konteks informasi yang telah diberikan pada retrieval component. Generation component biasanya menggunakan LLM.

Manfaat RAG

- Mengurangi halusinasi LLM
- Memperbarui informasi secara real-time
tanpa pelatihan ulang model
- Mengimplementasikan chatbot dengan cara
yang lebih cepat, mudah, dan murah.

Lebih Detail Bagaimana RAG Bekerja

Secara lebih detail RAG bekerja sebagai berikut.



1. User bertanya pada chatbot
2. Kemudian chatbot akan mengubah pertanyaan dalam bentuk text tersebut menjadi vector
3. Vector akan digunakan untuk mencari topik yang relevan dengan pertanyaan user di vector database
4. Vector database kemudian akan mengeluarkan dokumen dokumentasi yang relevan
5. Dokumen tersebut dikirim ke LLM Model
6. LLM akan memproses dokumen tersebut dan menjawab pertanyaan user sesuai dengan referensi yang diberikan.
7. Output dari LLM akan dikirimkan kembali ke user.



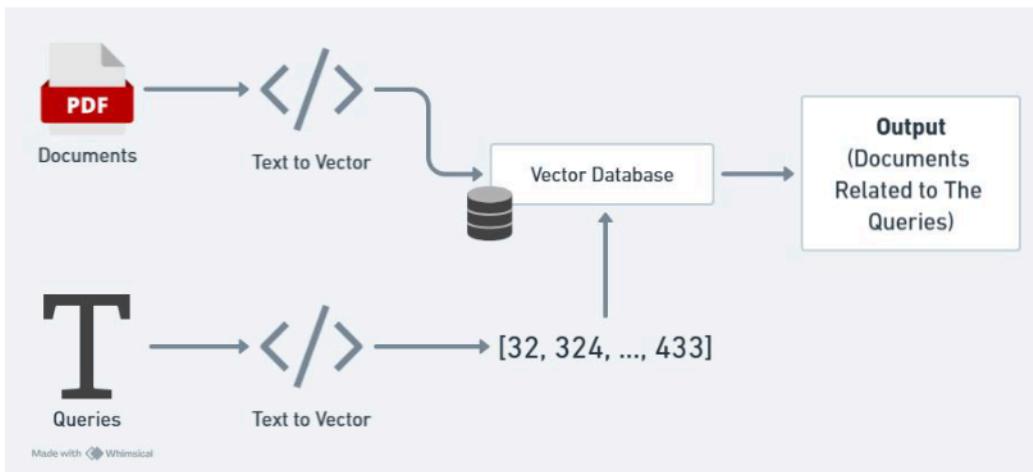
Vector Database

Apa itu Vector Database?

Vector Database adalah sistem penyimpanan data yang dioptimalkan untuk menyimpan, mengelola, dan mencari data dalam bentuk vektor (deretan angka).

Mirip seperti perpustakaan yang mengatur buku berdasarkan kesamaan topik, sehingga ketika user bertanya suatu topik, vector database akan memberikan data sesuai dengan topik yang ditanyakan oleh user.

Bagaimana Vector Database Bekerja?



Proses penyimpanan

Dokumen yang telah di transform ke dalam bentuk teks diubah ke vector kemudian disimpan dalam vector database

Proses pencarian

1. Pertanyaan dari user diubah terlebih dahulu ke dalam bentuk vector
2. Kemudian vector tersebut diinput ke pencarian vector database
3. Vector database melakukan pencarian vector terdekat menggunakan algoritma seperti cosine similarity
4. Top K Vector terdekat adalah outputnya

Komponen Utama pada Vector Database

Komponen Utama

1. **Embedding Model**
 - Contoh: BERT (teks), CLIP (gambar), Whisper (suara).
2. **Indexing Mechanism**
 - Algoritma: HNSW, IVF-PQ (*Faiss*), atau ANNOY untuk pencarian cepat.
3. **Similarity Metrics**
 - *Cosine Similarity*: Ukur sudut antar vektor.
 - *Euclidean Distance*: Jarak geometris antar titik.
4. **Storage Engine**
 - Format penyimpanan teroptimasi untuk operasi vektor.

Contoh Vector Database yang Sering Digunakan

Open Source:



Faiss



Chroma



Qdrant



CockroachDB

Enterprise:



Elastic Search



Azure AI Search



Vertex AI

Vertex AI Vector
Search



Contoh Kasus Penggunaan Database

Contoh Kasus Penggunaan

1. **RAG (Retrieval-Augmented Generation)**
 - o Cari dokumen relevan → Augmentasi ke LLM untuk jawaban akurat.
2. **Pencarian Gambar/Produk**
 - o Cari sepatu dengan desain mirip dari database 1 juta gambar dengan memanfaatkan pencarian vector
3. **Search Engine**
 - o Pengembangan Search Engine juga dapat menggunakan vector database sehingga topik yang muncul tidak bergantung pada "keyword" tapi berdasarkan kedekatan vector.

Maju Bareng AI

Presence - AI for Data Scientist

Buat yang belum absen,
absen yuk! WAJIB



<https://bit.ly/absensi-data-mba>



Pengenalan LangChain

Llama

Apa itu LangChain?

LangChain adalah framework Python untuk membangun aplikasi berbasis Language Model (LLM) seperti chatbot, generator teks, atau alat pemrosesan bahasa alami.

LangChain biasanya digunakan untuk menghubungkan LLM dengan komponen eksternal seperti:

- Model: LLM (contoh: Llama3, GPT, dll)
- Prompt: Instruksi/petunjuk untuk model
- Chain: Rangkaian operasi sederhana
- Memory: Penyimpanan percakapan

Contoh penggunaan dasar: membuat chatbot, generator teks, atau alat terjemahan.

Llama

Manfaat LangChain

1. Integrasi Mudah dengan Berbagai LLM

LangChain menyediakan kode yang konsisten untuk berbagai model bahasa, memudahkan developer dalam menguji dan mengimplementasikan model yang berbeda tanpa perubahan kode yang signifikan.

2. Pengembangan Aplikasi AI yang Lebih Cepat

Dengan komponen modular seperti Prompt Templates dan Chains, pengembang dapat membangun dan menguji prototipe aplikasi AI dengan lebih efisien.

3. Fleksibilitas dalam Integrasi Data

Kemampuan untuk menghubungkan LLM dengan sumber data eksternal memungkinkan aplikasi untuk memberikan respons yang lebih relevan dan ter-update.

Llama

Komponen Dasar LangChain: Prompt Template

```
from langchain_core.prompts import ChatPromptTemplate

system_template = "Translate the following from {language_from} into {language_to}"

prompt_template = ChatPromptTemplate.from_messages(
    [("system", system_template), ("user", "{text}")])
)

prompt = prompt_template.invoke({"language_from": "Indonesian", "language_to": "Italian",
                                 "text": "Saya saat ini sedang belajar generative AI"})
```

Apa itu Prompt Template? Prompt Template adalah salah satu komponen dasar pada LangChain yang membantu developer dalam penulisan template. Prompt template memungkinkan developer untuk membuat pola teks yang mengandung placeholder (penanda) untuk input yang dinamis. Manfaat dari prompt template ini adalah:

1. Menjaga konsistensi instruksi ke model
2. Memudahkan penggunaan ulang pola yang sama
3. Mengontrol format output



Llama

Llama

Apa itu Llama?

Llama adalah singkatan dari Large Language Model Meta AI. Seperti nama panjangnya, Llama adalah Large Language Model yang dirancang oleh Meta untuk menjadi model bahasa yang efisien dan berorientasi pada penelitian, dengan berbagai ukuran model yang dapat disesuaikan dengan kebutuhan dan sumber daya komputasi yang berbeda.

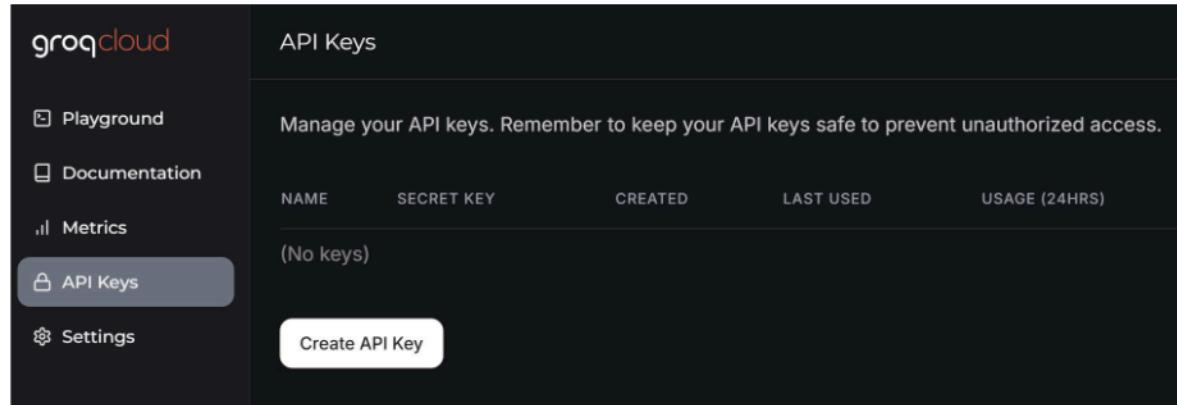
Kegunaan Llama pada dasarnya sama seperti LLM yang lain, yaitu text generation, summarization, code generation, dan lainnya. Hanya saja, sifatnya yang open source membuat siapa saja bisa dengan mudah mengakses Llama.



Llama

Llama Installation & Setup

1. Buka <https://console.groq.com/> kemudian login dengan akun yang telah dibuat. Selanjutnya, kamu akan diarahkan ke Playground dimana kamu bisa mencoba berbagai model yang tersedia.
2. Di bagian sidebar ada pilihan API Keys, click, lalu pilih create API Keys. Seperti pada screenshot di bawah. Simpan API Key di tempat yang aman karena kamu hanya bisa mengaksesnya sekali.
3. API Key tersebut dapat kamu gunakan untuk mengakses Llama melalui environment yang kamu bangun.



Llama

Parameter Tuning pada LLama

Parameter yang digunakan Llama juga sama seperti parameter pada LLM lain. Dengan parameter tuning, kamu dapat mengatur jawaban pada Llama sesuai dengan kebutuhanmu. Beberapa parameter yang bisa kamu coba:

Temperature:

- Mengontrol kreativitas dalam output. Semakin tinggi nilainya semakin kreatif jawaban yang akan diberikan oleh model.

Max Tokens:

- Membatasi jumlah token (kata/frase) dalam satu output. Jika nilainya terlalu kecil, output bisa terpotong sebelum selesai. Perlu diatur sesuai kebutuhan agar respons tidak terlalu pendek atau boros token.

Top-p (Nucleus Sampling)

- Alternatif dari temperature, membatasi pemilihan kata hanya dari probabilitas tertinggi.
- Karena Top-p dan temperature memiliki fungsi yang sama, jadi bisa dipilih salah satunya saja.

Llama

Parameter Tuning pada LLama

Parameter yang digunakan Llama juga sama seperti parameter pada LLM lain. Dengan parameter tuning, kamu dapat mengatur jawaban pada Llama sesuai dengan kebutuhanmu. Beberapa parameter yang bisa kamu coba:

Frequency Penalty

- Mencegah pengulangan kata atau frasa yang berlebihan. Nilai tinggi: Model cenderung lebih variatif dalam pemilihan kata.

Presence Penalty

- Mendorong model untuk menyebutkan kata-kata baru yang belum muncul dalam respons. Cocok untuk eksplorasi ide baru dalam percakapan.

Maju Bareng AI

Presence - AI for Data Scientist

Buat yang belum absen,
absen yuk! WAJIB



<https://bit.ly/absensi-data-mba>

**Jangan Lupa Untuk Cek
Dokumentasi Code Groq Cloud!**

<https://console.groq.com/docs/quickstart>



Exercise

Hands-on

Latihan - Implementasi RAG

- **RAG-Llama:**

https://colab.research.google.com/drive/1b4xHmOBmDvG78tIQEfBKH94ddPURRm_N?usp=sharing

- **RAG-Gemini:**

<https://colab.research.google.com/drive/1QGcfi3ocdDofXi0WINVDO1s9kD6cmz8t?usp=sharing>

Jangan lupa untuk duplicate
Notebooknya ke Drive kamu ya!



Quiz

LLM-Based Tools and Gemini API Integration for Data Scientists

Quiz Session 4



<https://bit.ly/quizdata4>

Due Date:
Hari ini, 23.59 WIB

Ingat, absensi juga dihitung
dari pengisian Quiz.

Certificate from Hacktiv8

Phase	Task	Due Date
Session 1	Presence	Session 1
Session 2	Presence & Quiz 1	Session 2
Session 3	Presence & Quiz 2	Session 3
Session 4	Presence & Quiz 3	Session 4
Session 5	Presence & Quiz 4	Session 5
Final Project	Form Submission: Final Project	H+2 Session 5

✨ Special Reward! ✨

Di akhir program, akan ada merchandise yang dibagikan untuk peserta yang beruntung yang telah menyelesaikan seluruh rangkaian program* 🎁

*Satu pemenang untuk satu training topics.



Thank You

Hacktiv8

www.hacktiv8.com

(021) 8067 5787
halo@hacktiv8.com