# Chapter 14 - Using APIs with LangChain

> **Harrison Chase, LangChain Creator**
>
> A system is more "agentic" the more an LLM decides how the system can behave.

two way AI and API interact:

- Use API to call a LLM
- LLM call an API
- LangChain & LangGraph are open source frameworks for creating agenting applications.

Terms that you will see in this chapter:

- Agent
  - A system that uses a LLM to decide the control flow of an application
  - Agents are not preprogrammed, they use model to reason and decide the flow of a conversation
  - They can execute tool calls that are suggested by function-calling models
- Function-Calling Model
  - Specialized type of model that considers available functions or tools and suggests when they should be used.
  - They don't call the tool directly, they give that suggestion to agent, who do the calling
- Models
  - Means AI models, can be local can be from web API
- Model Families
  - Multiple models that share a name and architecture
- Toolkit
  - Collection of multiple tools that an agent will use to perform tasks
- Tools or Functions
  - Code that provides extra skills to agent

Tools in this chapter:

- LangChain: Python Library used to create tools and toolkits that allow agents to use your API
- LangGraph: Python library used to create an agent
- Sonnet: Model used to provide reasoning to the LangGraph agent
- Pydantic: Python library used to perform validation in your toolkit

## Creating a LangGraph Agent

- Check out System Card or Model Card of a AI model before using it
  - Anthropic Model Card

> LangChain Post
>
> Letting an LLM decide the control flow of an application is attractive, as they can unlock a variety of tasks that couldn't previously be automated. In practice, however, it is incredibly difficult to build systems that reliably execute on these tasks.

- LangGraph is a project focused on creating applications that have one or more agents working together.
  - The method LangChain is using is "Legacy Method": allowing more developer control and supporting multi-agent application.
  - It uses terminology from Mathematical Graph Theory like Airflow
    - LangGraph agents = Nodes = Processes that update the state of the application
    - Edges = Flow between one node and another:
      - LangGraph allow Cyclical Graph: nodes and edges can loop multiple times

> **Rotating a Credential**
>
> Delete the keys and create new ones when your API got expose or things like that.

`NameError: name '__file__' is not defined`

Because you append it in interactive shell, you need to create `file.py` and then run it, now it works

## Additional Resources

- Building Effective AI Agents \ Anthropic
- Agents - Google
- LangGraph
- Introduction | 🦜️🔗 LangChain
- Chat models | 🦜️🔗 LangChain
- Custom Tools | 🦜️🔗 LangChain
- LangChain Overview - Docs by LangChain
- Messages | 🦜️🔗 LangChain
- Converting HumanMessage and AIMessage to Strings in LangChain
- Anthropic Model - AvalAI document
- LangChain Overview - Docs by LangChain
- OpenAI Platform
- environment variables - What is the use of python-dotenv? - Stack Overflow