What is LangFlow?

LangFlow is an open source low-code tool for building AI agents and other AI applications through a drag-and-drop visual interface. It allows users to orchestrate large language models (LLMs), APIs, vector databases and custom components into agentic workflows without requiring advanced programming skills.

Your subscription will be delivered in English. You will find an unsubscribe link in every newsletter. You can manage your subscriptions or unsubscribe here. Refer to our IBM Privacy Statement for more information.

What is LangFlow used for?

LangFlow is used to create agentic AI applications within a low-code or no-code graphical user interface (GUI). Users connect components together, and the connections determine the flow of data through the app.

If building an agentic AI chatbot application to automate customer service, a user might first connect a chat interface to a large language model (LLM). They might also connect the LLM to their company's internal vector database to create a retrieval augmented generation (RAG) system, enabling the LLM to consult data including customer order histories.

The LLM can access tools through API keys, which can also be dropped into the AI workflow as modular components. To complete the agentic application, the LLM would be linked to a second chat component to deliver the output back to the user through the chatbot.

Key LangFlow features and functions

LangFlow's utility stems from its user-friendly core features and functions, which include:

- Low-code or no-code visual interface
- Extensive integrations

- Component library
- Exportable flows
- Open source

Low-code or no-code visual interface

LangFlow's ease-of-use is in large part due to its presentation. Users can build AI applications through a modular, drag-and-drop visual interface. Each component of the machine learning (ML) process is positioned in sequence, connected to the other components as needed by the AI workflow.

The visual interface transforms a complex coding project into an intuitive flowchart, complete with connections that dictate the flow of data through the artificial intelligence (AI) system. Beginners can use LangFlow to streamline AI orchestration by adding various models, components and data sources as needed. Meanwhile, users with Python experience can build c within within LangFlow.

As an example of no-code use, LangFlow allows users to enact limited hyperparameter tuning for their chosen LLMs with a simple slider. Users can adjust the temperature—a hyperparameter that controls the degree of randomness in an LLM output—with a quick nudge to the left or right.

Is LangFlow considered vibe coding?

Use of LangFlow is not the same as vibe coding, which is when a user instructs an LLM with natural language prompts to generate code. The user tells the LLM what the code should do and relies on the LLM to generate the actual code.

LangFlow tasks users with building the AI application they want, and replaces the need for coding with premade modular components. Users can still use code to create custom components for more advanced agentic automation.

Extensive integrations

LangFlow offers significant flexibility due to its wide array of integrations. LangFlow supports integration with numerous ML frameworks, and like its parent framework LangChain, covers the same range of API, vector database and other connection options.

LangFlow also supports LLM chaining, where multiple models are connected in sequence within a single pipeline. Chaining is different from multi-agent orchestration, where multiple agents—each potentially using its own LLM, tools or data sources—collaborate on a shared task. LangFlow's modular design supports both approaches.

What's the difference between LangFlow and LangChain?

LangChain is an open source, code-based ML framework for AI development. LangFlow is a visual tool that sits on top of ML frameworks such as LangChain, allowing users to build and rapidly prototype LLM apps. LangFlow was originally built on LangChain and is still closely tied to it, but it now supports other frameworks and integrations.

LangGraph, another platform within the same family, is also used to build agentic systems. But instead of a modular GUI, LangGraph depicts agentic systems as graphs while providing more granular control.

Component library

The component library contains all the components users can add to their agentic workflows: LLMs such as OpenAI's GPT family, Meta's Llama and others, chat interfaces, calculators, web browsers and more. LangFlow groups components into two categories:

- **Core components** that form the backbone of most LangFlow creations.
- Provider-specific **bundles** that support specific third-party integrations.

Exportable flows

LangFlow projects are exportable as flows in JSON format. Creators can export their flows and share them with other users, who can then import them into their own respective LangFlow instance to use and modify. Exportable flows enhance collaboration and streamline project workflows by making flows reusable.

Open source

Like its parent framework LangChain, LangFlow is open source, meaning that its code is publicly available for inspection, contribution and modification. Open source AI tools help increase AI explainability and provide operational transparency. However, using a closed-source LLM or other component within LangFlow does not grant similar access to its inner workings.

LangFlow use cases

LangFlow's ease of use makes it an ideal tool for streamlining and automating workflows with agentic AI. Real-world use cases for LangFlow include:

- Rapid prototyping
- AI agent development
- RAG applications
- Customer service automation

Rapid prototyping

LangFlow's drag-and-drop GUI lends itself well to rapid prototyping for AI applications. Users can draft a pipeline with LangFlow's modular components, share it with others, test it and iterate as needed. Integration with Hugging Face Spaces allows for quick ML demoing as well.

AI agent development

One of LangFlow's core use cases is no-code AI agent development. Through the component library, users can connect an LLM with tools, databases and other add-ons, enabling the agent to access what it needs to fill its intended function. Users can also chain LLMs together or build multi-agent systems.

RAG applications

With components for chat interfaces and vector databases, LangFlow can easily build RAG systems. Natural language prompts are converted to embeddings, which the retrieval model uses to query the connected vector database.

The database contains information relevant to the system's intended use case. For example, a RAG system designed to help new employees with onboarding can reference training docs in the dataset. Then, the LLM combines the retrieved data with the prompt to return a natural language output to the user.

Customer service automation

Chatbots are often used for customer service automation. Customers first interact with the chatbot, which can retrieve relevant data such as order histories and product information. Should the customer query prove too complex, the chatbot can escalate to a human representative.

A LangFlow user can quickly create a customer service chatbot with just a few components:

1. A **chat input** receives customer queries in natural language.
2. An **embedding component** converts the input into a vector embedding for semantic search.
3. A v**ector database** containing company data is queried for similar embeddings.
4. An **LLM** combines the retrieved data with the customer's query to generate a response.
5. A **chat output** returns the response to the user in natural language.