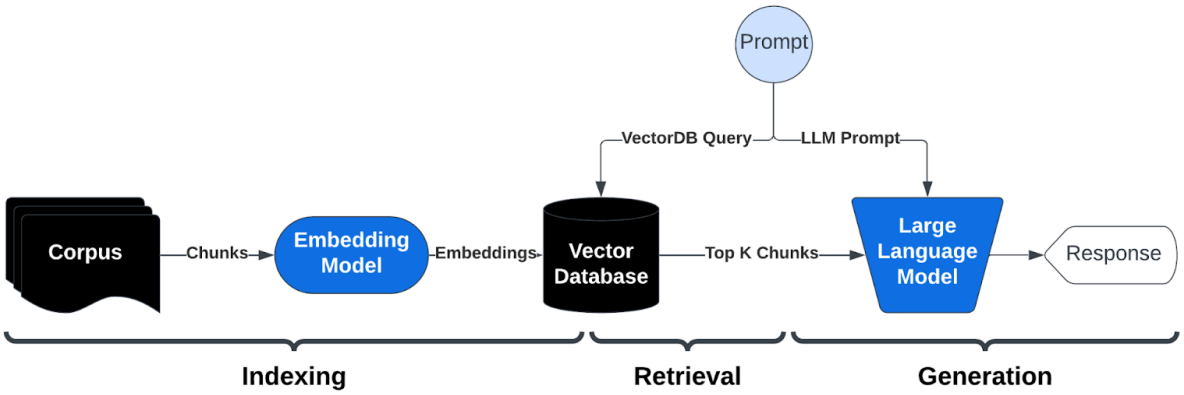
Penjelasan

Latar belakang

- AI memerlukan pengetahuan yang spesifik bagi suatu bidang untuk menjawab pertanyaan terkait bidang tersebut
    - Misalnya: sebuah aplikasi bantuan *chat* untuk keuangan harus memahami tren pasar dan produk yang ditawarkan oleh bank tertentu.
- Solusi umum: RAG (*Retrieval-Augmented Generation*)
    - Mengambil data relevan dari *knowledge base*
    - menggabungkannya dengan permintaan pengguna
    - meningkatkan kualitas model
- Pengetahuan bisnis sering terdapat dalam format seperti PDF, presentasi PowerPoint, atau *scan document*
    - Sulit untuk mengekstrak dan menyiapkan bagian-bagian yang relevan untuk diinjeksikan ke dalam *prompt* yang dapat dikirim ke model LLM

Ringkasan RAG Dasar

- RAG (*Retrieval-Augmented Generation*): model mengakses dan menggunakan pengetahuan eksternal dalam jumlah besar.
- Memproses *knowledge base* besar dan secara dinamis mengambil informasi relevan pada saat *runtime*



Efficient Document Retrieval Menggunakan *Vision Language Models*

- ColPali: sebuah pendekatan baru untuk *image retrieval* yang menghemat proses *document retrieval* dengan *vision language models*.
    - *Direct indexing* & *embedding* dari *document pages* sebagai gambar-gambar (kotak merah muda)
    - *Retrieval* berdasarkan *visual semantic similarity* (kotak hijau)

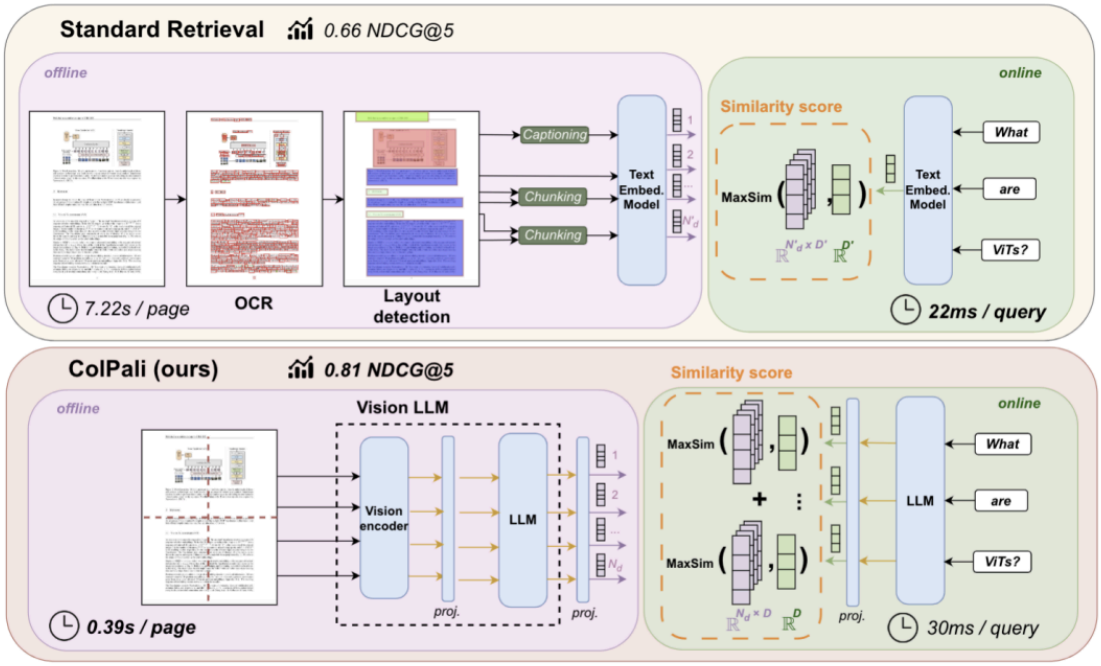
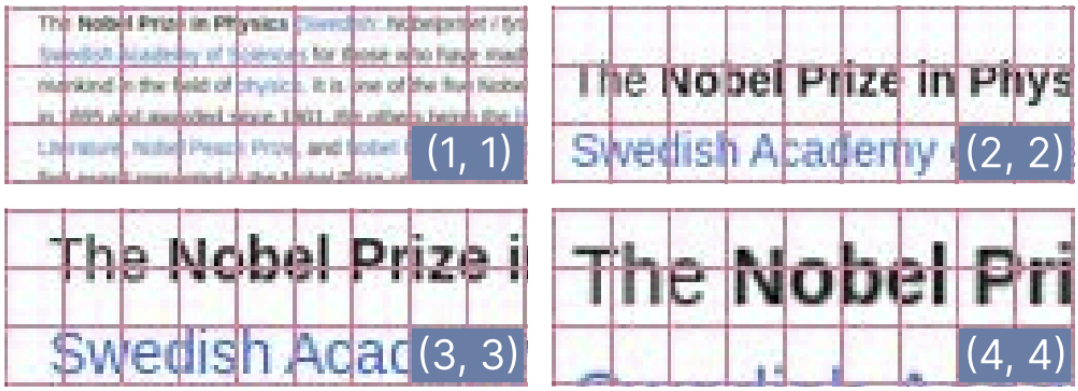


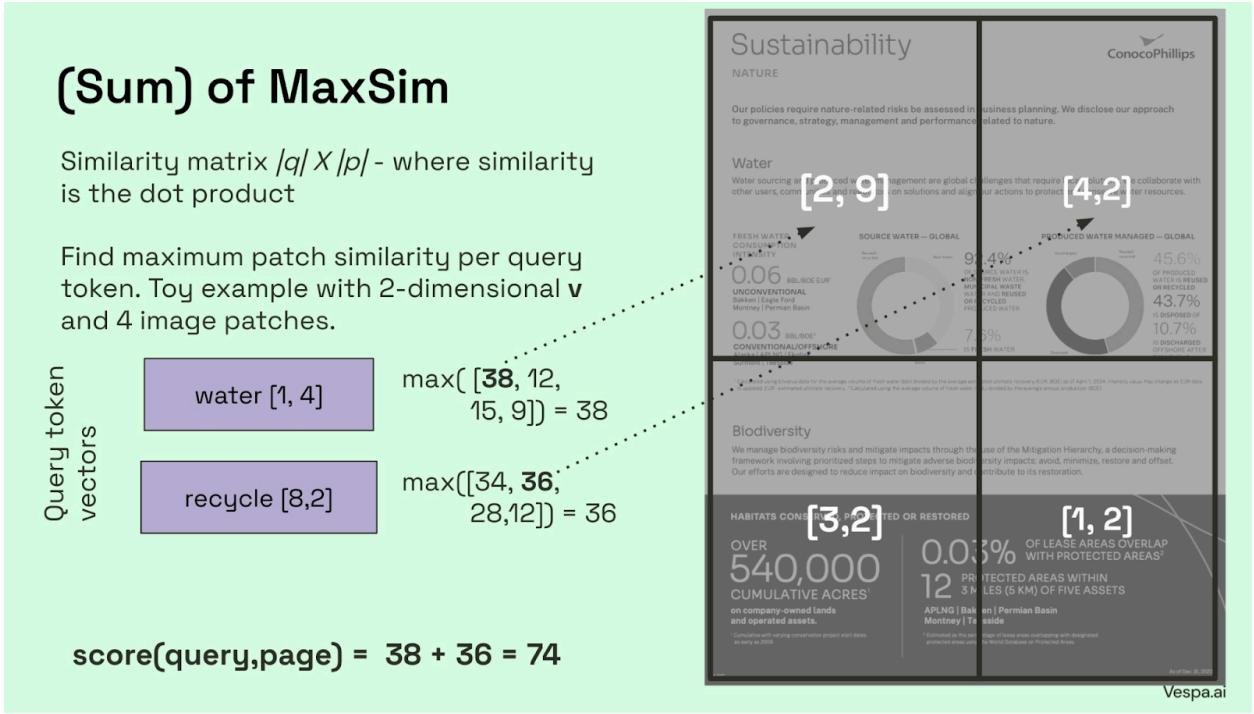
Figure 2: *ColPali* simplifies document retrieval w.r.t. standard retrieval methods while achieving stronger performances with better latencies. Latencies and results are detailed in section 5 and subsection B.5.

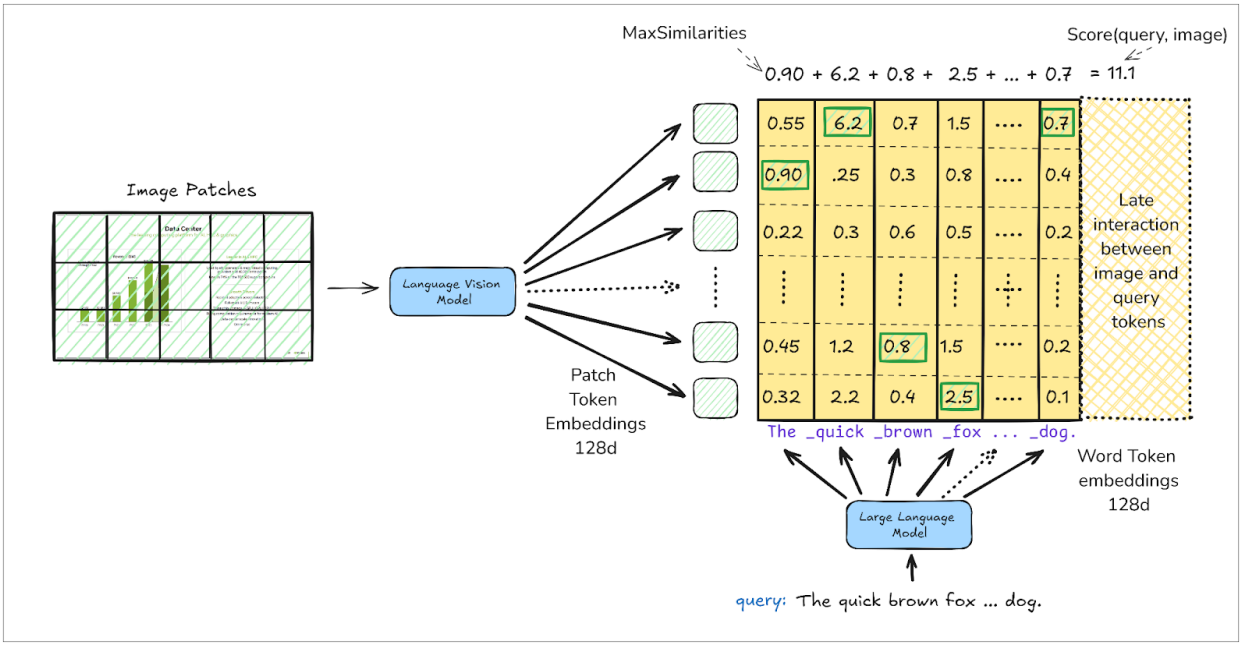
Cara Kerja ColPali:

- Indeks *knowledge base*:
    - *Encoder* membagi setiap gambar menjadi *patches* dan menyimpan informasi tekstual dan visual sebagai vektor
    - Vektor-vektor ini dapat disimpan dengan efisien dalam basis data vektor untuk *retrieval* cepat.



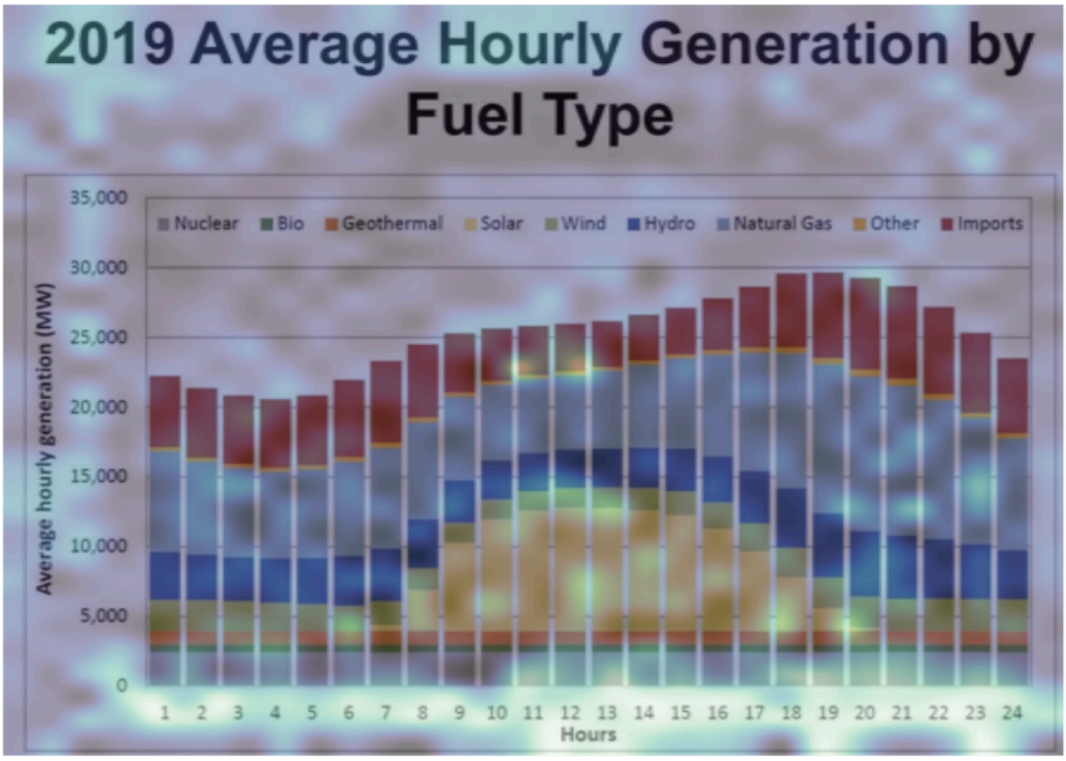
- *Query Processing & Retrieval*:
    - Ketika pengguna memasukkan *prompt*, *prompt* tersebut diproses dan *query* diekstrak
    - Sistem kemudian mencari dalam basis data vektor untuk potongan-potongan teks yang semantiknya sama dengan *query*





- Interaksi antara *vision tokens* dengan *language tokens* ini memungkinkan interaksi semantik yang sangat kaya antara *query* dan dokumen untuk mendapatkan kesamaan.
- Proses ini menghasilkan *retrieval* dan *ranking* halaman dokumen yang paling relevan terhadap permintaan
- ColPali dapat memproduksi *heatmap* semantik (menampilkan bagian dokumen yang paling dekat dengan *query*), sehingga memberikan pengguna wawasan intuitif tentang proses *retrieval*.
- Potongan-potongan yang paling relevan diambil dan disematkan ke dalam *prompt* yang dikirim ke model generatif AI

- *Response Generation*:
  - Model AI kemudian menggunakan informasi yang diperoleh dan pengetahuan *pre-trained* untuk menghasilkan respons
  - Konteks yang diberikan mengurangi halusinasi dan kita dapat memberi tahu sumber dari jawaban kita



Query: "Which hour of the day had the highest overall eletricity generation in 2019?"

- ColPali menganggap semua dokumen menjadi gambar
    - Setiap dokumen diperlakukan sama tanpa perlu penanganan spesifik
    - Pendekatan ini menyimpan *layout* asli dokumen (ini sangat penting dalam menjaga konteks dan arti) terutama pada dokumen yang kaya visual -> pemahaman menyeluruh dari isi dokumen
    - Kemampuan ini sangat berguna saat menghadapi dokumen yang kombinasi teks, grafik, diagram, dan data visual lainnya.

Kekurangan ColPali

- Kelemahan ColPali: harus berhadapan dengan lebih banyak vektor dibandingkan dengan metode konvensional.

- Salah satu cara mengatasi: DSE (Document Screenshot Embedding)
- DSE menggunakan pendekatan *bi-encoder* untuk *image retrieval*, di mana semua vektor *patch* gambar disederhanakan ke dalam satu vektor yang sama dengan *query*
- Menggunakan metrik jarak seperti kosinus atau euclidean -> Kesamaan antara vektor gambar dan vektor *query*
- Kekurangannya adalah vektornya tidak semantik kaya seperti pendekatan multi-vektor per halaman dokumen.

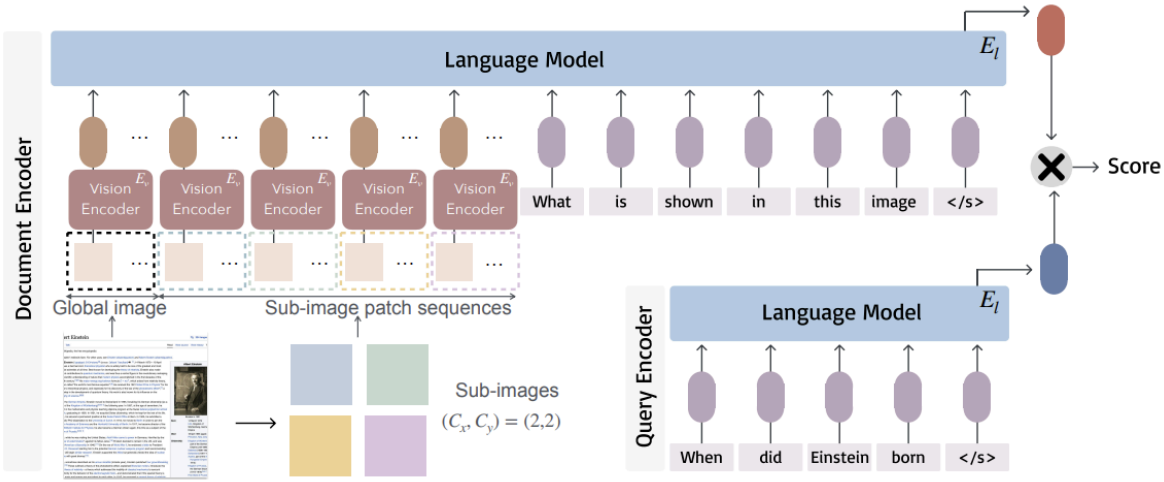
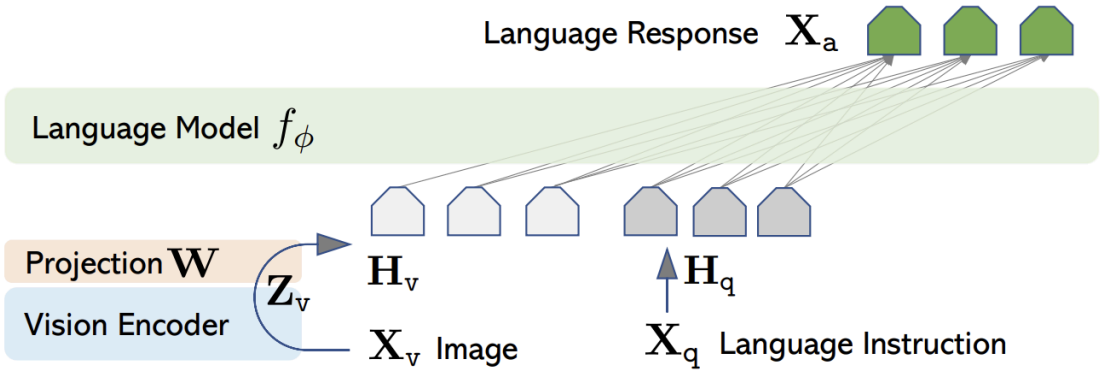


Figure 2: Overview of DSE encoder architecture. DSE adopts a bi-encoder architecture, where the document tower encodes the document screenshot into dense vector by taking vision input and the query tower encodes the query by taking text input. Document and query encoders share the same language model.

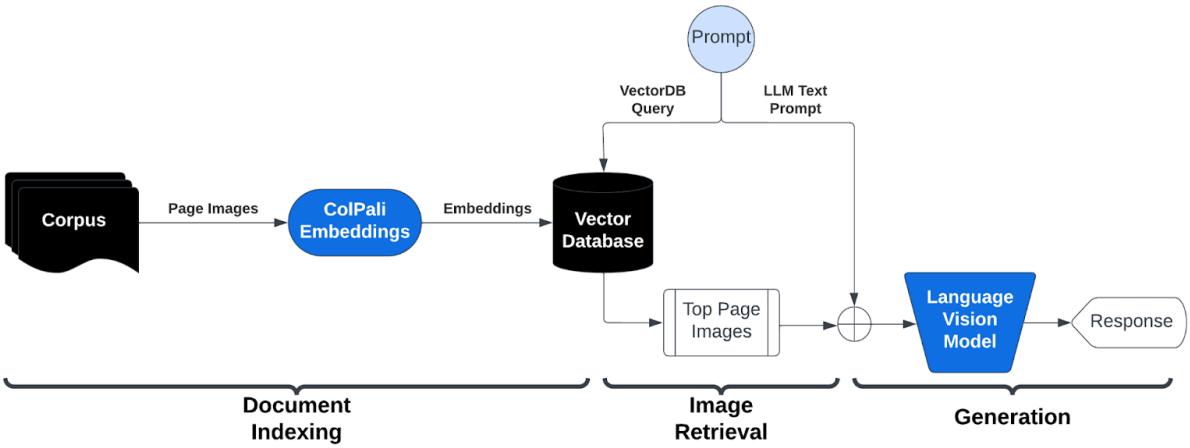
Menggunakan Llama 3.2 Vision untuk Pemahaman Gambar

- ColPali me-*retrieve* dan me-*ranking* halaman dokumen yang relevan berdasarkan *query*
- ColPali
    - Bisa tahu gambar/halaman mana jawaban/konten relevan berada
    - Tidak akan menghasilkan jawaban dari pertanyaan tersebut.
- Seri baru Llama 3.2 *vision* menggunakan teknik disebut *instruksi visual tuning*
    - model bahasa bisa“melihat” dan memproses gambar
- Kemampuan *vision* LLM didapat dengan menerapkan token-token gambar ke dalam ruang-ruang laten yang sama seperti token-token teks dan melatih untuk menyatukannya.

- *Efficiency retrieval* ColPali + Llama 3.2
    - bisa menemukan halaman/gambar yang benar
    - memahami dan menjawab pertanyaan tentang konten-konten mereka

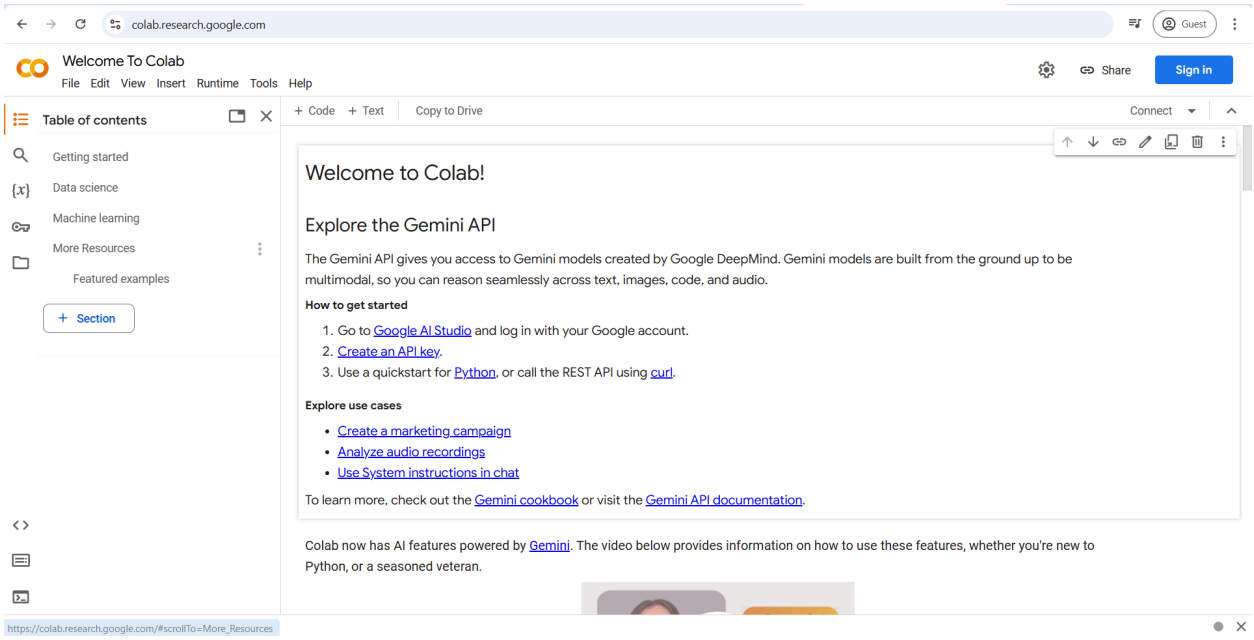


- Setelah ColPali mengidentifikasi halaman relevan teratas untuk permintaan tertentu, kita bisa melewatkan halaman-halaman tersebut bersamaan dengan *prompt* ke Llama 3.2 untuk *generation*.

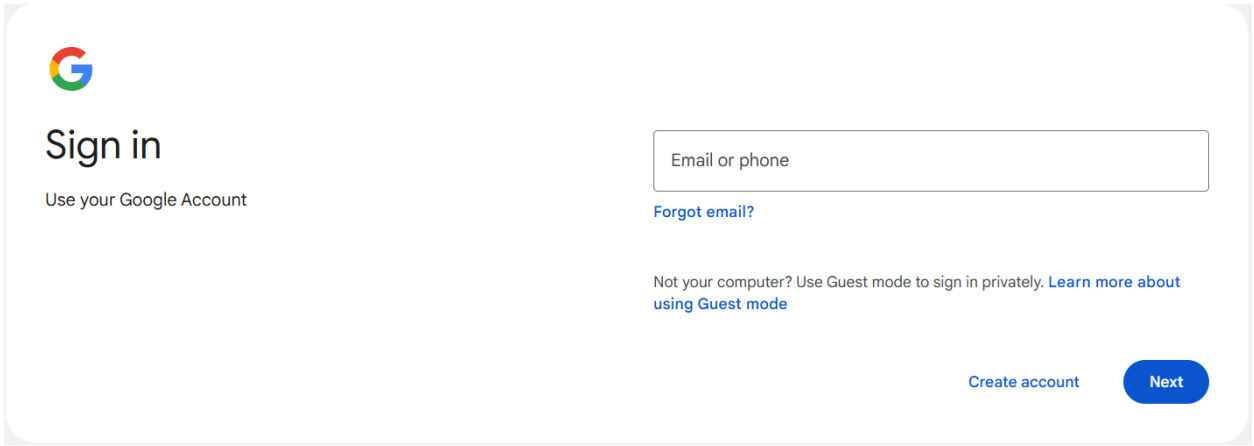


Login colab

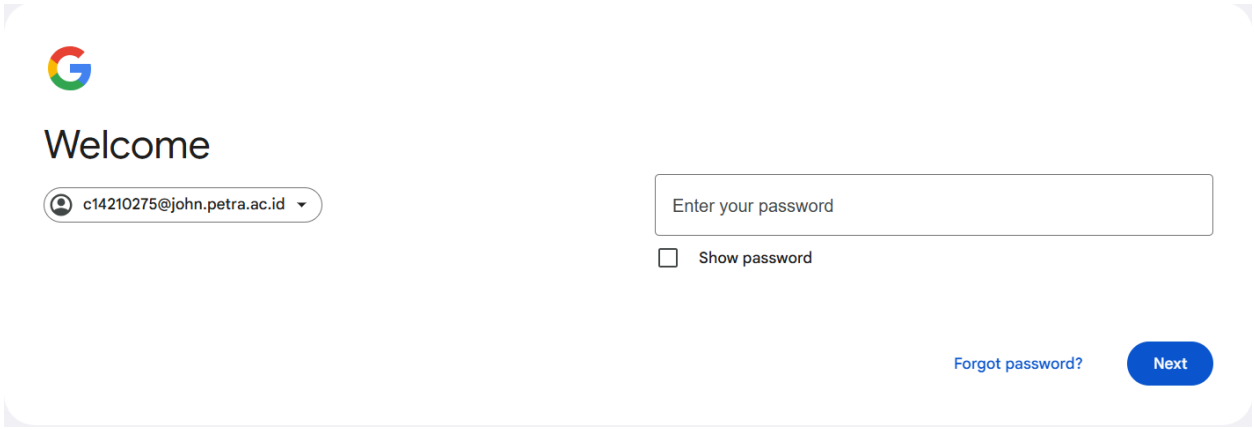
- colab.research.google.com



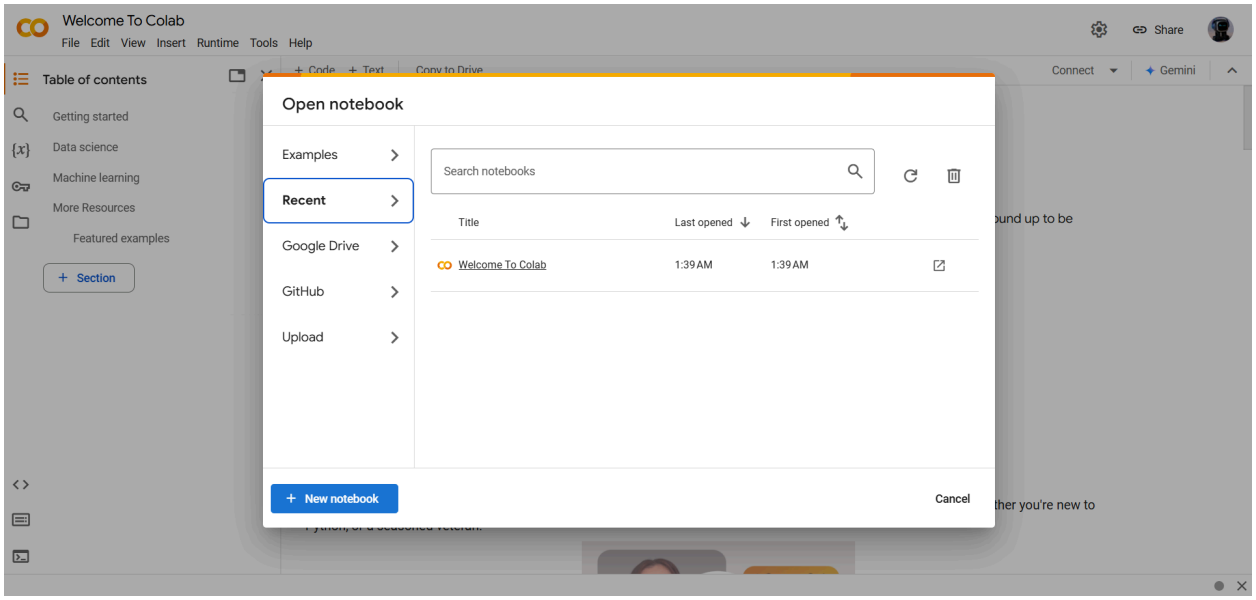
- Tekan tombol *sign in*



- Masukkan surel



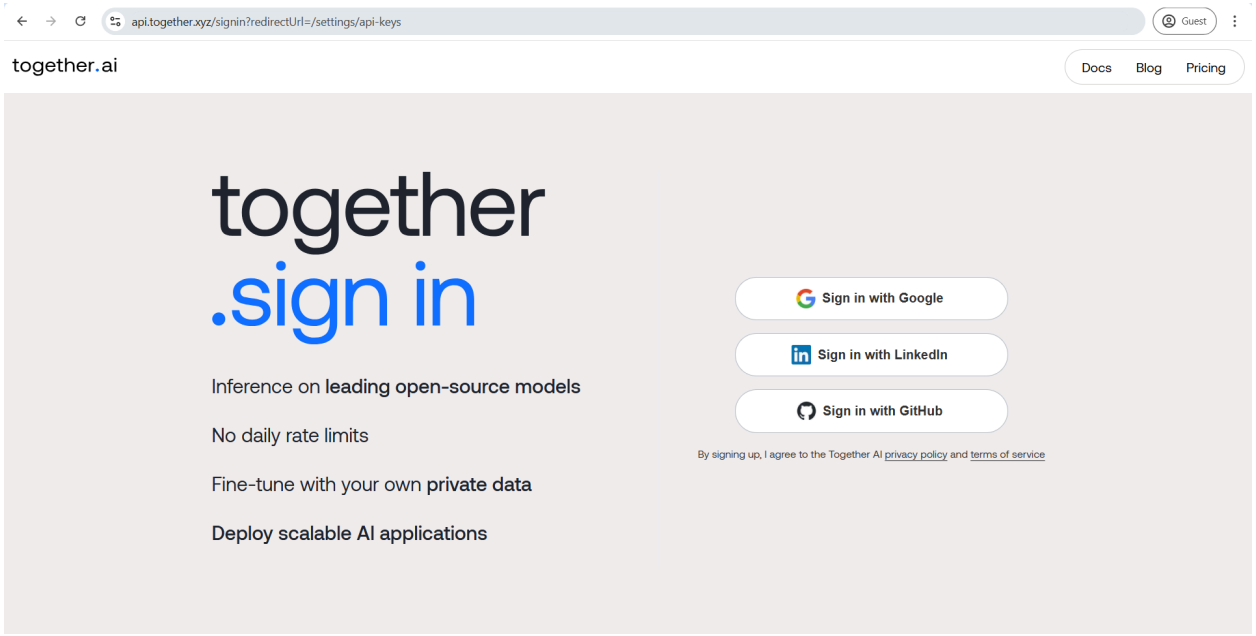
- Masukkan kata sandi



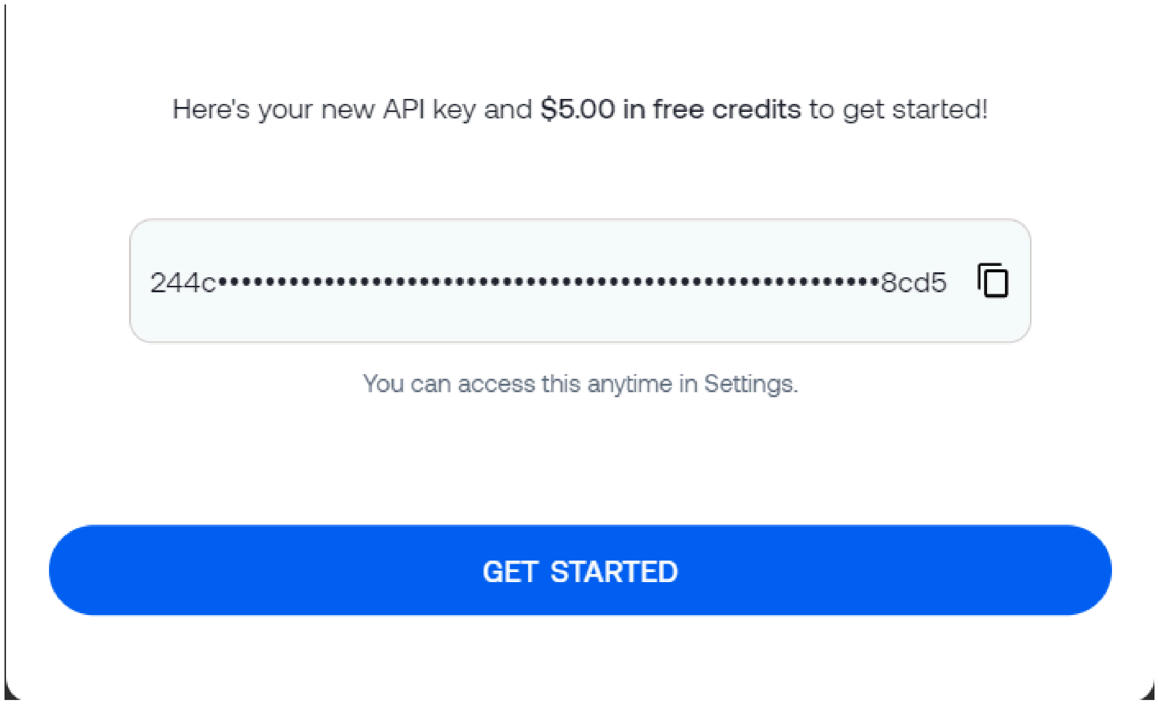
- Berhasil

Together API Key

https://api.together.xyz/settings/api-keys
Gunakan link tersebut untuk mendapatkan kunci



Sign in dengan akun kalian



Salin API key yang didapat

Cara gunakan berkas colabnya

[Link github](#)
[Link colab](#)

Unduh pustaka yang dibutuhkan

```
[ ] !pip install byaldi together pdf2image
```

```
[ ] !sudo apt-get install -y poppler-utils
```

```
[ ] # Paste in your Together AI API Key or load it
    api_key = os.environ.get("TOGETHER_API_KEY")
```

Ganti isi variabel api_key dengan api key yang sudah ada di *clipboard* kalian
Contoh:
api_key = “aaaaaaaaaaaaaaa”

Inisialisasi Model ColPali

```python
import os
from pathlib import Path
from byaldi import RAGMultiModalModel

# Initialize RAGMultiModalModel
model = RAGMultiModalModel.from_pretrained("vidore/colqwen2-v0.1")
```

Dapatkan contoh dokumen untuk percobaan

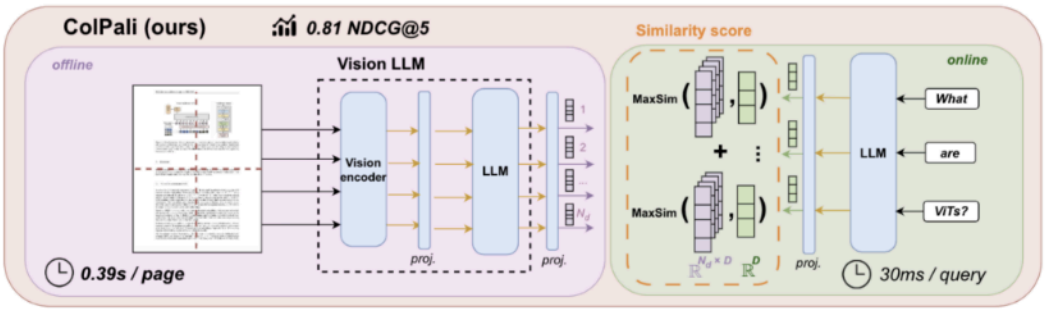
```python
# Dowload and rename the last presentation from Nvidia to investors
!wget https://s201.q4cdn.com/141608511/files/doc_presentations/2023/Oct/01/ndr_presentation_oct_2023_final.pdf
!mv ndr_presentation_oct_2023_final.pdf nvidia_presentation.pdf
```

```
--2024-10-04 14:34:23--  https://s201.q4cdn.com/141608511/files/doc_presentations/2023/Oct/01/ndr_presentation_oct_2023_final.pdf
Resolving s201.q4cdn.com (s201.q4cdn.com)... 68.70.205.3, 68.70.205.4, 68.70.205.1, ...
Connecting to s201.q4cdn.com (s201.q4cdn.com)|68.70.205.3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8609256 (8.2M) [application/pdf]
Saving to: ‘ndr_presentation_oct_2023_final.pdf’

ndr_presentation_oc 100%[===================>]   8.21M  24.4MB/s    in 0.3s

2024-10-04 14:34:24 (24.4 MB/s) - ‘ndr_presentation_oct_2023_final.pdf’ saved [8609256/8609256]
```

Buat indeks dari gambar-gambar

```python
# Use ColQwen2 to index and store the presentation
index_name = "nvidia_index"
model.index(input_path=Path("/content/nvidia_presentation.pdf"),
    index_name=index_name,
    store_collection_with_index=True, # Stores base64 images along with the vectors
    overwrite=True
)
```

This concludes the indexing of the PDF phase - everything below happens at query time.

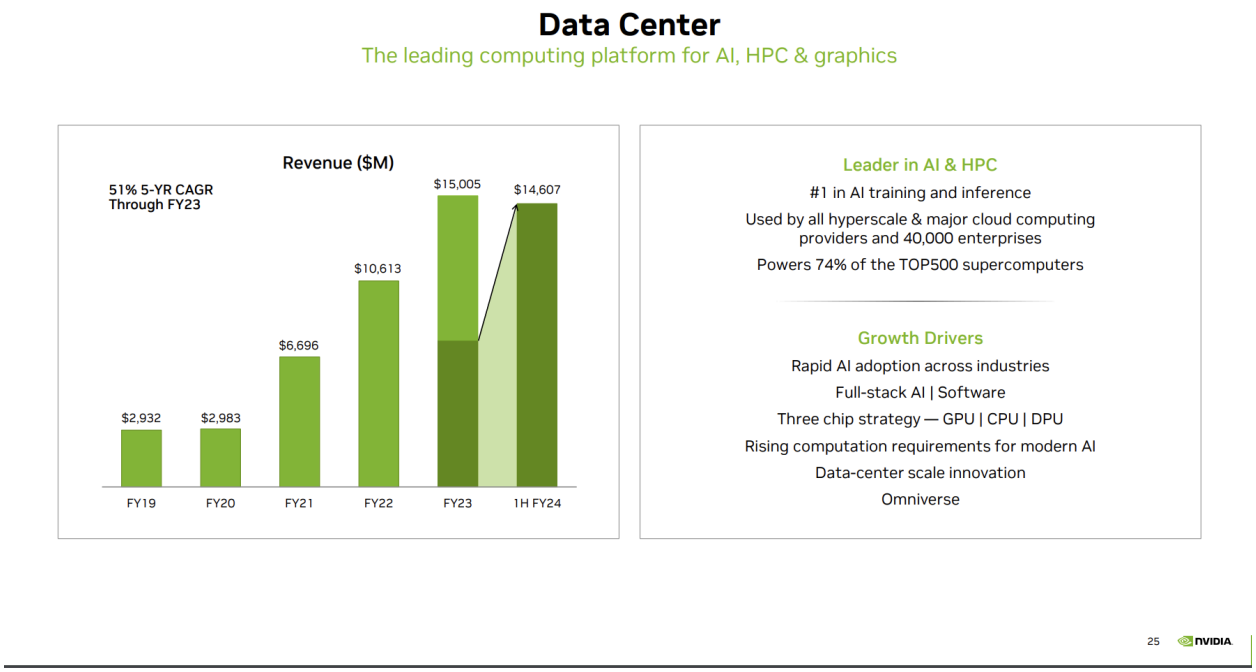


Query dari gambar-gambar yang telah disimpan

```python
# Lets query our index and retrieve the page that has content with the highest similarity to the query

# The Data Centre revenue results are on page 25 - for context!
query = "What are the half year data centre renevue results and the 5 year CAGR for Nvidia data centre revenue?"
results = model.search(query, k=5)

print(f"Search results for '{query}':")
for result in results:
    print(f"Doc ID: {result.doc_id}, Page: {result.page_num}, Score: {result.score}")

print("Test completed successfully!")
```

```
Search results for 'What are the half year data centre renevue results and the 5 year CAGR for Nvidia data centre revenue?':
Doc ID: 0, Page: 25, Score: 25.875
Doc ID: 0, Page: 24, Score: 25.0
Doc ID: 0, Page: 28, Score: 23.75
Doc ID: 0, Page: 32, Score: 23.75
Doc ID: 0, Page: 31, Score: 23.75
Test completed successfully!
```

Gambar yang tepat:

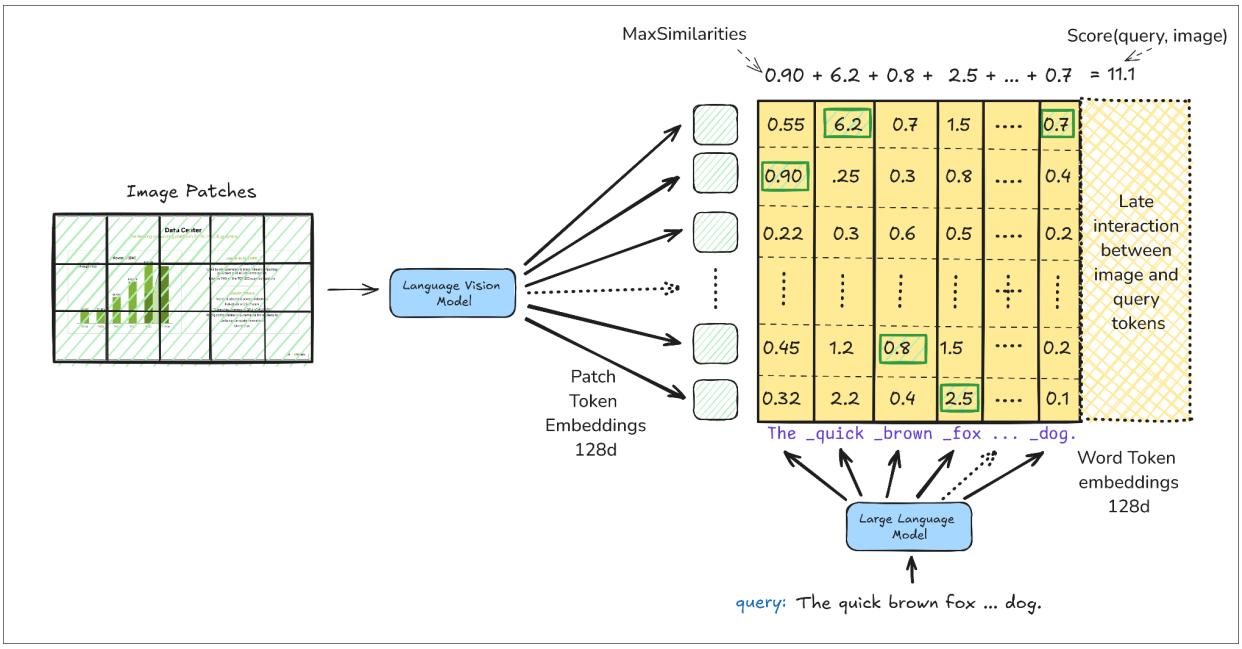


Interaksi antara gambar dan *query text* untuk memberikan nilai pada setiap halaman
Gambar *diresize* ke 16x16
Lakukan MaxSim

Gambar prediksi

```
# Since we stored the collection along with the index we have the base64 images of all PDF pages aswell!
model.search(query, k=1)
```

```
returned_page = model.search(query, k=1)[0].base64
```



Llama 3.2 dapatkan jawaban dari retrieved_page

```python
import os
from together import Together

client = Together(api_key = api_key)

response = client.chat.completions.create(
  model="meta-llama/Llama-3.2-90B-Vision-Instruct-Turbo",
  messages=[
    {
      "role": "user",
      "content": [
        {"type": "text", "text": query}, #query
        {
          "type": "image_url",
          "image_url": {
            "url": f"{returned_page}", #retrieved page image
          },
        },
      ],
    }
  ],
  max_tokens=300,
)

print(response.choices[0].message.content)
```

Referensi