Langgraph & Langchain

1. LLM Based Systems

$f(\text{input}) = y(\text{train})$ conditional

Training data

M	R
0.02	Y
90	Y
0.3	N

0.87 = (Y/N)

NLP

ML → Discriminative = $f(x) = y$ (Labelling) $P(y|x)$

→ Generative $P(x,y)$ = joint prob

Labelled data X

x = carry an umbrella

y = it rains

new generation

Autoregressive

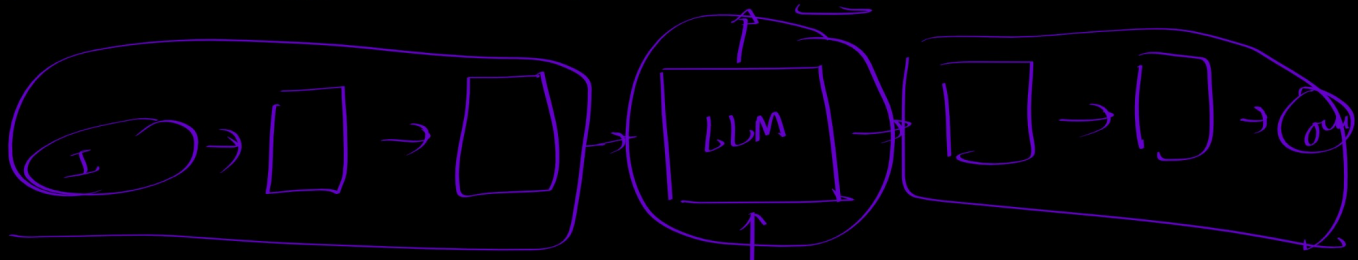
Sonnet 4 → 2 → Trg data

→ 1 → NLP (Where the mind is) ? (without)

GPT 4o

Gemini 2.5 ← 0 → Foundation model

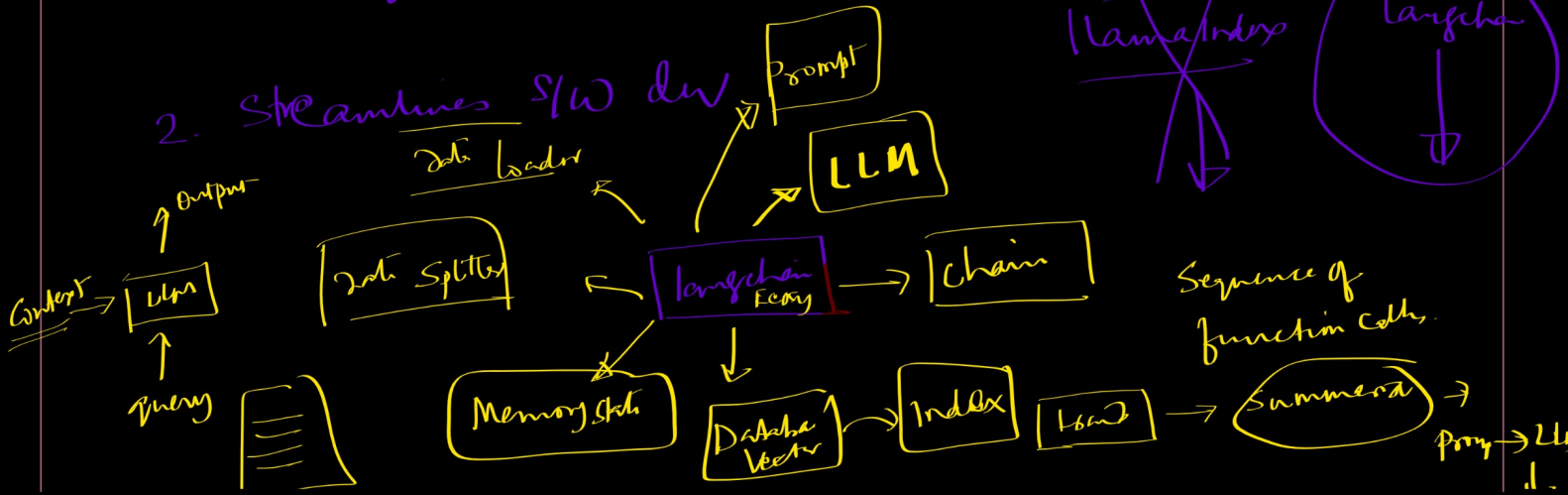
1. LLM based system



1. Tight coupling of LLM with the rest of the system

Where the mind is

2. Streamlines S/W dev



LlamaIndex

Langchain

Reoru

Vector db

Libraries
- Langchain → Linear flow of execution → memo
- Langgraph → workflows → non linear
- Langsmith → Test, monitor, observability
- Langserve → Fast API
- → memory
- memos

Chain = Data ld | Split | Emb | Promp | LLM | Parse

Persona = OpenAI

LLM

Where the mind is without fear

1. Corpus as input → Training set → Internet → Wiki, Reddit; → Books; → libraries

2. Tokenization — where | the | mind | is

where | stand

BPE

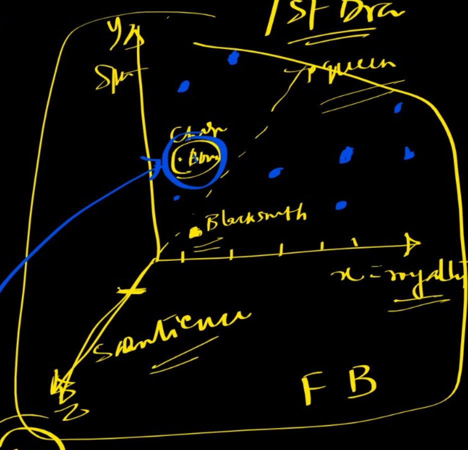
Studying, Standing, stand, stood → Stand

Stand, Sta, Stan, Queen

royal, king, queen

3. Embeddings

king, Country, 1st Brain



LLM

Semantics

Embedding

| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 12 |

embedding size (3)

feature dim

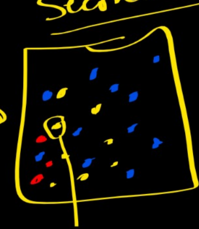
1088

Second Brain

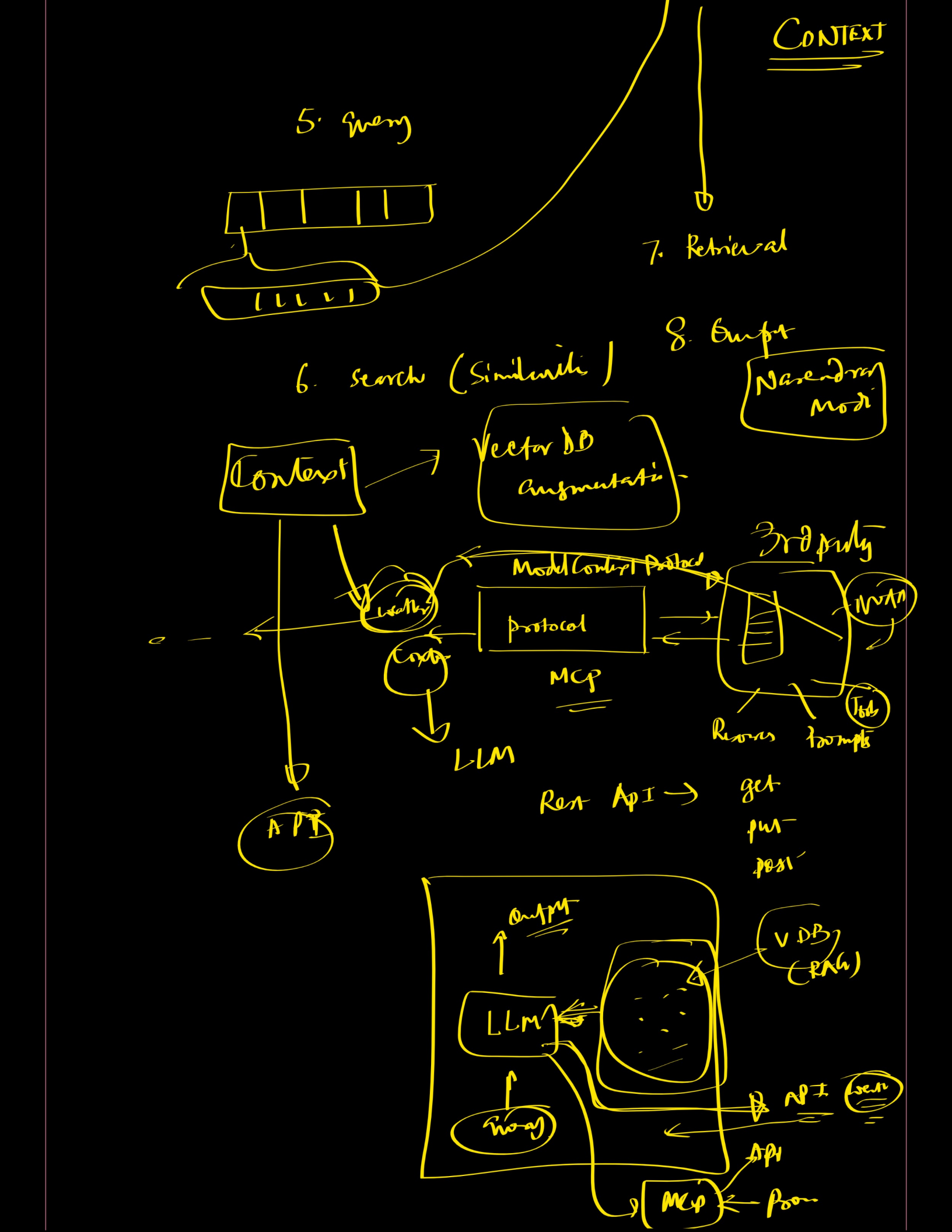
RAG

4. Vector DB

brings →



Vectordb


CONTEXT
5. Query
7. Retrieval
6. search (Similarity)
8. Output
Narendra Modi
Context
Vector DB
Augmentation
3rd party
Model Context Protocol
protocol
MCP
LLM
API
Rest API → get post
output
LLM
V DB (RAG)
API
Query
MCP
Prom


Data