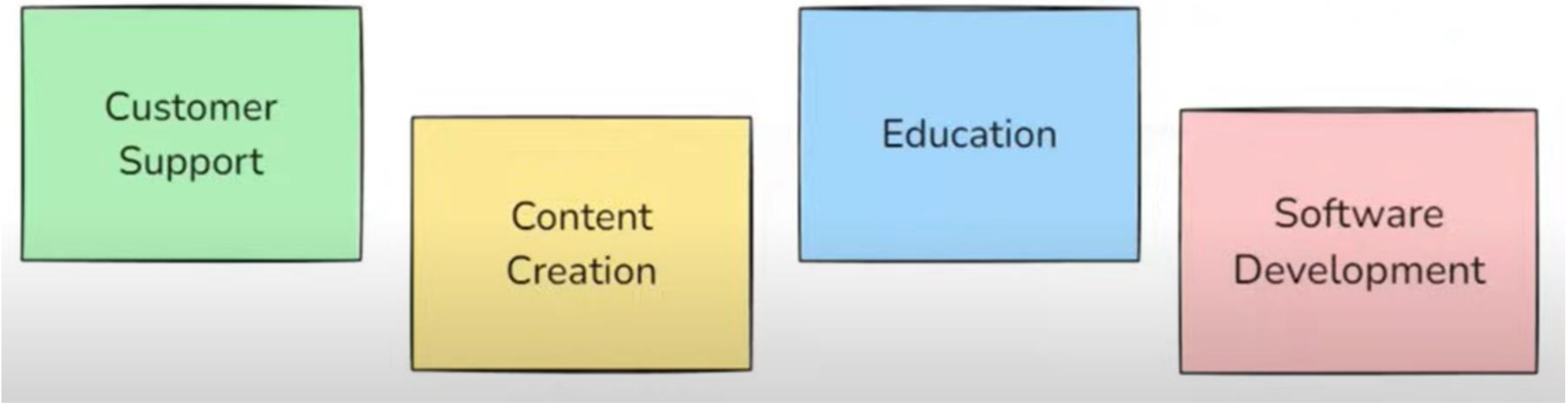
Chat Models-Introdction

What is GenAI

Generative AI is a type of artificial intelligence that creates new content—such as text, images, music, or code—by learning patterns from existing data, mimicking human creativity.

GenAI Impact areas



Is GenAI Successful ?

Does it solve real world problems?

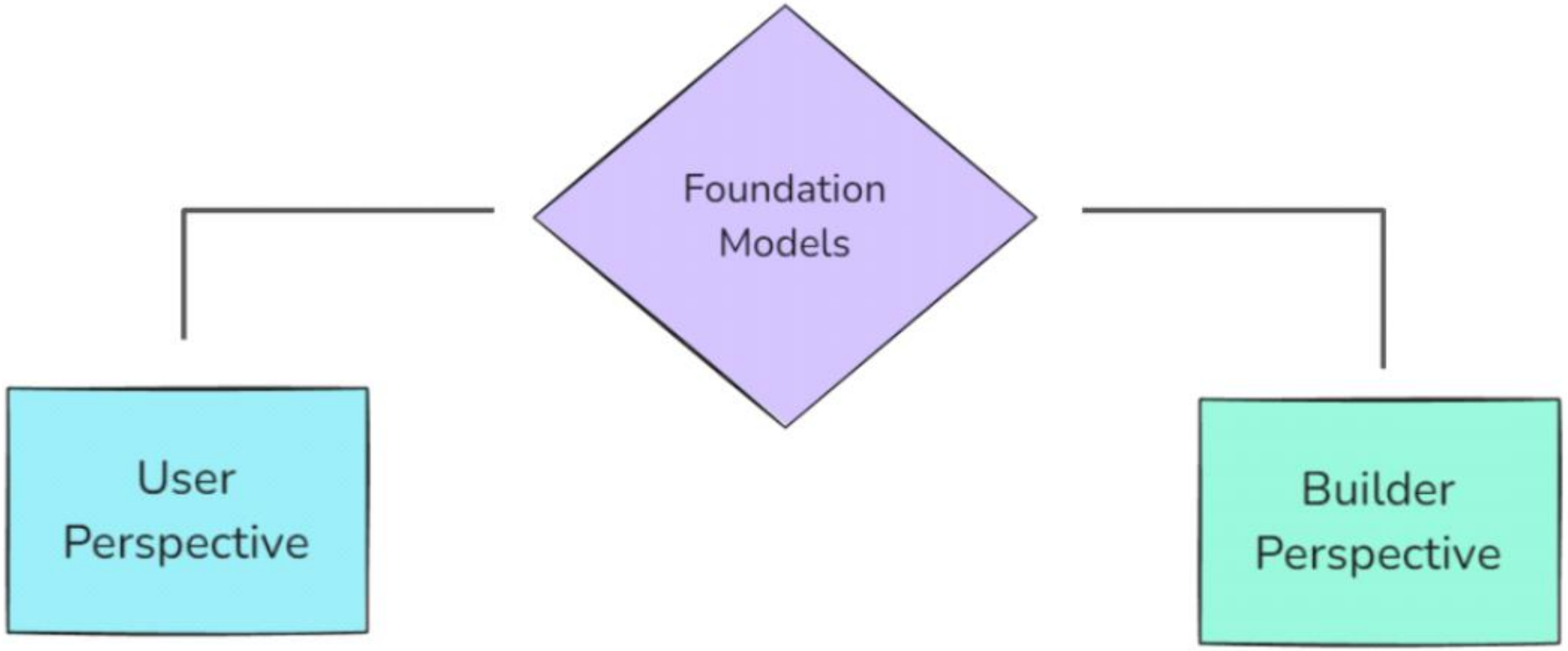
Is it useful on a daily basis?

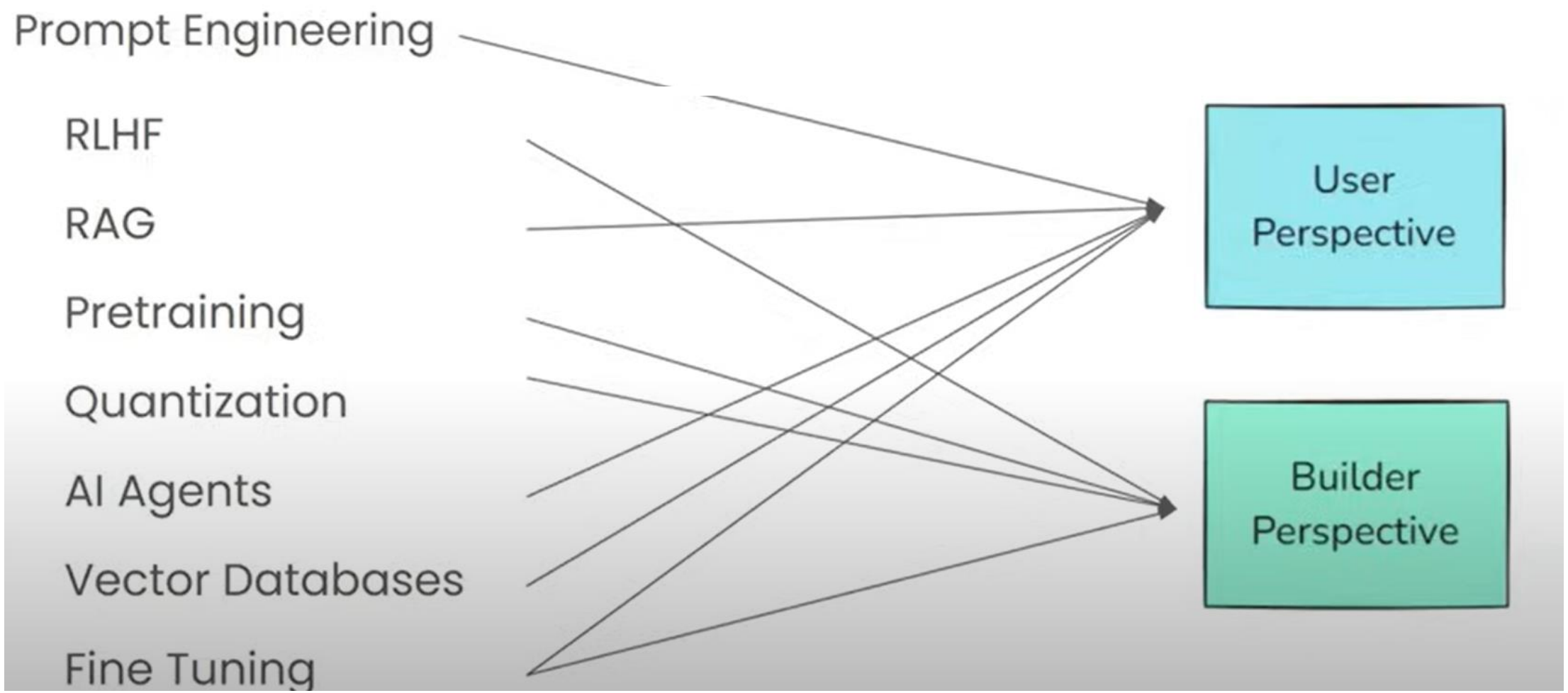
Is it impacting the world economics?

Is it creating new jobs?

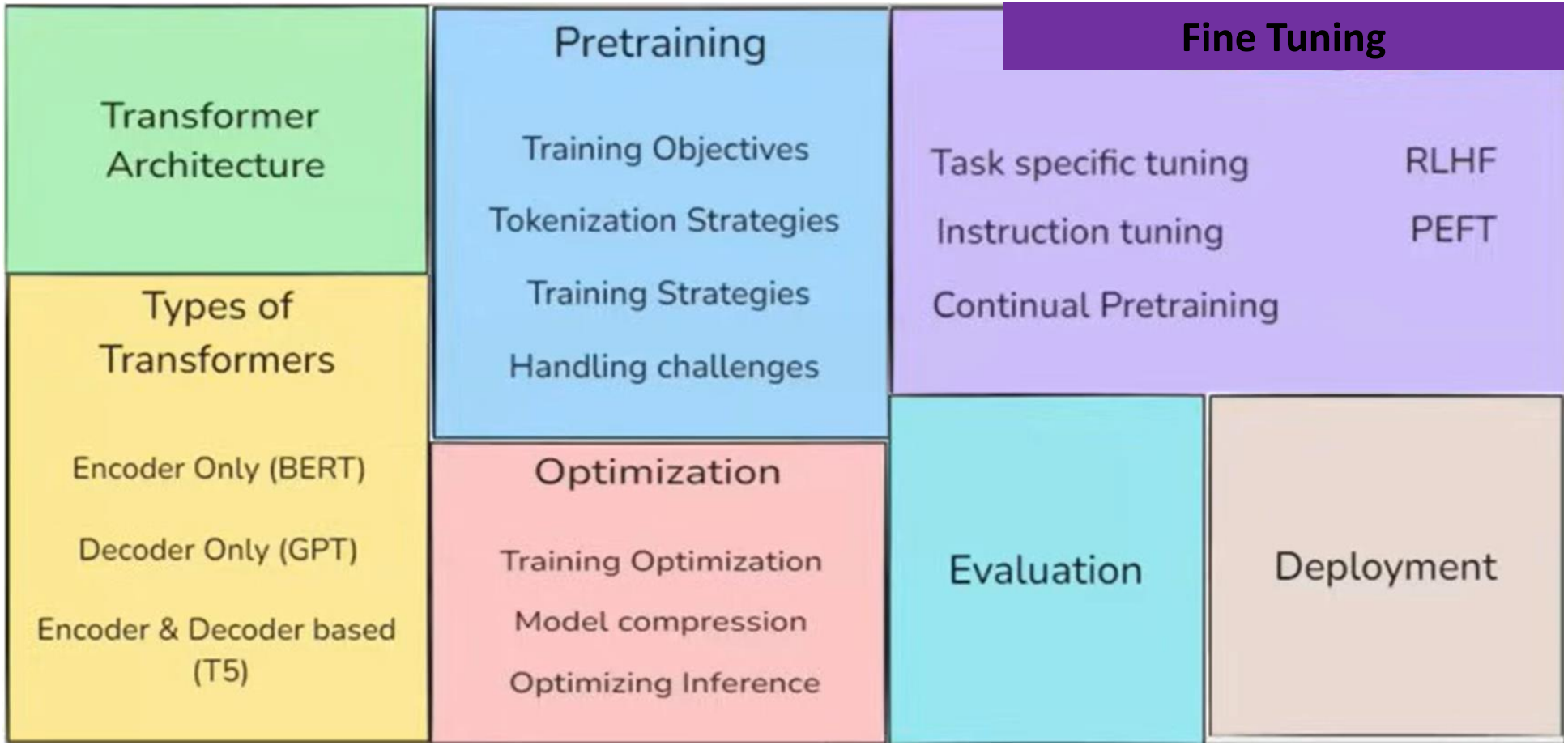
Is it accessible?

background

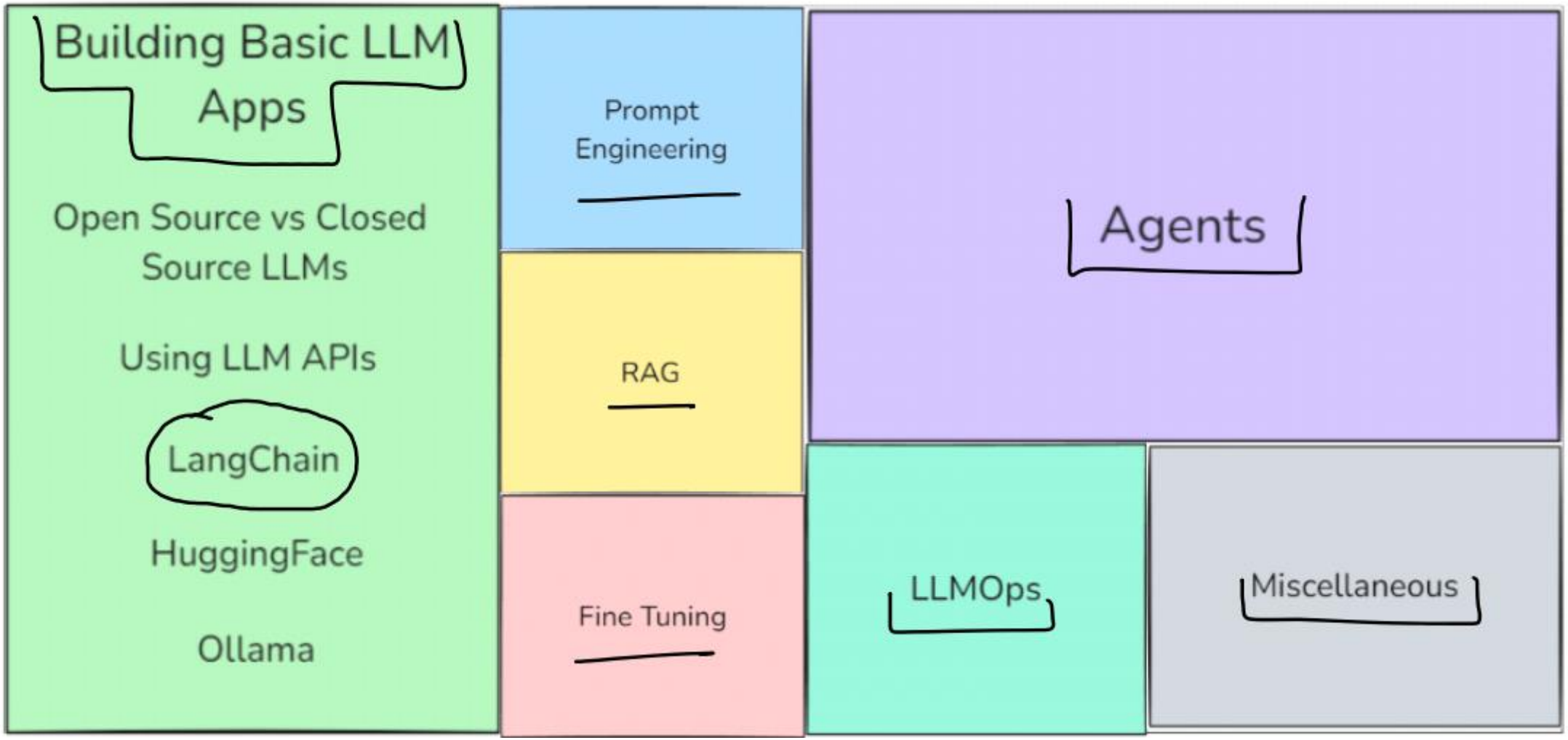



Prompt Engineering
RLHF
RAG
Pretraining
Quantization
AI Agents
Vector Databases
Fine Tuning
User Perspective
Builder Perspective

Builder Perspective



User Perspective

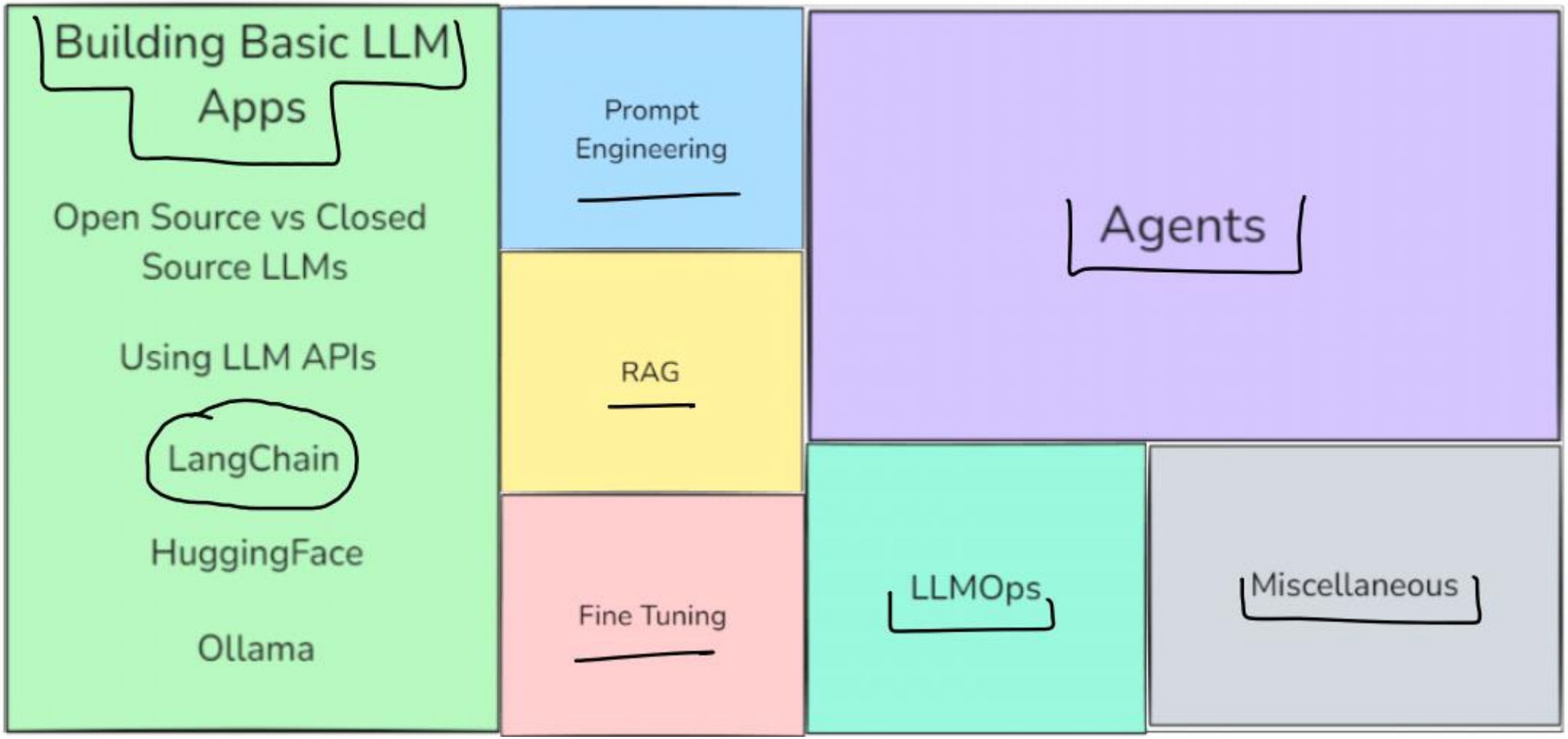


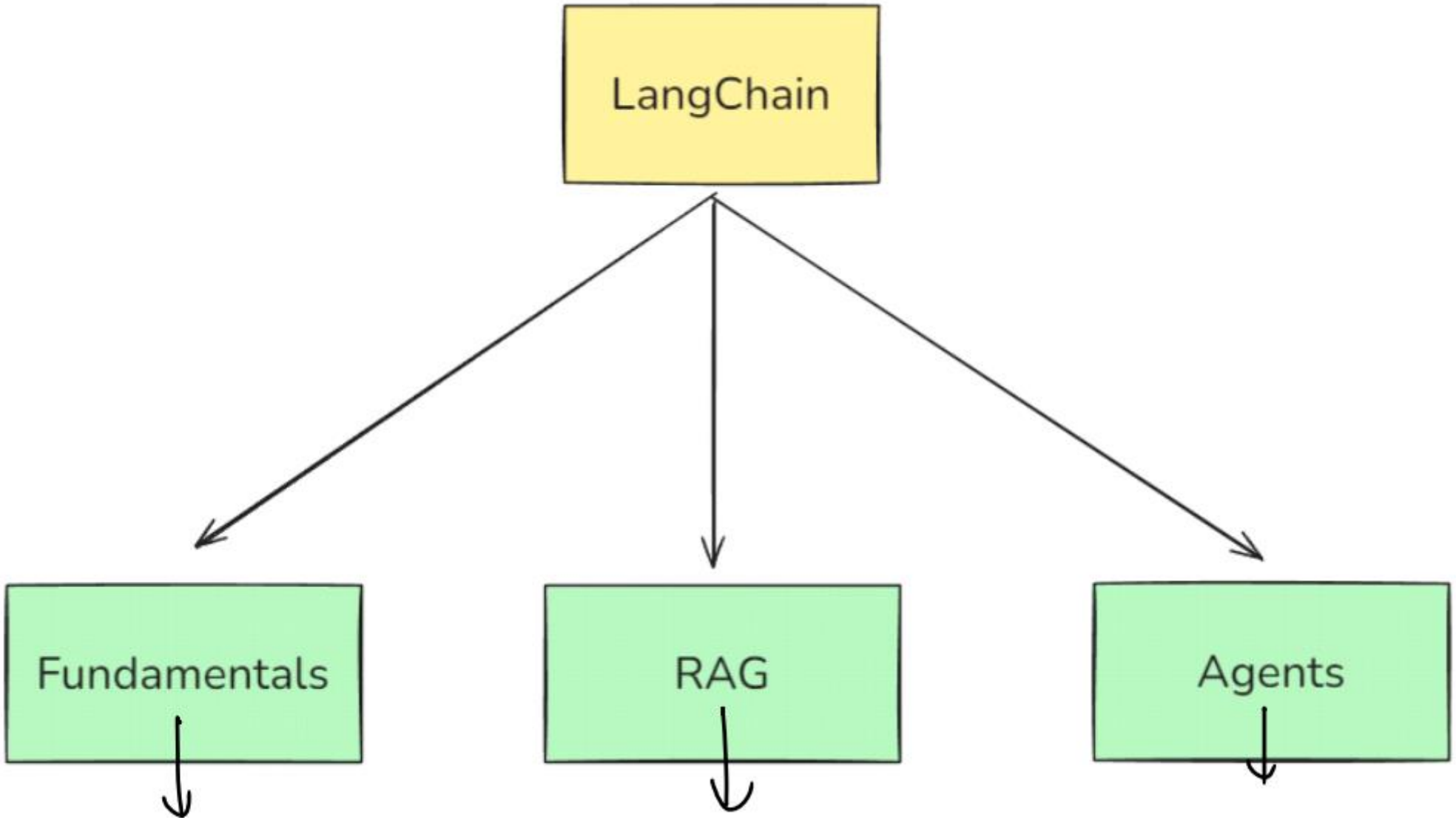
What is Langchain?

LangChain is an open source framework that helps in building LLM based applications. It provides modular components and end-to-end tools that help developers build complex AI applications, such as chatbots, question-answering systems, retrieval-augmented generation (RAG), autonomous agents, and more

1. Supports all the major LLMs
2. Simplifies developing LLM based applications
3. Integrations available for all major tools
4. Open source/Free/Actively developed
5. Supports all major GenAI use cases

Why Langchain first



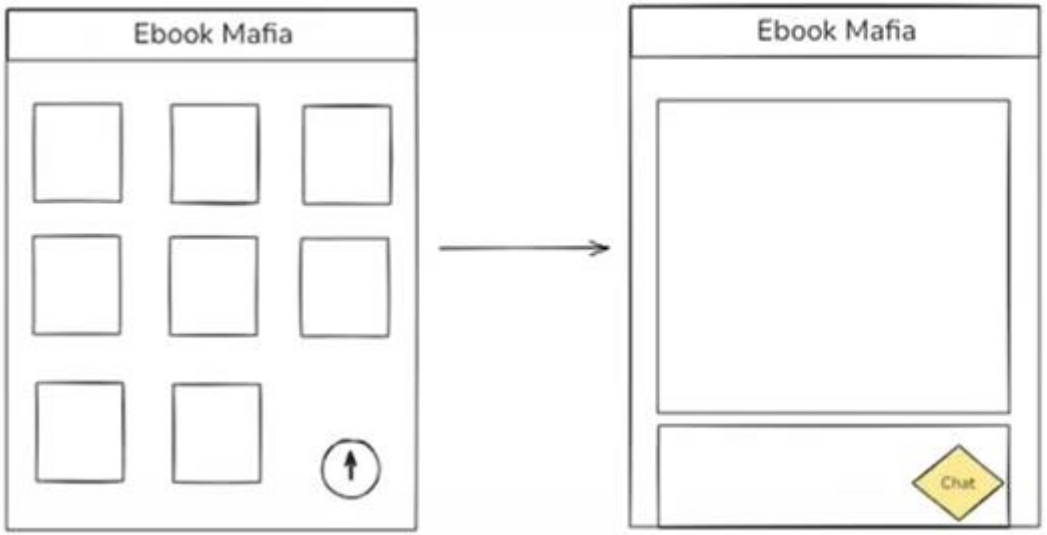

LangChain
Fundamentals
RAG
Agents

1. Updated information
2. Clarity
3. Conceptual understanding
4. The 80 percent approach

What is LangChain

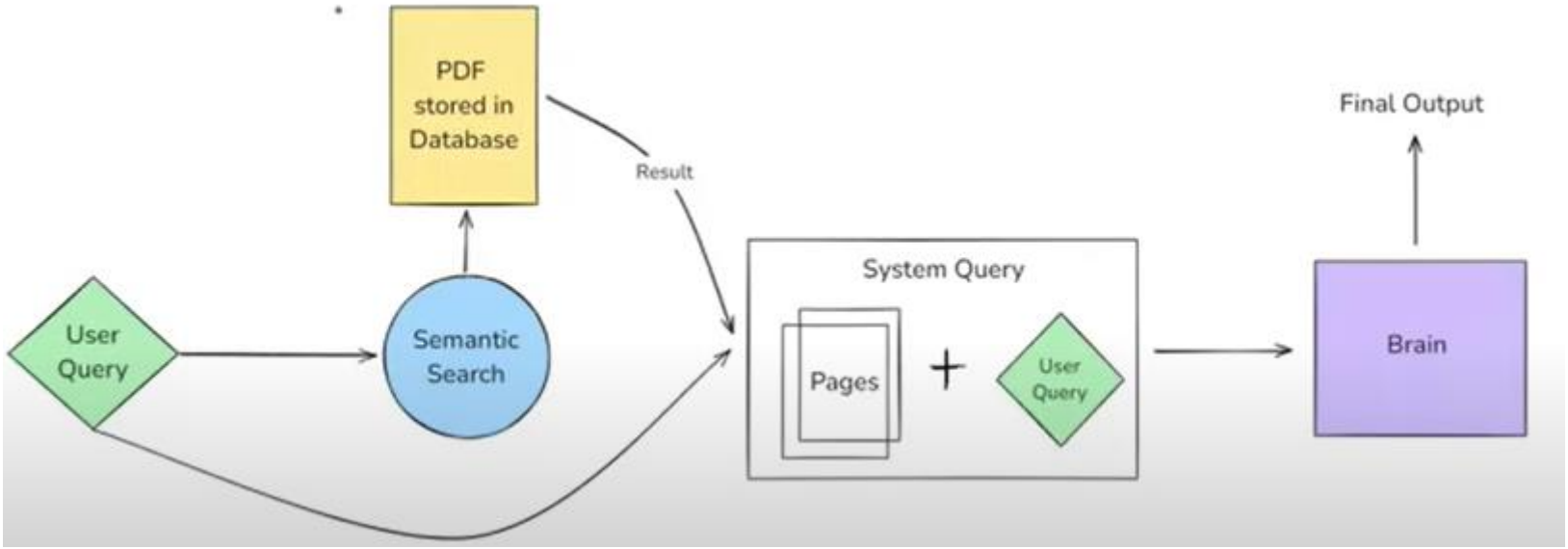
- LangChain is an open-source framework for developing applications powered by large language models (LLMs).

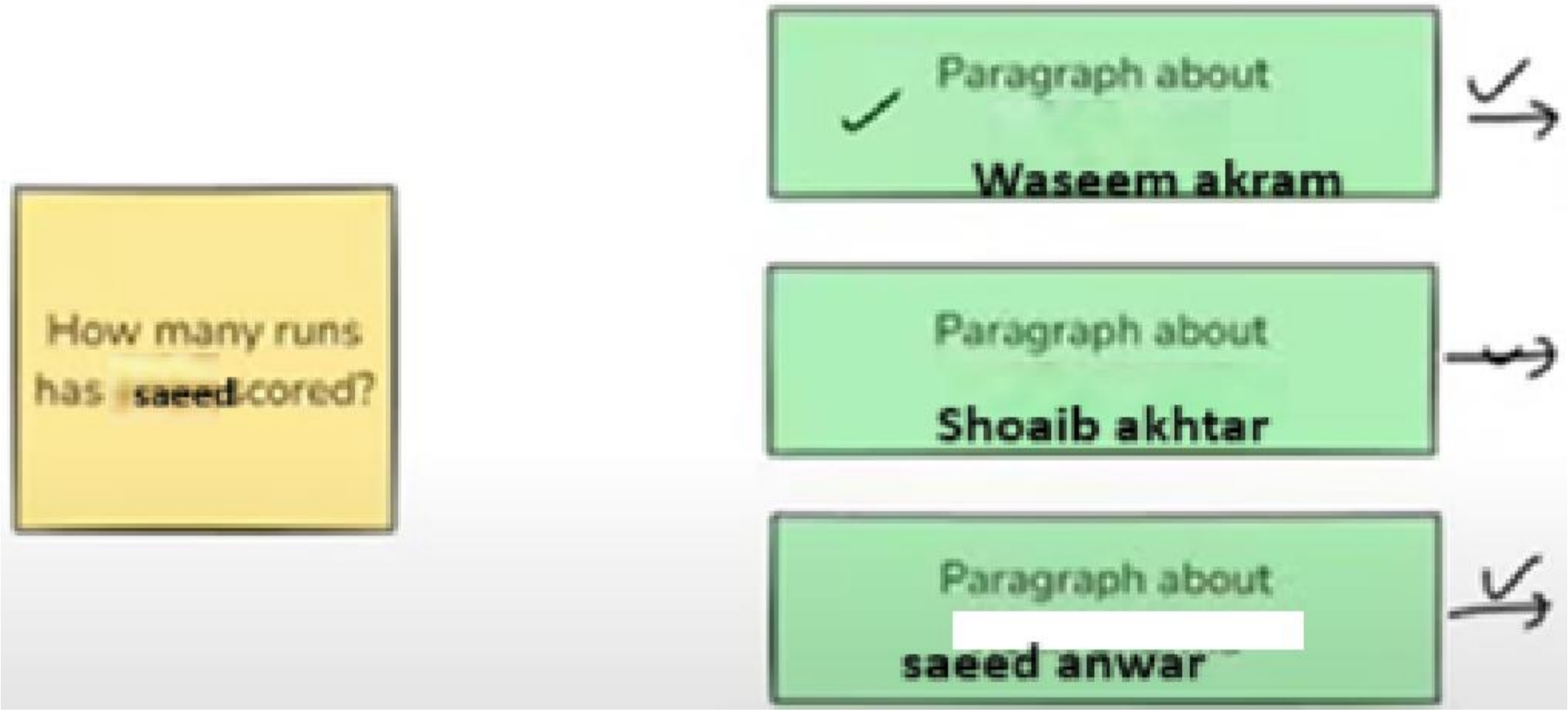
Why we Need LangChain?

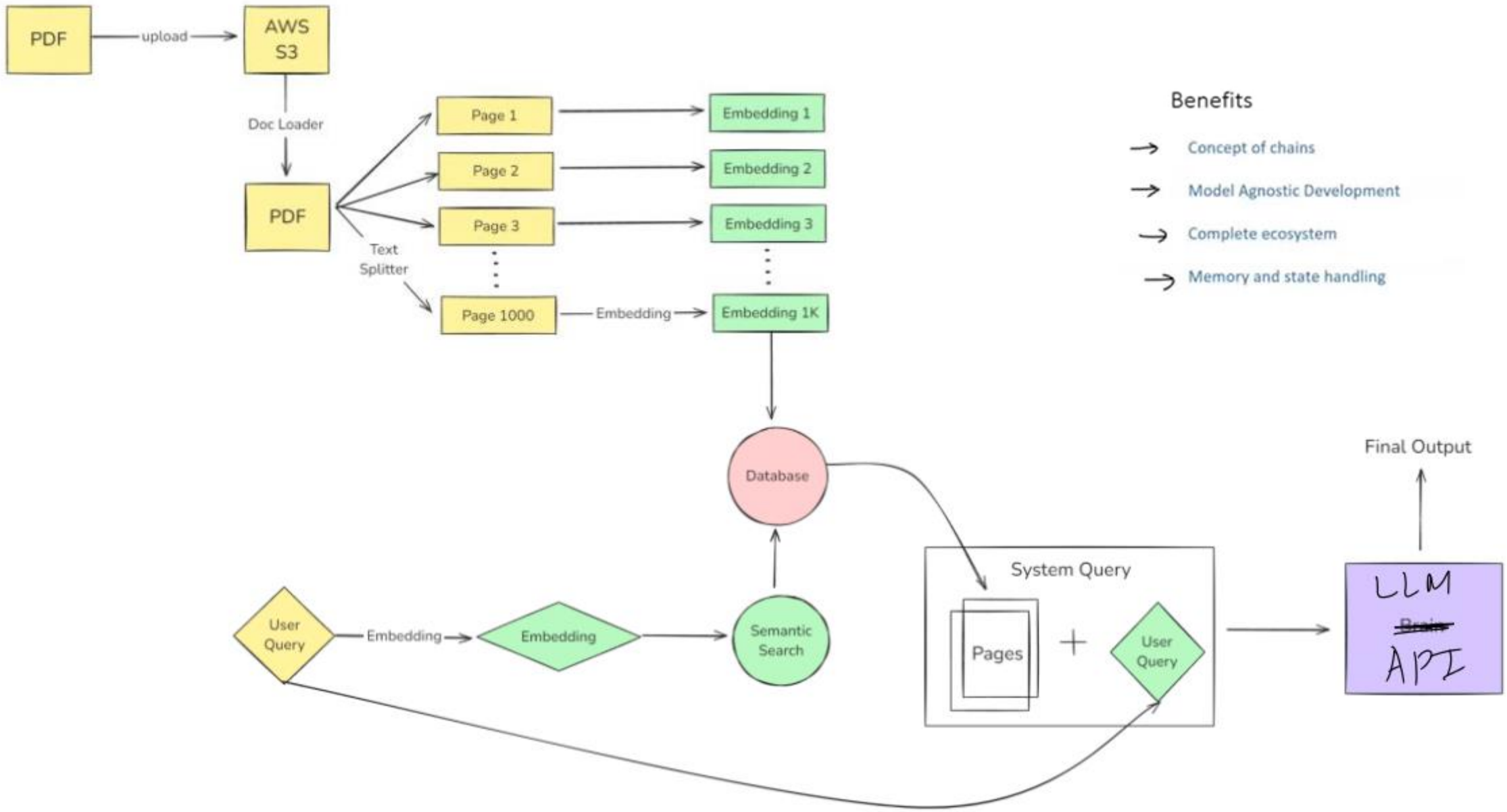


Example Queries

1. Explain page number 5 as if I am a 5 year old
2. Generate a True False exercise on Linear Regression
3. Generate notes for Decision Trees




How many runs has saeed cored?
Paragraph about
Waseem akram
Paragraph about
Shoaib akhtar
Paragraph about
saeed anwar


PDF
upload
AWS S3
Doc Loader
PDF
Text Splitter
Page 1
Page 2
Page 3
Page 1000
Embedding
Embedding 1
Embedding 2
Embedding 3
Embedding 1K
Database
User Query
Embedding
Embedding
Semantic Search
System Query
Pages
User Query
LLM
API
Final Output
Benefits
Concept of chains
Model Agnostic Development
Complete ecosystem
Memory and state handling

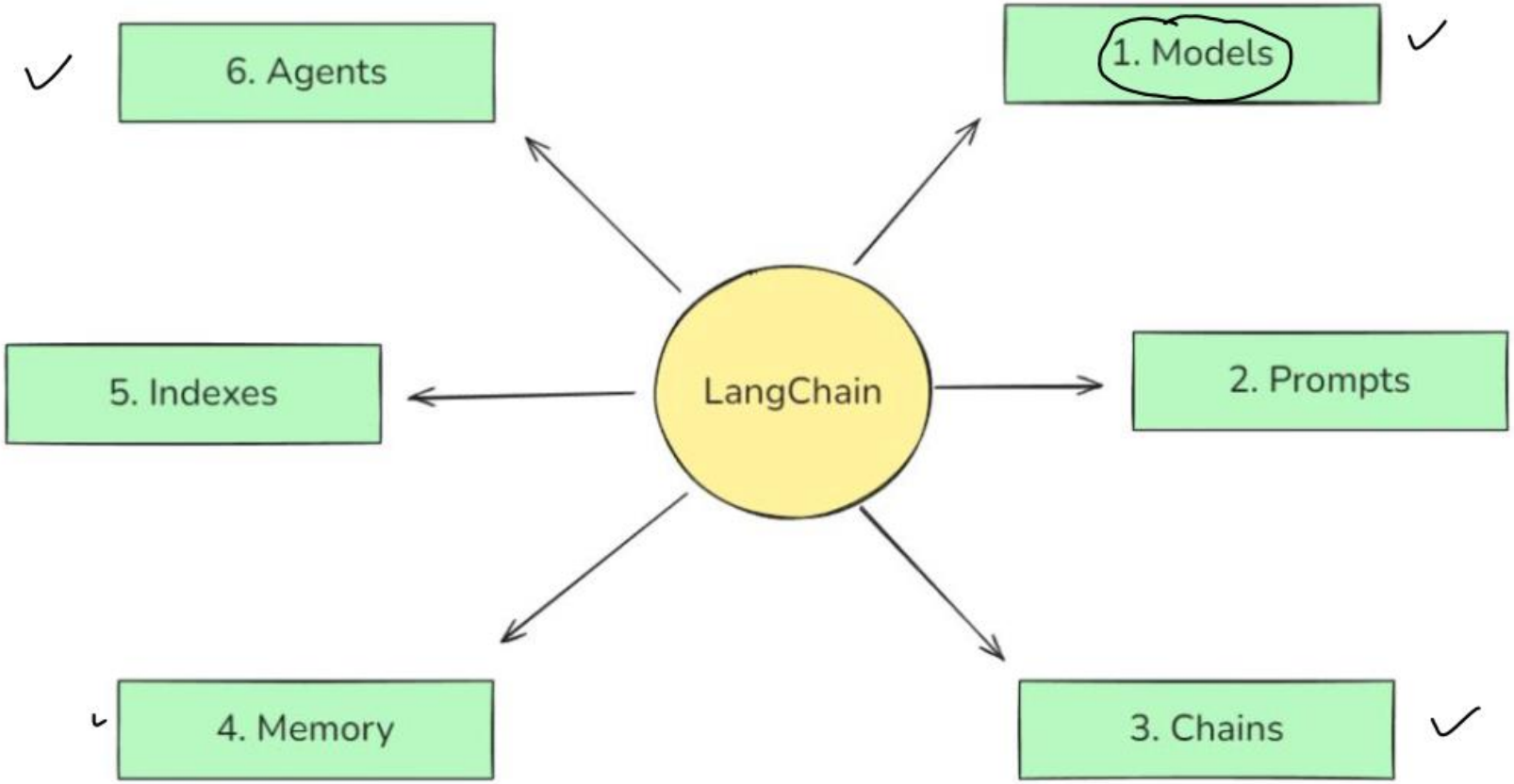
What can we build?

1. Conversational Chatbots
2. AI Knowledge Assistants
3. AI Agents
4. Workflow Automation
5. Summarization/Research Helpers

Alternatives

- LlamaIndex
- Haystack

Langchain Components



Model

- In LangChain, “models” are the core interfaces through which you interact with AI models.

Create a human-like response to a prompt

```python
from openai import OpenAI
client = OpenAI()

completion = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {
            "role": "user",
            "content": "Write a haiku about recursion in programming."
        }
    ]
)

print(completion.choices[0].message)
```

claude_quickstart.py

```python
import anthropic

client = anthropic.Anthropic()

message = client.messages.create(
    model="claude-3-5-sonnet-20241022",
    max_tokens=1000,
    temperature=0,
    system="You are a world-class poet. Respond only with short poems.",
    messages=[
        {
            "role": "user",
            "content": [
                {
                    "type": "text",
                    "text": "Why is the ocean salty?"
                }
            ]
        }
    ]
)
print(message.content)
```

```python
from langchain_openai import ChatOpenAI
from dotenv import load_dotenv

load_dotenv()

model = ChatOpenAI(model='gpt-4', temperature=0)

result = model.invoke("Now divide the result by 1.5")

print(result.content)
```

```python
from langchain_anthropic import ChatAnthropic
from dotenv import load_dotenv

load_dotenv()

model = ChatAnthropic(model='claude-3-opus-20240229')

result = model.invoke("Hi who are you")

print(result.content)
```

Prompts

1. Dynamic & Reusable Prompts

```
from langchain_core.prompts import PromptTemplate

prompt = PromptTemplate.from_template('Summarize {topic} in {emotion} tone')

print(prompt.format(topic='Cricket', length='fun'))
```

2. Role-Based Prompts

```
# Define the ChatPromptTemplate using from_template
chat_prompt = ChatPromptTemplate.from_template([
    ("system", "Hi you are a experienced {profession}"),
    ("user", "Tell me about {topic}"),
])

# Format the prompt with the variable
formatted_messages = chat_prompt.format_messages(profession="Doctor", topic="Viral Fever")
```

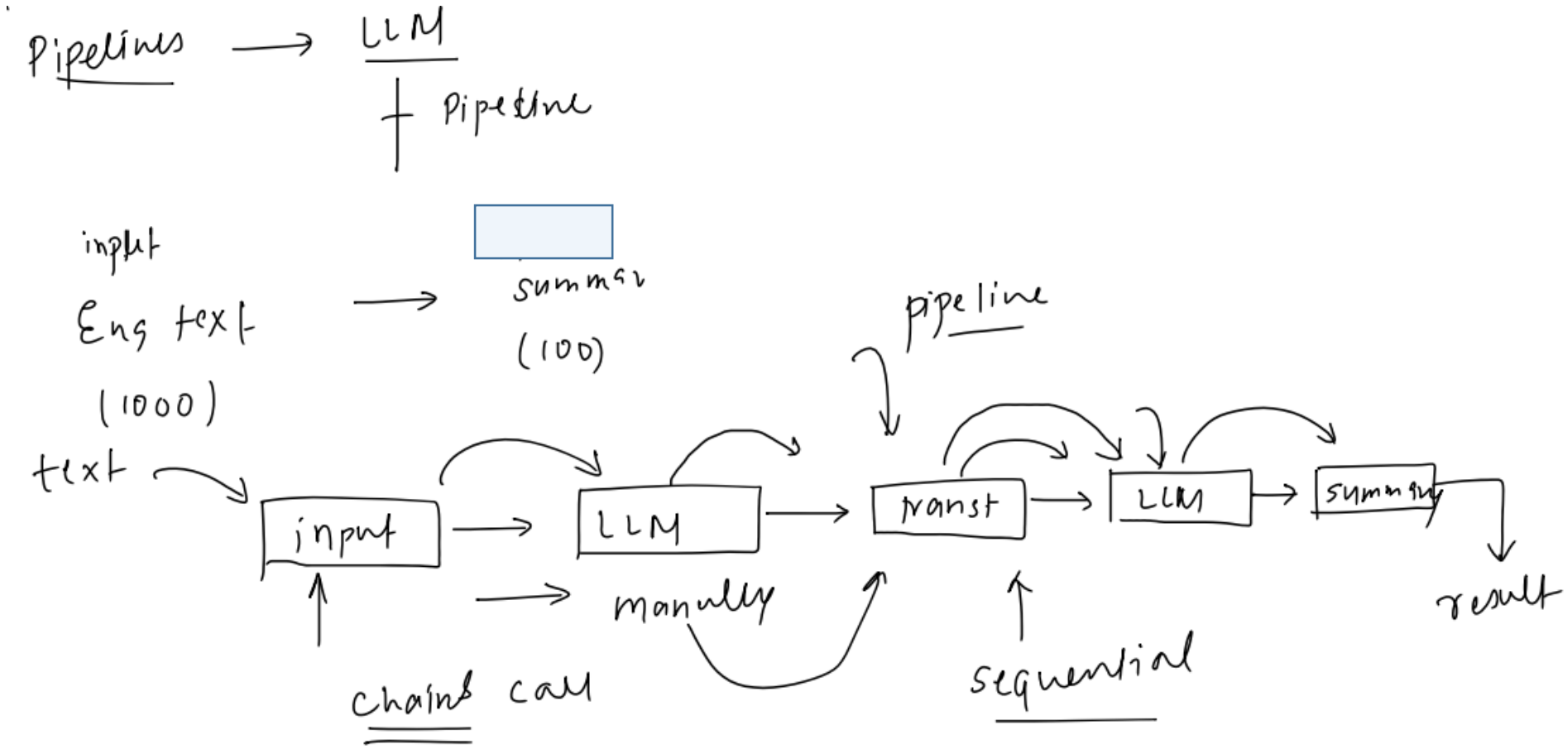
3. Few Shot Prompting

```
examples = [
    {"input": "I was charged twice for my subscription this month.", "output": "Billing Issue"},
    {"input": "The app crashes every time I try to log in.", "output": "Technical Problem"},
    {"input": "Can you explain how to upgrade my plan?", "output": "General Inquiry"},
    {"input": "I need a refund for a payment I didn't authorize.", "output": "Billing Issue"},
]
```

```
# Step 2: Create an example template
example_template = """
Ticket: {input}
Category: {output}
"""
```

```
# Step 3: Build the few-shot prompt template
few_shot_prompt = FewShotPromptTemplate(
    examples=examples,
    example_prompt=PromptTemplate(input_variables=["input", "output"], template=example_template),
    prefix="Classify the following customer support tickets into one of the categories: 'Billing
    Issue', 'Technical Problem', or 'General Inquiry'.\n\n",
    suffix="\nTicket: {user_input}\nCategory:",
    input_variables=["user_input"],
)
```

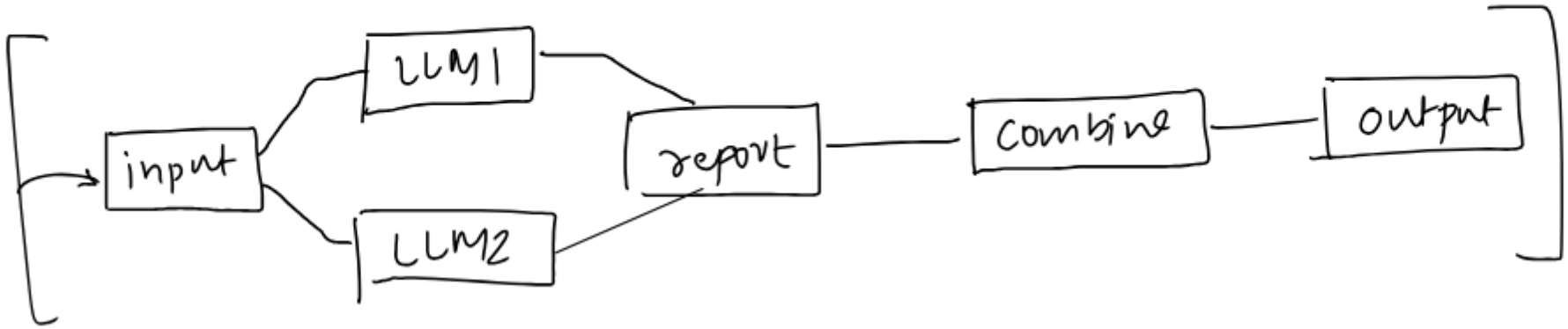
Chains

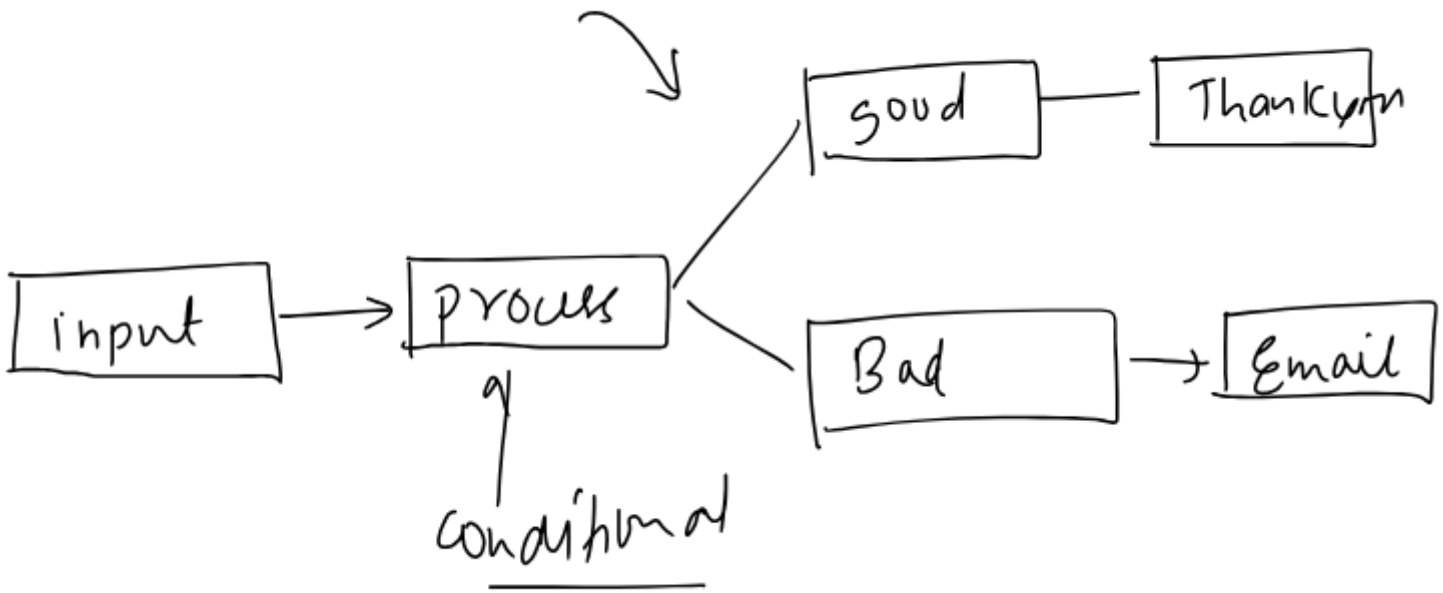

Pipelines
LLM
+ Pipeline
input
Eng text
(1000)
summary
(100)
text
input
LLM
manually
pipeline
transl
LLM
summary
result
chains call
sequential

Complex pipeline

Conditional chains

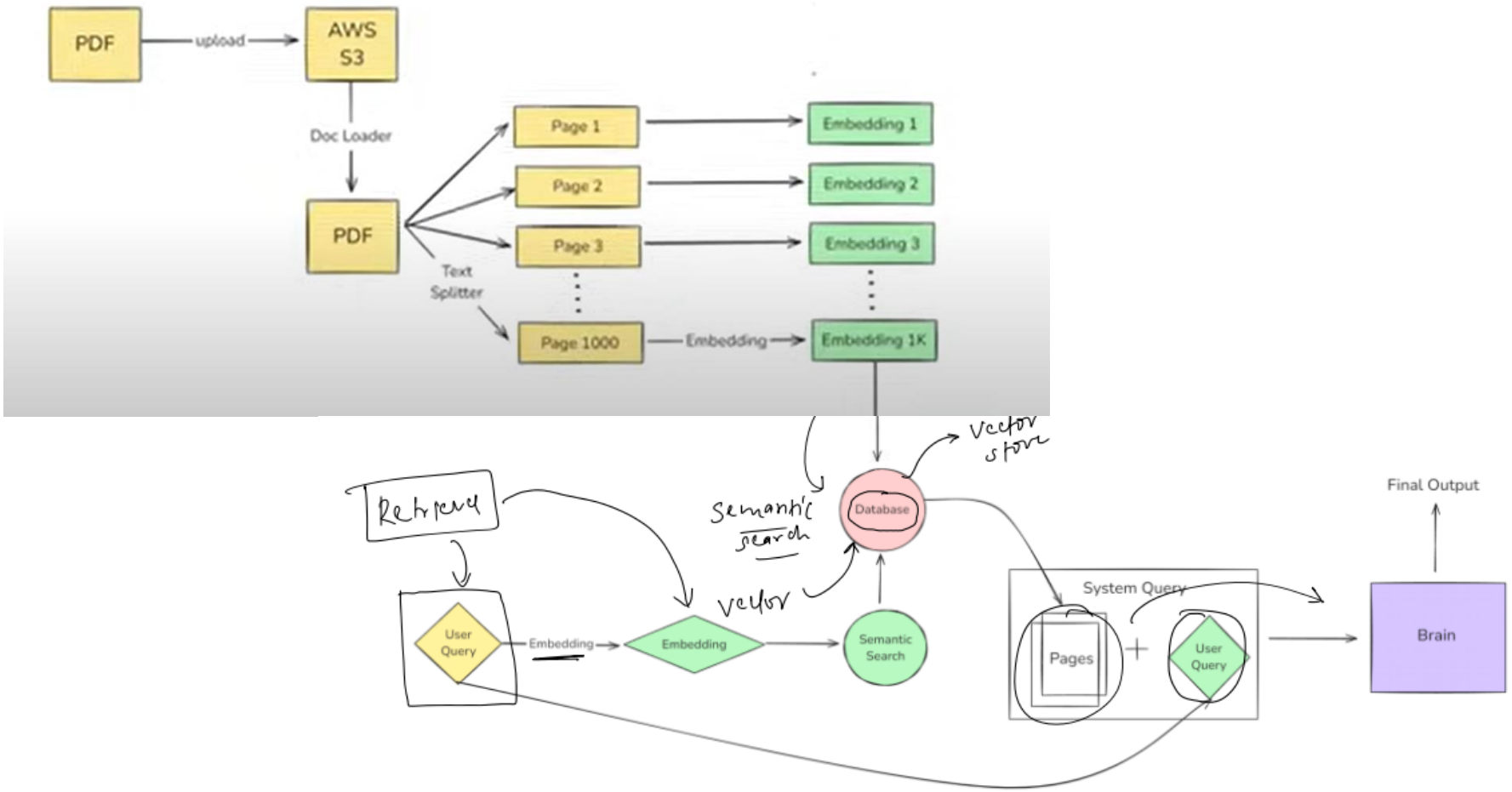
Parallel chain




input
process
conditional
good
Thankyou
Bad
Email

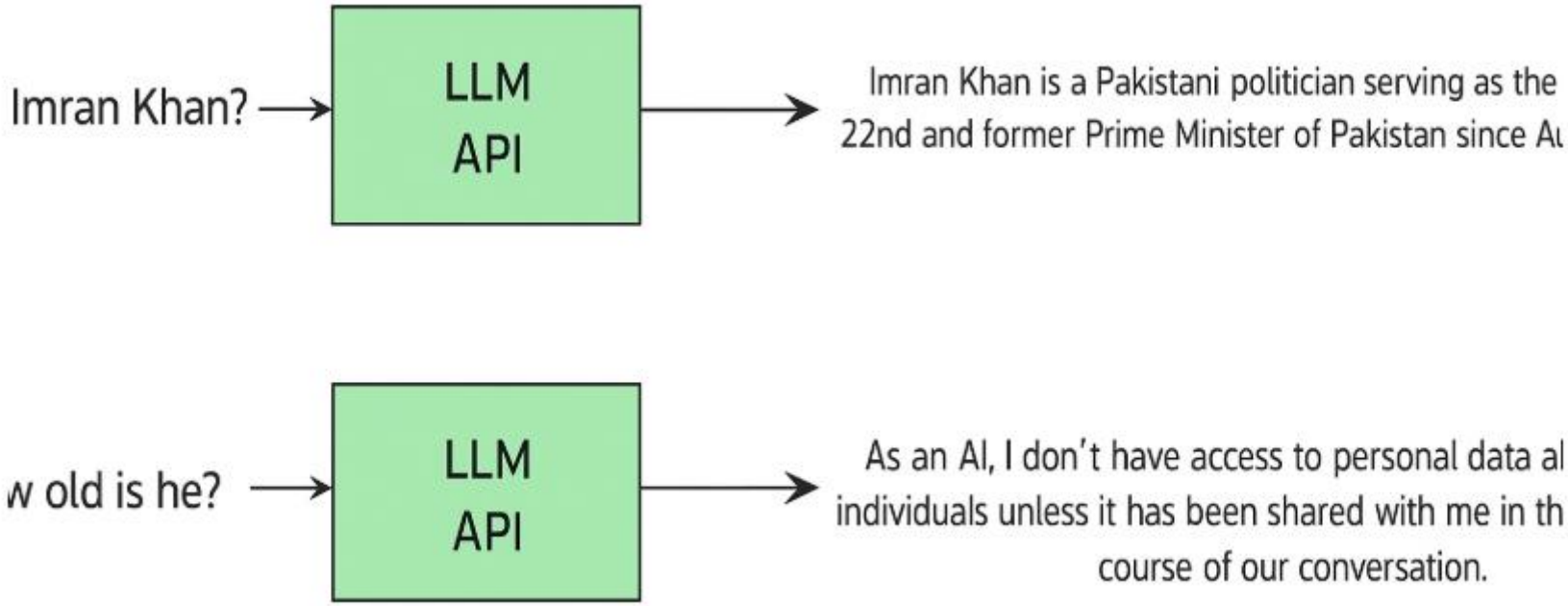
Indexes

- Indexes connect your application to external knowledge—such as PDFs, websites or databases



Memory

LLM API calls are stateless



- ConversationBufferMemory: Stores a transcript of recent messages. Great for short chats but can grow large quickly.
- ConversationBufferWindowMemory: Only keeps the last N interactions to avoid excessive token usage.
- Summarizer-Based Memory: Periodically summarizes older chat segments to keep a condensed memory footprint.
- Custom Memory: For advanced use cases, you can store specialized state (e.g., the user's preferences or key facts about them) in a custom memory class.

Agents

AI Agents

+ LLMs (NLU + Text gen) → chatbot

+

Chatbot

AI agent

↓

Chatbot with super

travel website

+ Shimla

+ Manali

24th Jan

API

Indigo

Book the flight

AI agent

+ [Reasoning capability]

+ Tools

Tools → Calculator, Weather API

↓

User → AI agent → Can you multi today's temp of Delhi with 3

+ Chain of Thought

→ [Delhi temp]

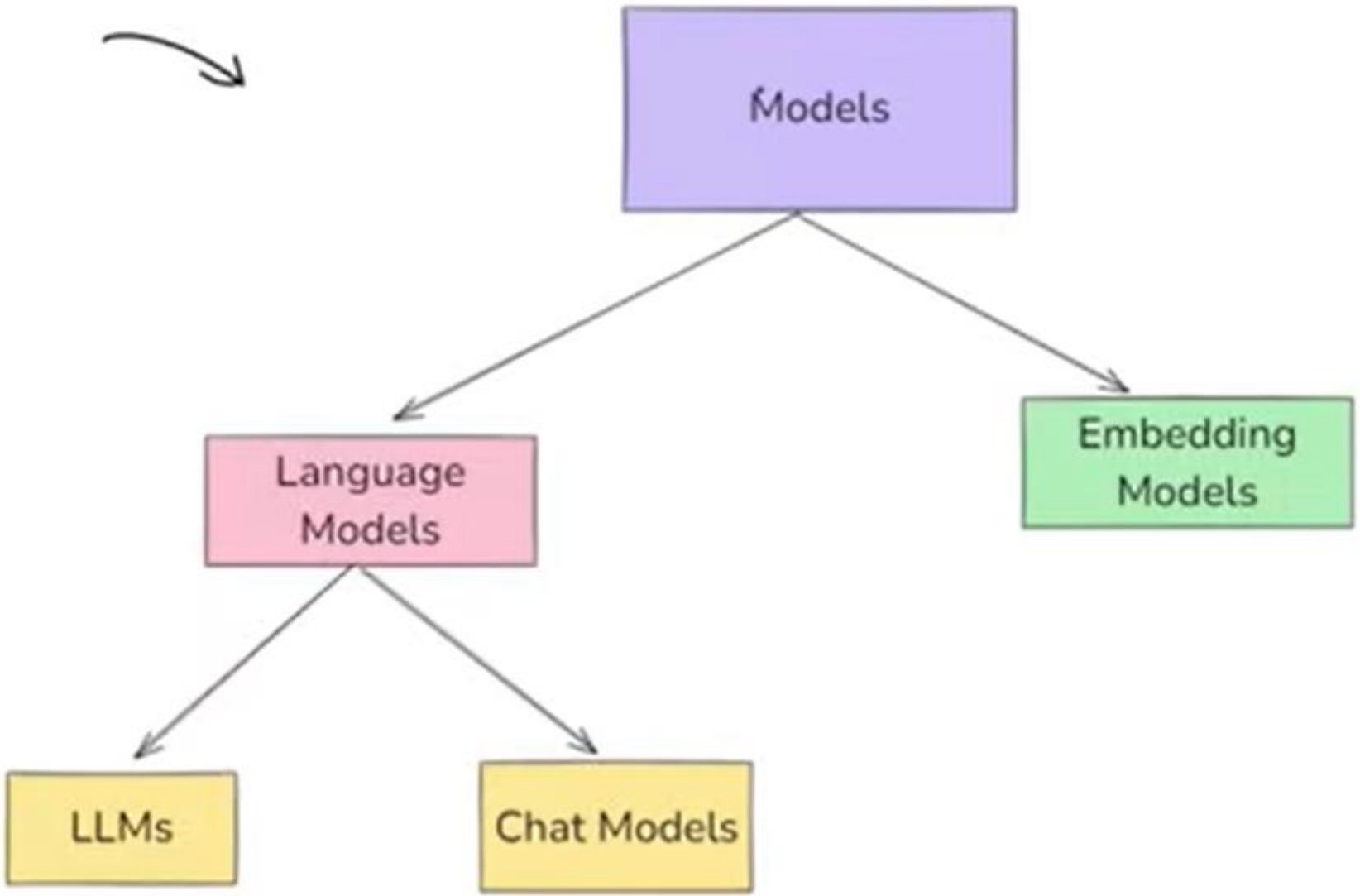
→ 25°C

→ 25, 3, *

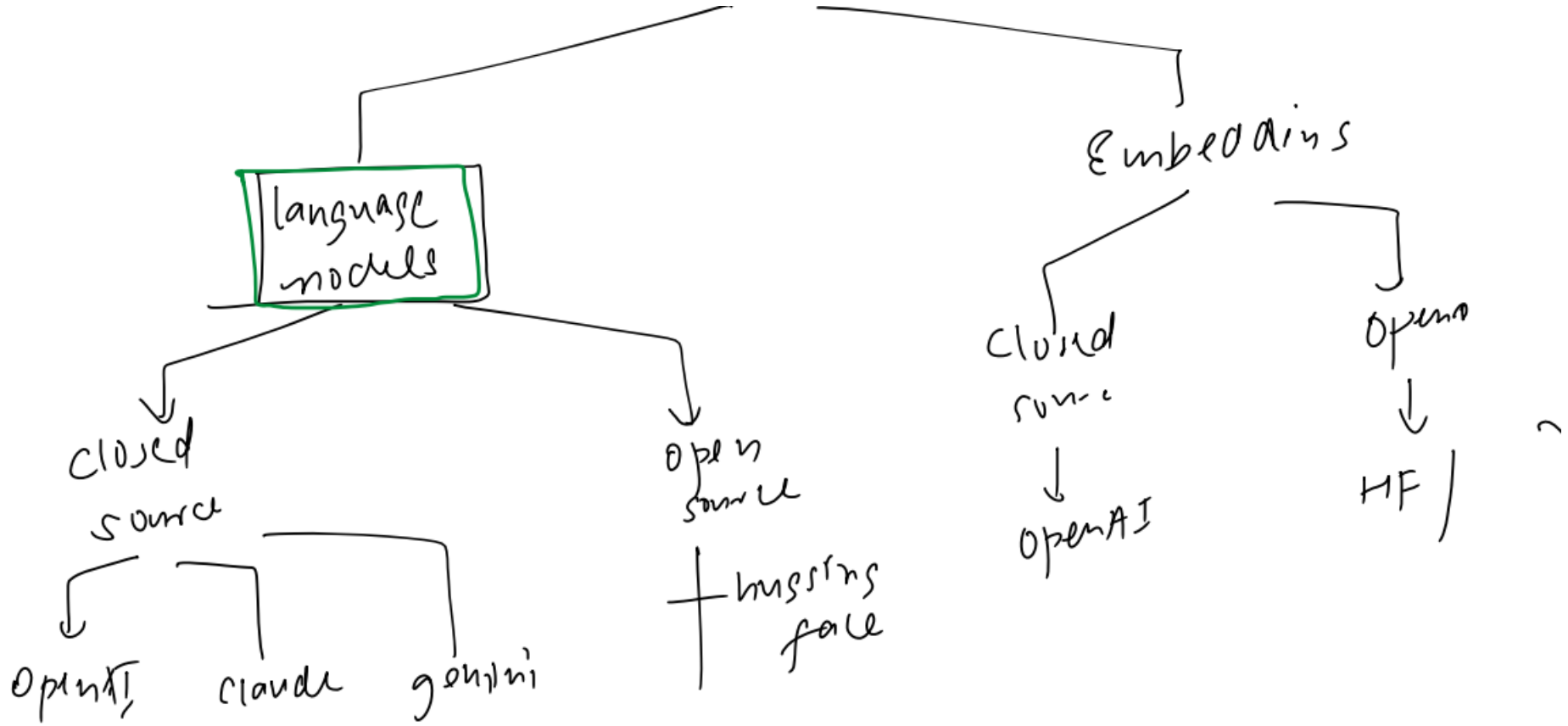
75 → 75

What are Models

- The Model Component in LangChain is a crucial part of the framework, designed to facilitate interactions with various language models and embedding models.
- It abstracts the complexity of working directly with different LLMs, chat models, and embedding models, providing a uniform interface to communicate with them. This makes it easier to build applications that rely on AI-generated text, text embeddings for similarity search, and retrieval-augmented generation (RAG)

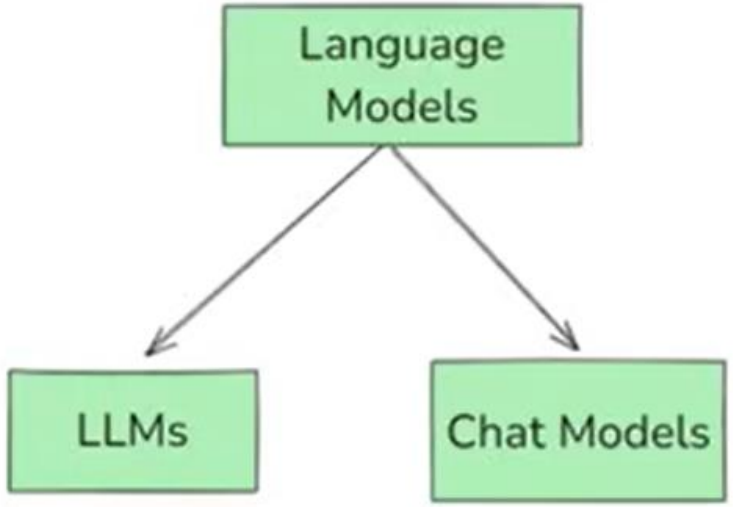

Models
Language Models
Embedding Models
LLMs
Chat Models

Plan of actions



Language Models

- Language Models are AI systems designed to process, generate, and understand natural language text.



- LLMs - General-purpose models that is used for raw text generation. They take a string(or plain text) as input and returns a string(plain text). These are traditionally older models and are not used much now.
- Chat Models - Language models that are specialized for conversational tasks. They take a sequence of messages as inputs and return chat messages as outputs (as opposed to using plain text). These are traditionally newer models and used more in comparison to the LLMs.

Feature	LLMs (Base Models)	Chat Models (Instruction-Tuned)
Purpose	Free-form text generation	Optimized for multi-turn conversations
Training Data	General text corpora (books, articles)	Fine-tuned on chat datasets (dialogues, user-assistant conversations)
Memory & Context	No built-in memory	Supports structured conversation history
Role Awareness	No understanding of "user" and "assistant" roles	Understands "system", "user", and "assistant" roles
Example Models	GPT-3, Llama-2-7B, Mistral-7B, OPT-1.3B	GPT-4, GPT-3.5-turbo, Llama-2-Chat, Mistral-Instruct, Claude
Use Cases	Text generation, summarization, translation, creative writing, code generation	Conversational AI, chatbots, virtual assistants, customer support, AI tutors

code

`temperature` is a parameter that **controls the randomness of a language model's output**. It affects how **creative or deterministic** the responses are.

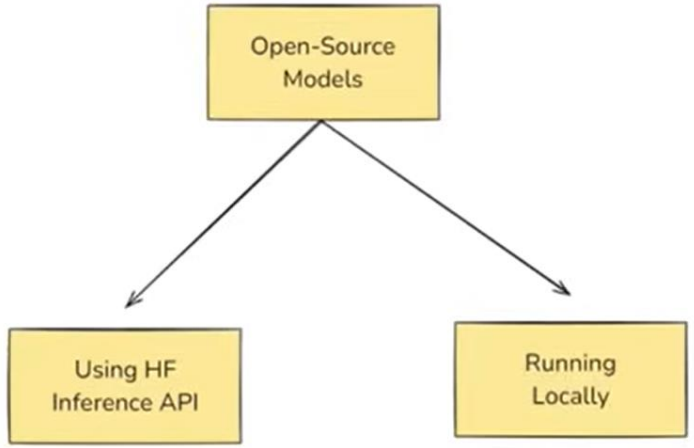
- **Lower values** (`0.0 - 0.3`) → More **deterministic** and predictable.
- **Higher values** (`0.7 - 1.5`) → More **random**, creative, and diverse.

Use Case	Recommended Temperature
Factual answers (math, code, facts)	`0.0 - 0.3`
Balanced response (general QA, explanations)	`0.5 - 0.7`
Creative writing, storytelling, jokes	`0.9 - 1.2`
Maximum randomness (wild ideas, brainstorming)	`1.5+`

Feature	Open-Source Models	Closed-Source Models
Cost	Free to use (no API costs)	Paid API usage (e.g., OpenAI charges per token)
Control	Can modify, fine-tune, and deploy anywhere	Locked to provider's infrastructure
Data Privacy	Runs locally (no data sent to external servers)	Sends queries to provider's servers
Customization	Can fine-tune on specific datasets	No access to fine-tuning in most cases
Deployment	Can be deployed on **on-premise** servers or **cloud**	Must use vendor's API

Some Famous Open Source Models

Model	Developer	Parameters	Best Use Case
LLaMA-2-7B/13B/70B	Meta AI	7B - 70B	General-purpose text generation
Mixtral-8x7B	Mistral AI	8x7B (MoE)	Efficient & fast responses
Mistral-7B	Mistral AI	7B	Best small-scale model (outperforms LLaMA-2-13B)
Falcon-7B/40B	TII UAE	7B - 40B	High-speed inference
BLOOM-176B	BigScience	176B	Multilingual text generation
GPT-J-6B	EleutherAI	6B	Lightweight and efficient
GPT-NeoX-20B	EleutherAI	20B	Large-scale applications
StableLM	Stability AI	3B - 7B	Compact models for chatbots



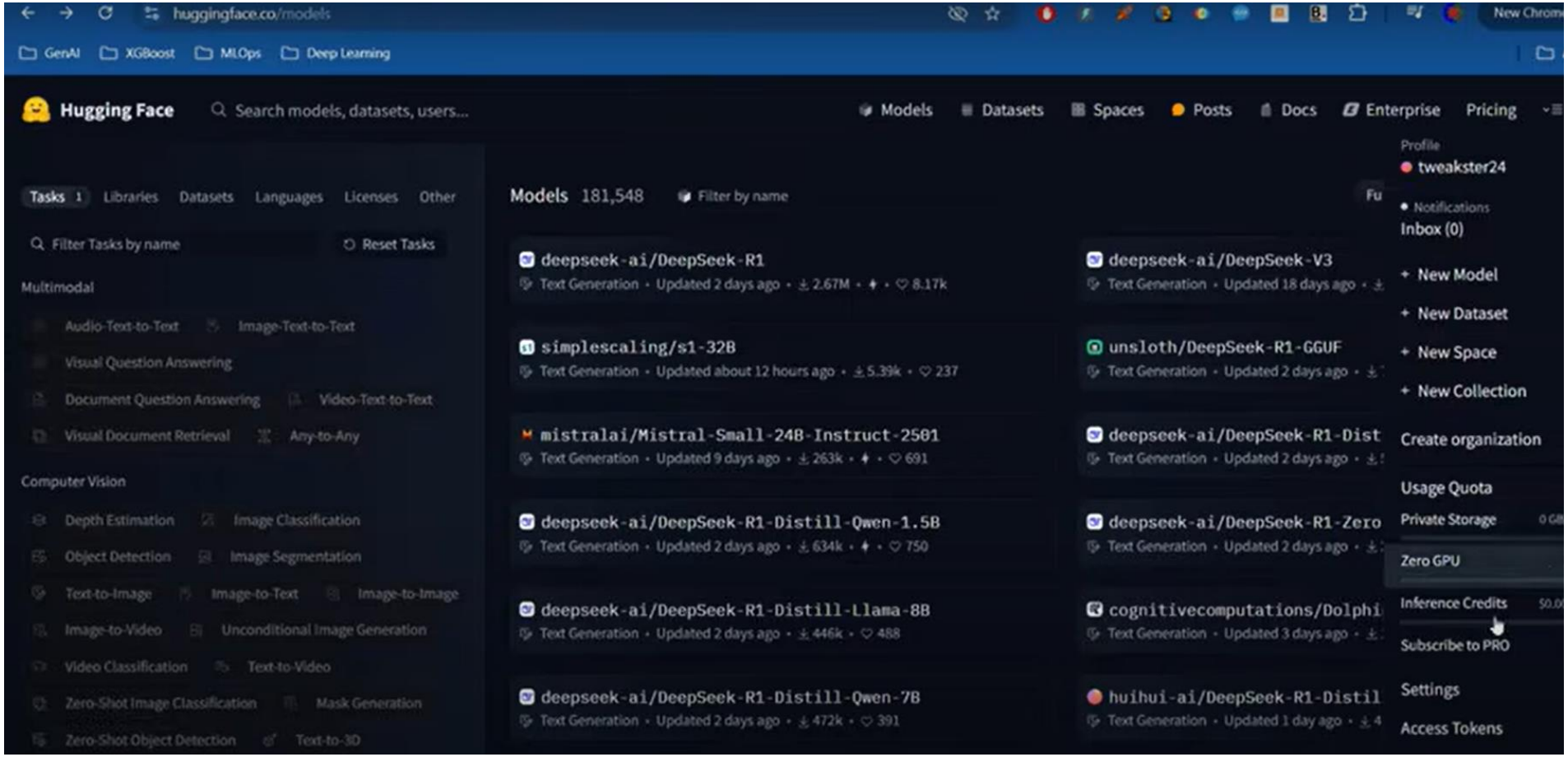
Disadvantages

Disadvantage	Details
High Hardware Requirements	Running large models (e.g., LLaMA-2-70B) requires expensive GPUs.
Setup Complexity	Requires installation of dependencies like **PyTorch, CUDA, transformers.**
Lack of RLHF	Most open-source models don't have **fine-tuning with human feedback**, making them weaker in instruction-following.
Limited Multimodal Abilities	Open models don't support **images, audio, or video** like GPT-4V.

Open Source Models

Open-source language models are freely available AI models that can be downloaded, modified, fine-tuned, and deployed without restrictions from a central provider. Unlike closed-source models such as OpenAI's GPT-4, Anthropic's Claude, or Google's Gemini, open-source models allow full control and customization.

Feature	Open-Source Models	Closed-Source Models
Cost	Free to use (no API costs)	Paid API usage (e.g., OpenAI charges per token)
Control	Can modify, fine-tune, and deploy anywhere	Locked to provider's infrastructure
Data Privacy	Runs locally (no data sent to external servers)	Sends queries to provider's servers
Customization	Can fine-tune on specific datasets	No access to fine-tuning in most cases
Deployment	Can be deployed on **on-premise** servers or **cloud**	Must use vendor's API


huggingface.co/models
GenAI
XGBoost
MLOps
Deep Learning
Hugging Face
Search models, datasets, users...
Models
Datasets
Spaces
Posts
Docs
Enterprise
Pricing
Tasks 1
Libraries
Datasets
Languages
Licenses
Other
Filter Tasks by name
Reset Tasks
Multimodal
Audio-Text-to-Text
Image-Text-to-Text
Visual Question Answering
Document Question Answering
Video-Text-to-Text
Visual Document Retrieval
Any-to-Any
Computer Vision
Depth Estimation
Image Classification
Object Detection
Image Segmentation
Text-to-Image
Image-to-Text
Image-to-Image
Image-to-Video
Unconditional Image Generation
Video Classification
Text-to-Video
Zero-Shot Image Classification
Mask Generation
Zero-Shot Object Detection
Text-to-3D
Models 181,548
Filter by name
deepseek-ai/DeepSeek-R1
Text Generation · Updated 2 days ago · 2.67M · 8.17k
deepseek-ai/DeepSeek-V3
Text Generation · Updated 18 days ago
simplescaling/s1-32B
Text Generation · Updated about 12 hours ago · 5.39k · 237
unsloth/DeepSeek-R1-GGUF
Text Generation · Updated 2 days ago
mistralai/Mistral-Small-24B-Instruct-2501
Text Generation · Updated 9 days ago · 263k · 691
deepseek-ai/DeepSeek-R1-Dist
Text Generation · Updated 2 days ago
deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B
Text Generation · Updated 2 days ago · 634k · 750
deepseek-ai/DeepSeek-R1-Zero
Text Generation · Updated 2 days ago
deepseek-ai/DeepSeek-R1-Distill-Llama-8B
Text Generation · Updated 2 days ago · 446k · 488
cognitivecomputations/Dolphi
Text Generation · Updated 3 days ago
deepseek-ai/DeepSeek-R1-Distill-Qwen-7B
Text Generation · Updated 2 days ago · 472k · 391
huihui-ai/DeepSeek-R1-Distil
Text Generation · Updated 1 day ago
Profile
tweakster24
Notifications
Inbox (0)
New Model
New Dataset
New Space
New Collection
Create organization
Usage Quota
Private Storage
Zero GPU
Inference Credits
Subscribe to PRO
Settings
Access Tokens

Embedding Models