# **What is LangChain?**

LangChain is a framework that helps developers build applications using large language models (LLMs) like GPT. It makes it easy to connect LLMs with tools, data sources, and memory for smarter and more useful AI apps.

# **Why Do We Need LangChain?**

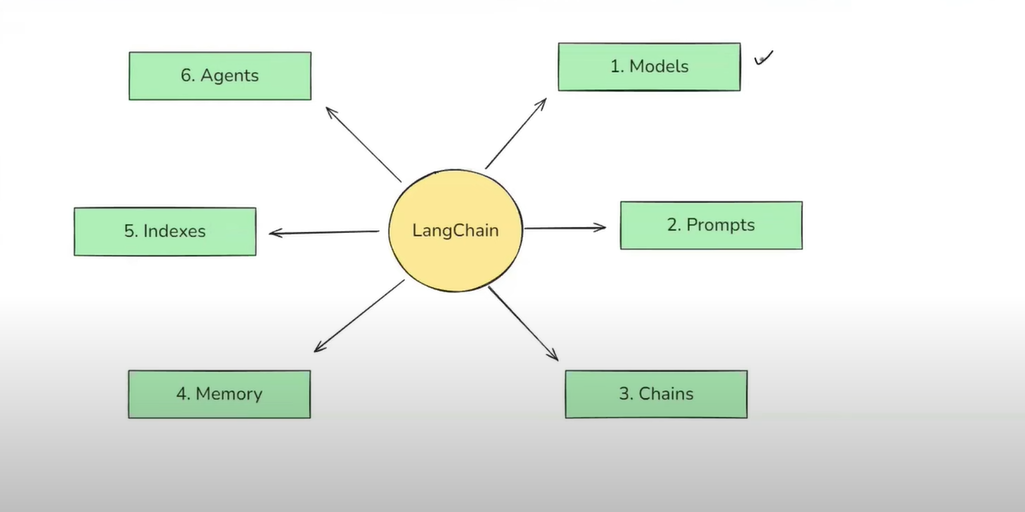
LLMs are powerful but limited — they can't remember past chats, use tools, or access external data on their own. LangChain solves this by:

* Adding memory to conversations
* Connecting LLMs to tools and APIs
* Allowing multi-step reasoning (chains)

It’s great for building chatbots, document search tools, and AI agents.

# **Generative AI Using LangChain:**

LangChain consists of six core components, which we will explore step by step in detailed notes.  
The source code and methodology for practical implementation are available on GitHub.

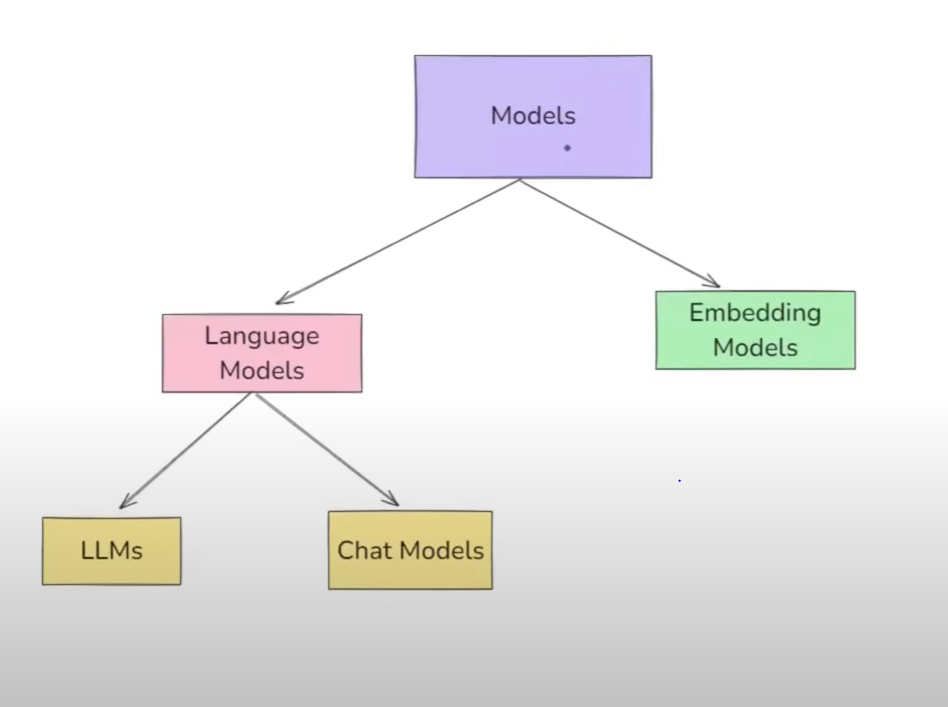


# **Models:**

The Model Component in LangChain is a crucial part of the framework, designed to facilitate interactions with various language models and embedding models.

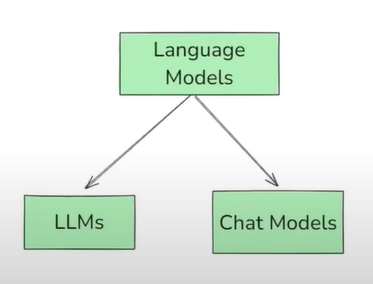
It abstracts the complexity of working directly with different LLMs, chat models, and embedding models, providing a uniform interface to communicate with them. This makes it easier to build applications that rely on AI-generated text, text embeddings for similarity search, and retrieval-augmented generation (RAG).

Embeddings models help in semantic search to match similarity.



# **Language Models**

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



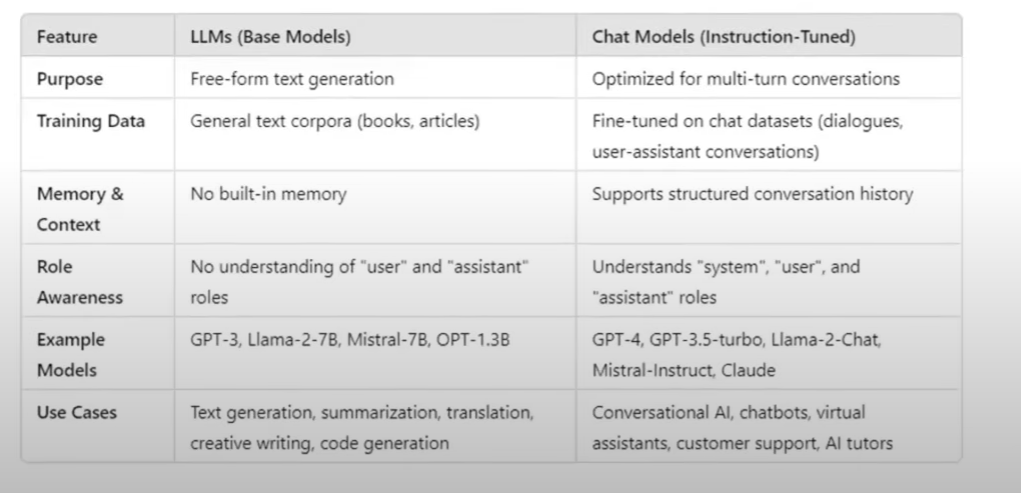
**LLMs:**

These are general-purpose models designed for raw text generation. They take plain text as input and return plain text as output. The text indicates these are traditionally older models and are not as widely used now.

**Chat Models:**

These are specialized language models built for conversational tasks. They take a sequence of messages as input and return chat messages as output, differentiating them from LLMs which use plain text. These are described as newer models and are more frequently used compared to LLMs.

**LLM vs Chat Model: Key Differences**



**Temperature:**

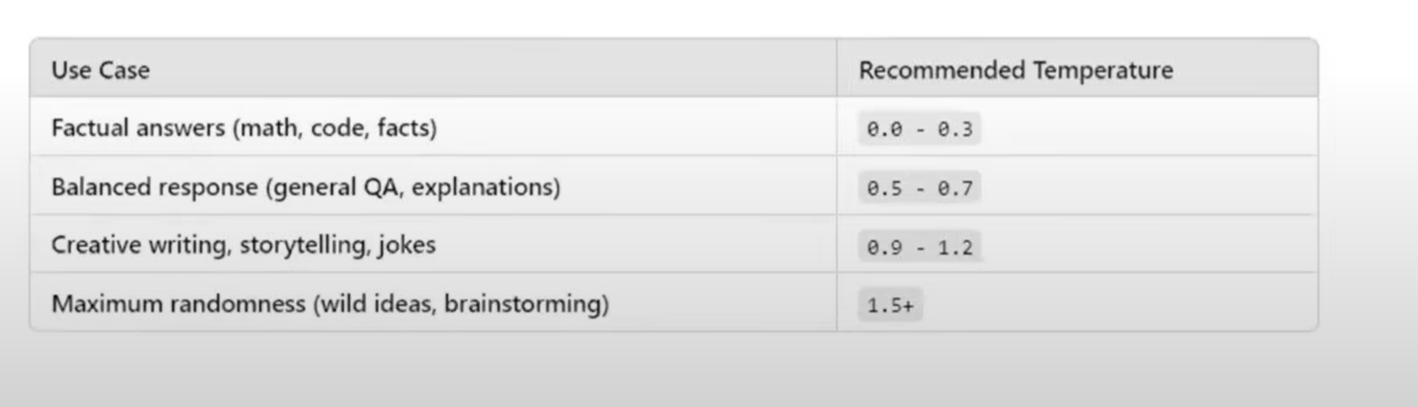
The statement describes how the 'temperature' parameter influences the output of a language model. It affects the creativity or determinism of the responses.

**Lower values (0.0 - 0.3):**

Result in more deterministic and predictable responses.

**Higher values (0.7 - 1.5):**

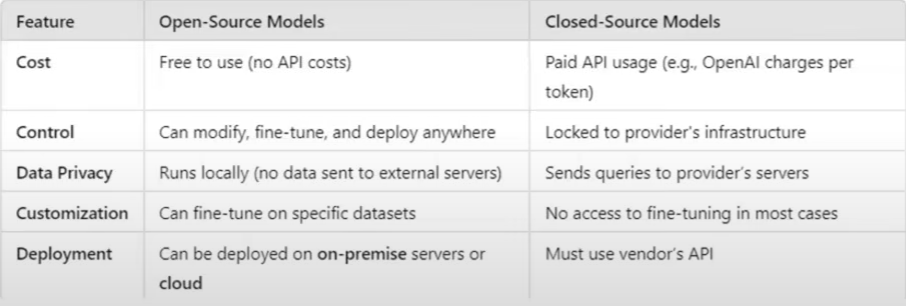
Lead to more random, creative, and diverse responses.



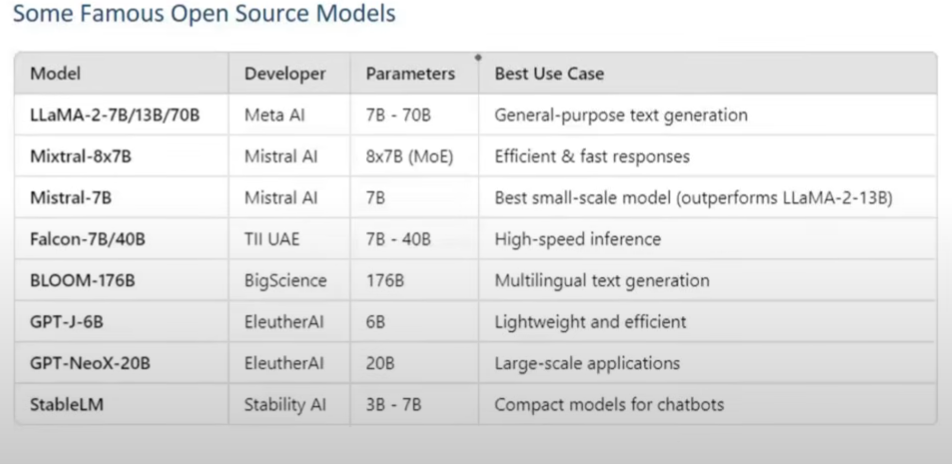
**Open Source Models**

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.

**Open Source Models** **Vs Closed source models**

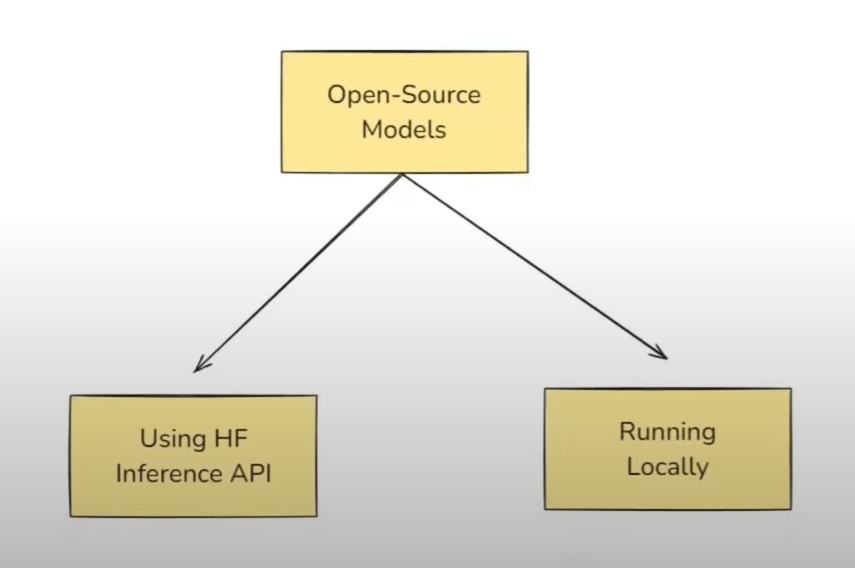


**Some Famous Open Source Models**

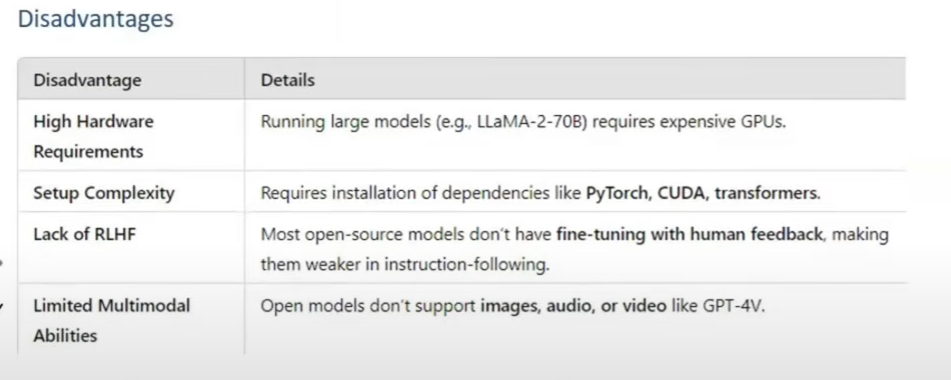


**Where to Find Them?**

You can find open-source models on platforms like **Hugging Face**, which provides several ways to use and integrate these models into your applications.



**Disadvatages:**



# **Prompts**

Prompts are the input instructions or queries given to a model to guide its output.

**Types of Prompts:**

Text Based **=> Main Focused in these days.**

Image Based **=> Multi Model Based.**

**Static Vs Dynamic Prompts:**

**Static Prompts**

**Explanation:** These are fixed, pre-defined instructions that do not change during an interaction. They are independent of the model's previous responses.

**Example:**

* "Tell me a fact about the moon."
* "List three benefits of exercise."

**Dynamic Prompts**

**Explanation:** These prompts adapt and evolve based on prior interactions, model outputs, or external information. They allow for iterative refinement and more complex conversations.

**Example:**

* (After model lists moon facts) "Now, elaborate on the moon's atmosphere from your previous answer."
* (After model lists exercise benefits) "Suggest a beginner-friendly workout routine that incorporates those benefits."

**Prompt Template:**

A PromptTemplate in LangChain is a structured way to create prompts dynamically by inserting variables into a predefined template. Instead of hardcoding prompts, PromptTemplate allows you to define placeholders that can be filled in at runtime with different inputs.

This makes it reusable, flexible, and easy to manage, especially when working with dynamic user inputs or automated workflows.

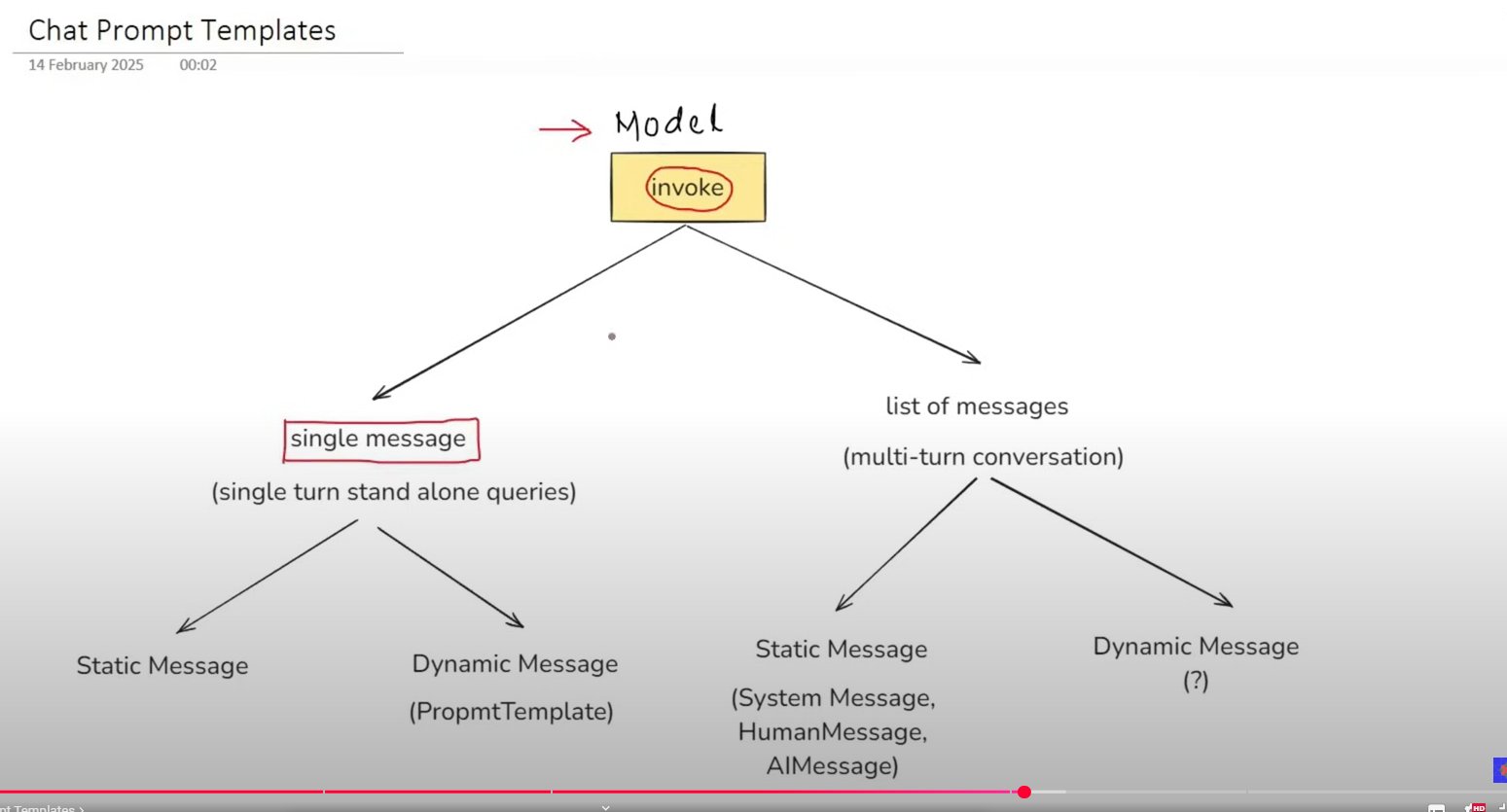
Why use PromptTemplate over f strings?

1. Default validation

2. Reusable

3. LangChain Ecosystem

**Chat PromptTemplate:**



**Message Placeholder**

A MessagesPlaceholder in LangChain is a special placeholder used inside a ChatPromptTemplate to dynamically insert chat history or a list of messages at runtime.