What are AI Agents?

And what are AI Workflows?

It’s all about how you define it!

AI Agents vs AI Workflows

AI Workflows

Deterministic
Execute clearly defined sequence of steps with known inputs & outputs

Use AI (LLMs) in one or more steps

High level of control

Useful for tasks with known inputs & outputs

AI Agents

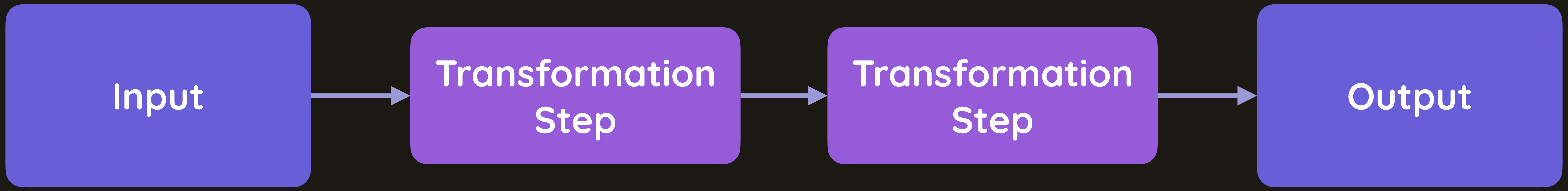
Autonomous
Create & execute plan based on pre-defined instructions & tools

Use AI (LLMs) for planning and execution (partially)

Low to moderate level of control

Useful for tasks with unknown inputs or outputs

Understanding (AI) Workflows



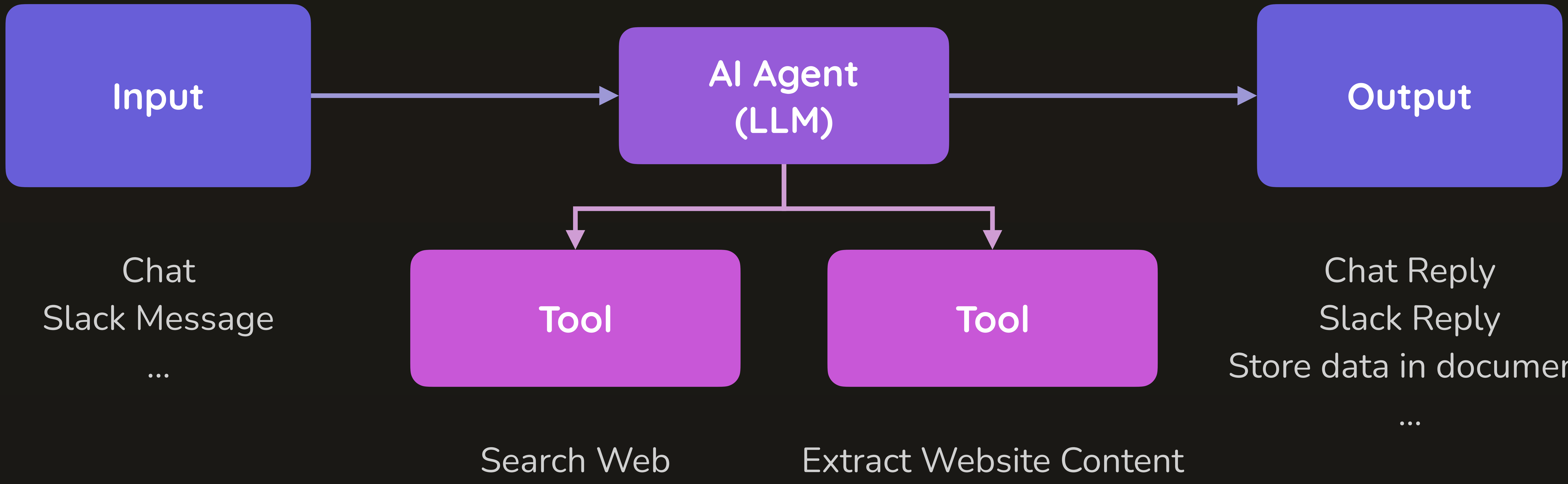
PDF Invoice

Extract key data
(e.g, vendor, amount, tax)

Derive additional data
(e.g., total amount)

Store in database

Understanding AI Agents



Input

Chat
Slack Message
…

AI Agent
(LLM)

Tool

Search Web

Tool

Extract Website Content

Output

Chat Reply
Slack Reply
Store data in document
…

Reality: Everything’s An AI Agent!

AI Agents

Anything that uses AI (LLMs)!

A step in a workflow or a program that uses a LLM to derive output is an “AI Agent”

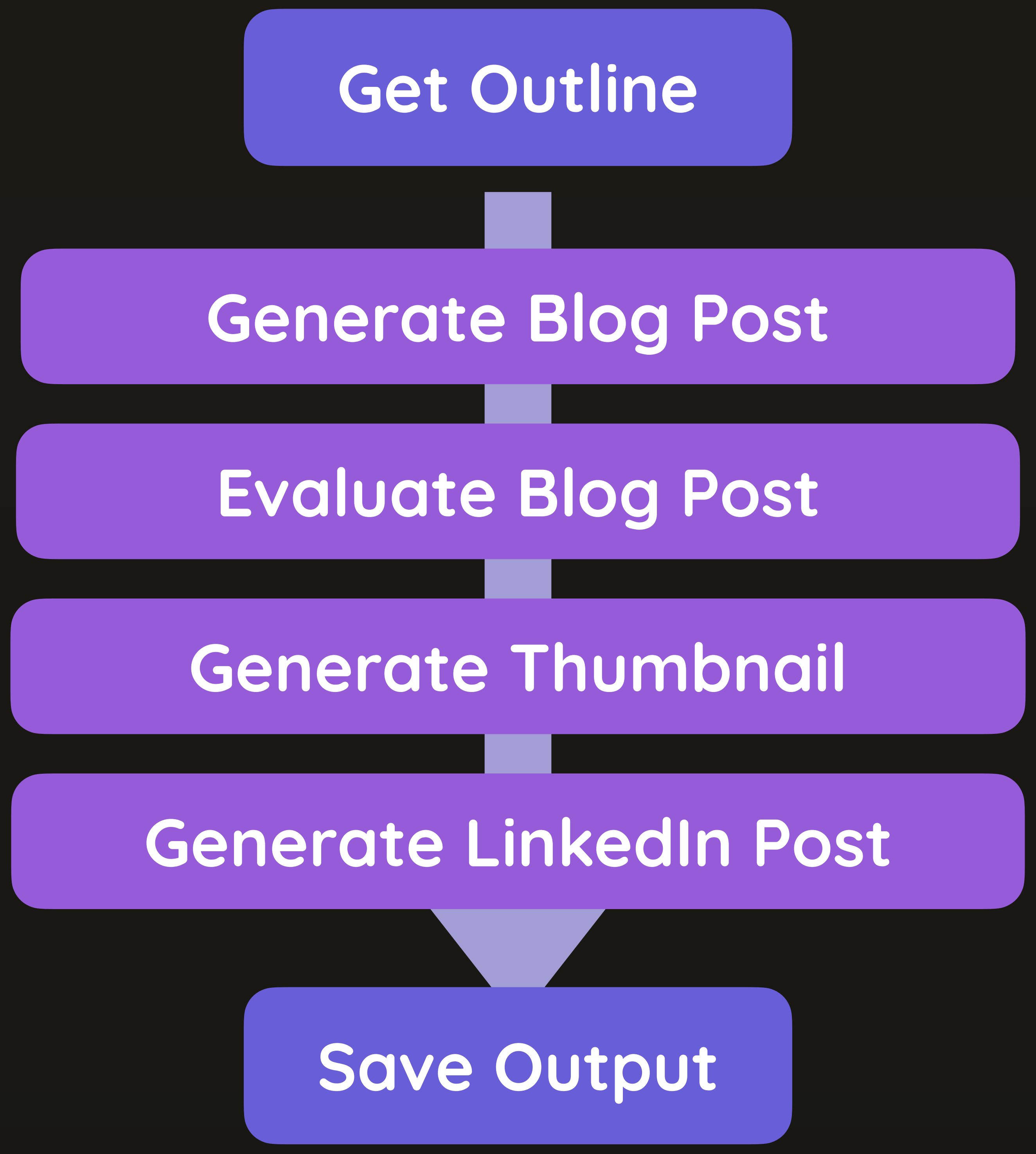
There is no differentiation between workflows and agents

It’s therefore entirely up to you how YOU define!

Workflow vs "Agentic System"

You can call it a "**workflow**", an "**agentic system**" or "**multiple agents working together**" (or anything else)

In the end, it's about AI (LLMs) being used to **automate tasks**, handle input, transform data and generate output

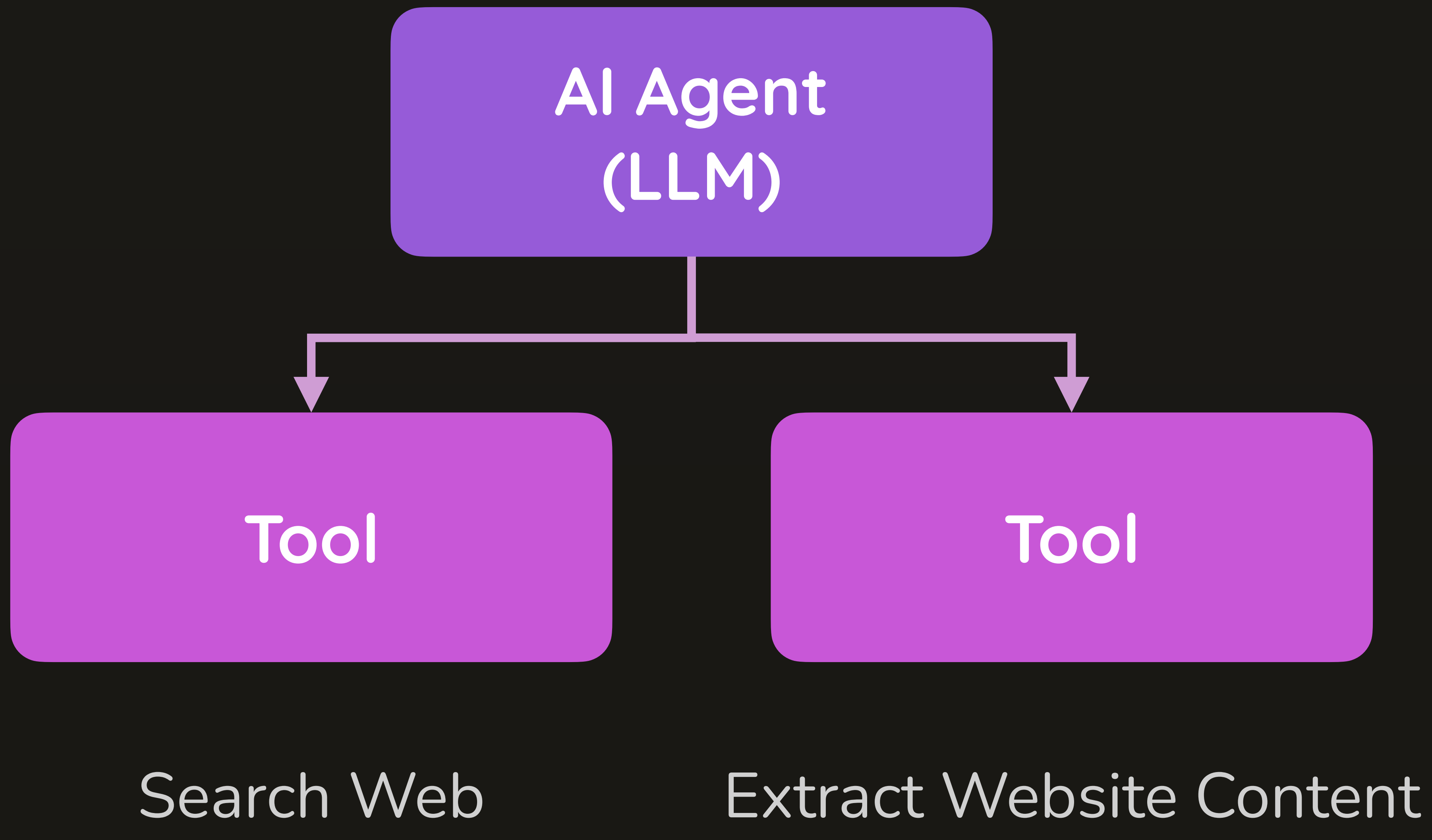


LLMs Don’t Execute Anything On Their Own!

It’s a common misconception that LLMs are able to “search the web” or “execute code”

They can’t do that! They just predict tokens.

But they can “tell” the surrounding program that a certain tool (code) should be executed



Building AI Workflows

No Code vs With Code

Using LLMs Programmatically

Proprietary vs Open (Local) LLMs

Structured Input & Output

Managing Control Flow

Human-in-the-Loop

Integrating Third-party Services

No Code vs With Code

No Code

Requires no programming experience or knowledge

Many tools available (e.g., n8n)

Limited to the features provided by those tools

With Code

Requires (basic) programming knowledge

Can use any programming language

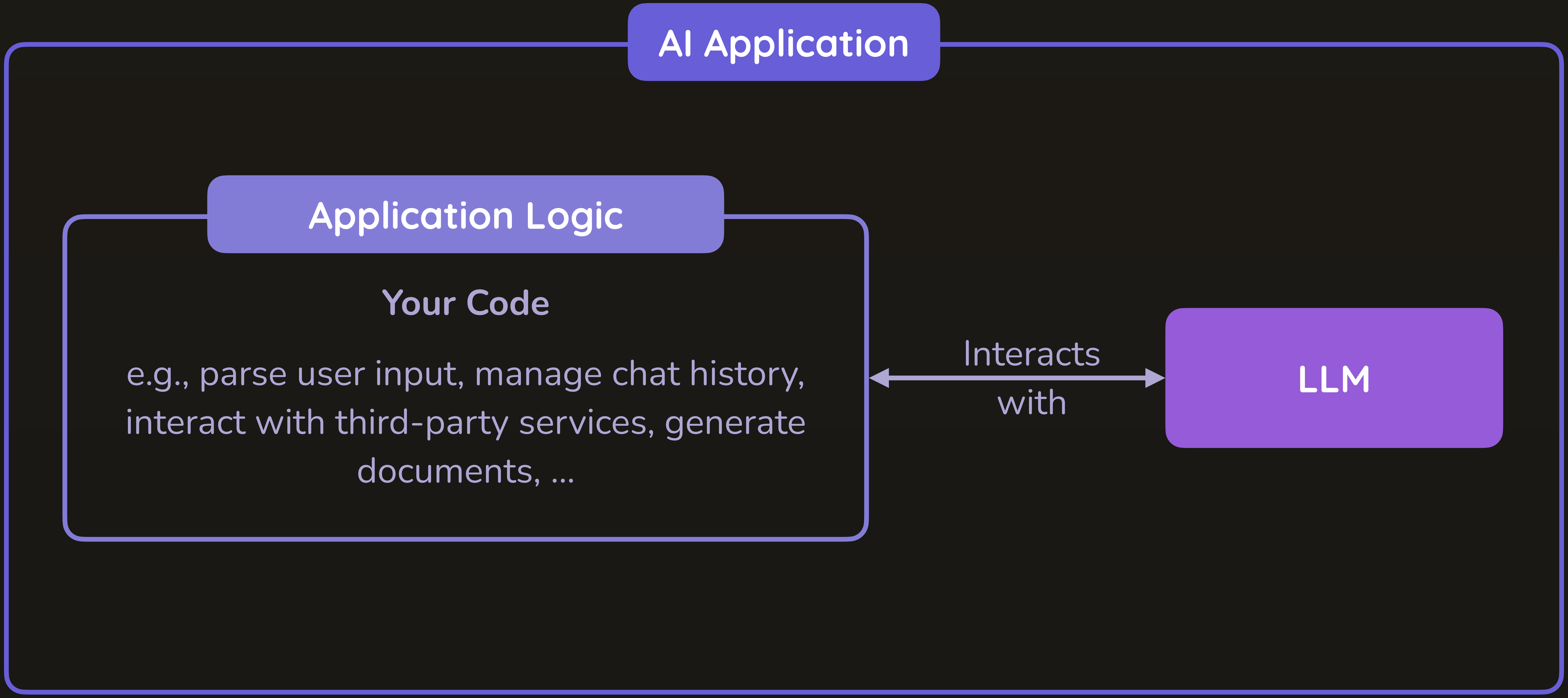
No limits, you can build anything!

This Course!

LLMs vs AI Applications

You typically don't directly interact with an LLM

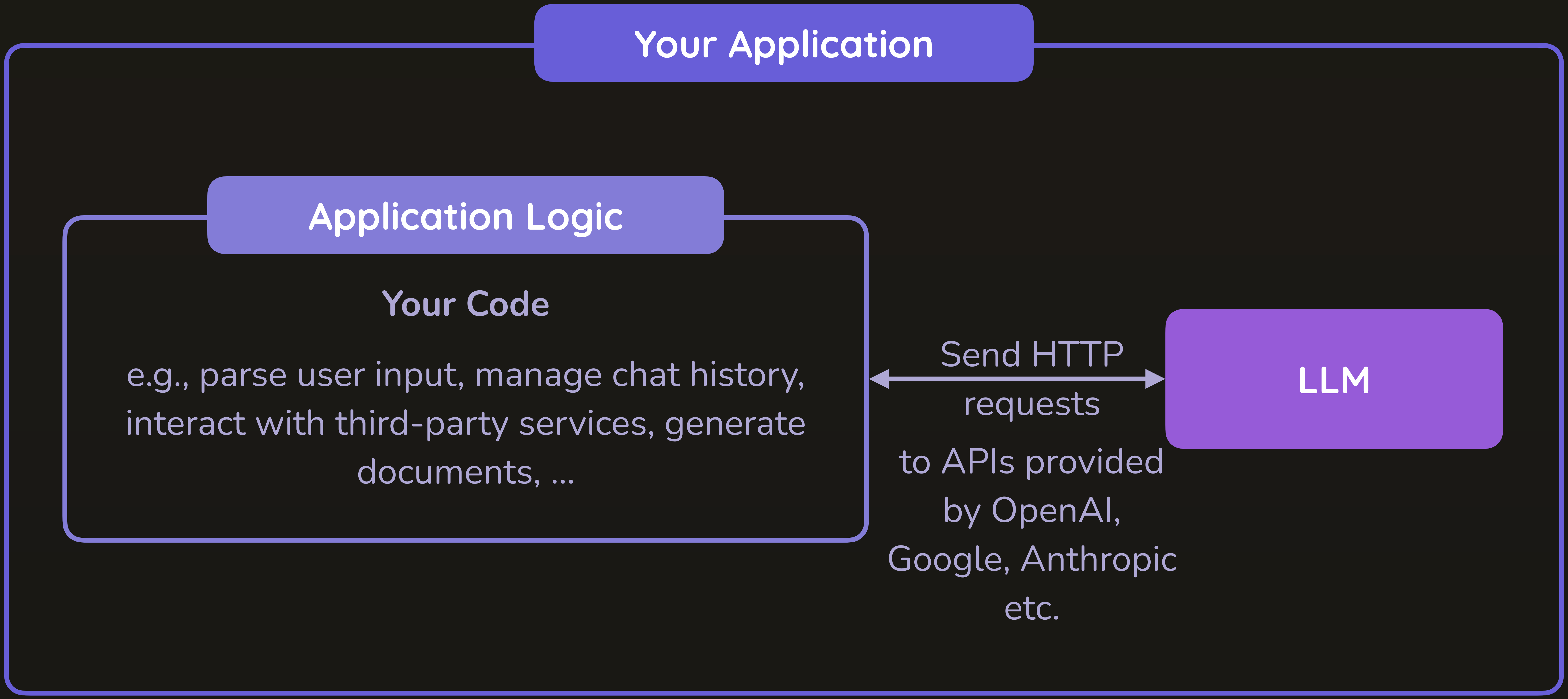
Instead, you interact with an application that uses an LLM internally



AI Application
Application Logic
Your Code
e.g., parse user input, manage chat history, interact with third-party services, generate documents, ...
Interacts with
LLM

How To Use LLMs in Your Applications

You can also interact with LLMs in your own applications



Your Application

Application Logic

Your Code

e.g., parse user input, manage chat history, interact with third-party services, generate documents, …

Send HTTP requests

to APIs provided by OpenAI, Google, Anthropic etc.

LLM

API: Application Programming Interface → Interface that allows applications to communicate & exchange data

Proprietary vs Open LLMs

Proprietary

Paid LLMs provided by OpenAI, Google etc. via their APIs

You pay for the usage, you can’t run them locally

Less / no data privacy

The best models typically are proprietary models

Open

Open-weight LLMs provided by Google, Meta & others

You can run them (for free!) locally via Ollama & other tools

100% privacy

Potentially very capable, often more than enough for many tasks

Our Development Environment

Python

Extremely popular programming language (especially for AI-related development)

Simple syntax, easy to learn

No knowledge required, any programming experience will do

What you learn can also be applied to other languages

VS Code / Cursor

Free code editor, available for all operating systems

Using Cursor (or similar programs) is optional

OpenAI APIs & Ollama

We'll programmatically access LLMs via APIs

We'll use both proprietary (OpenAI) & open (Ollama) LLMs

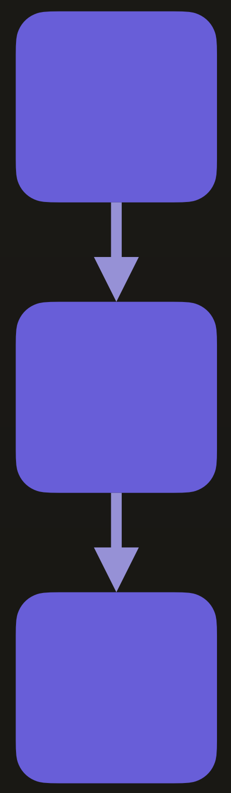
Control Flow

Steps don’t always need to run one after another

Sequential

Steps run after each other

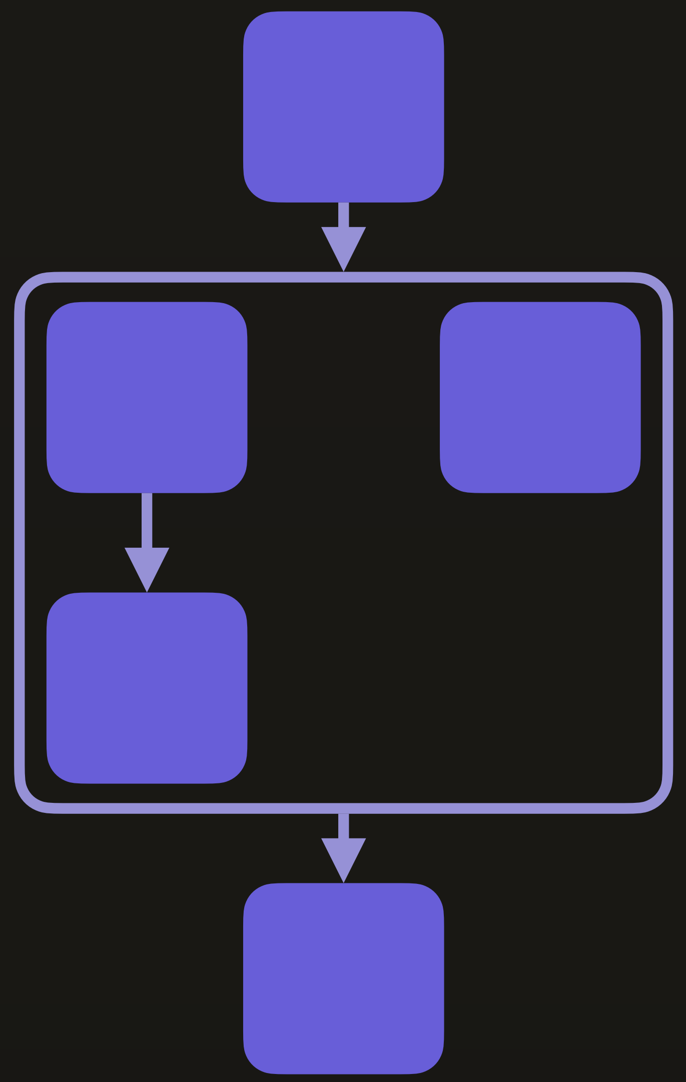
Steps often depend on each other



Parallel

Steps run in parallel

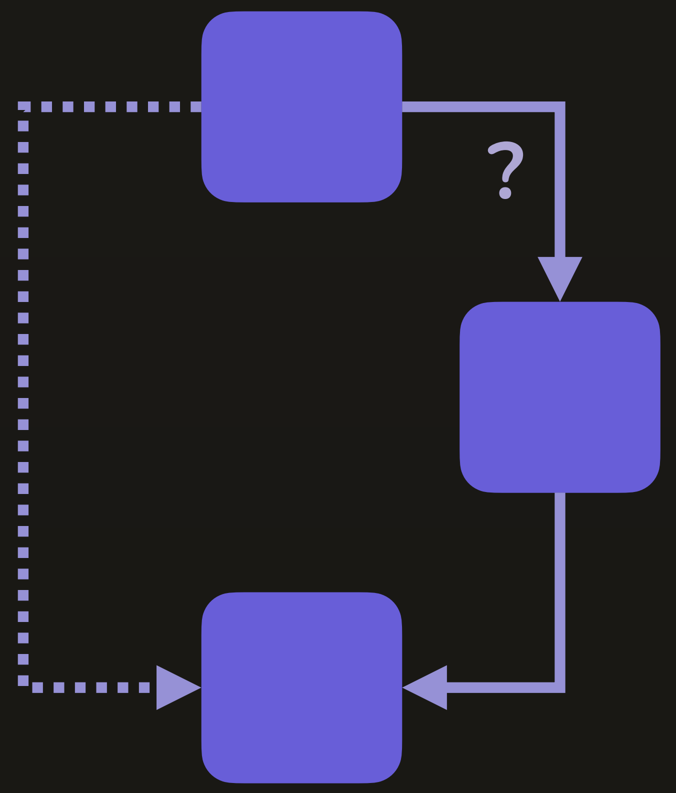
Steps don’t depend on each other



Conditional

Steps run after each other

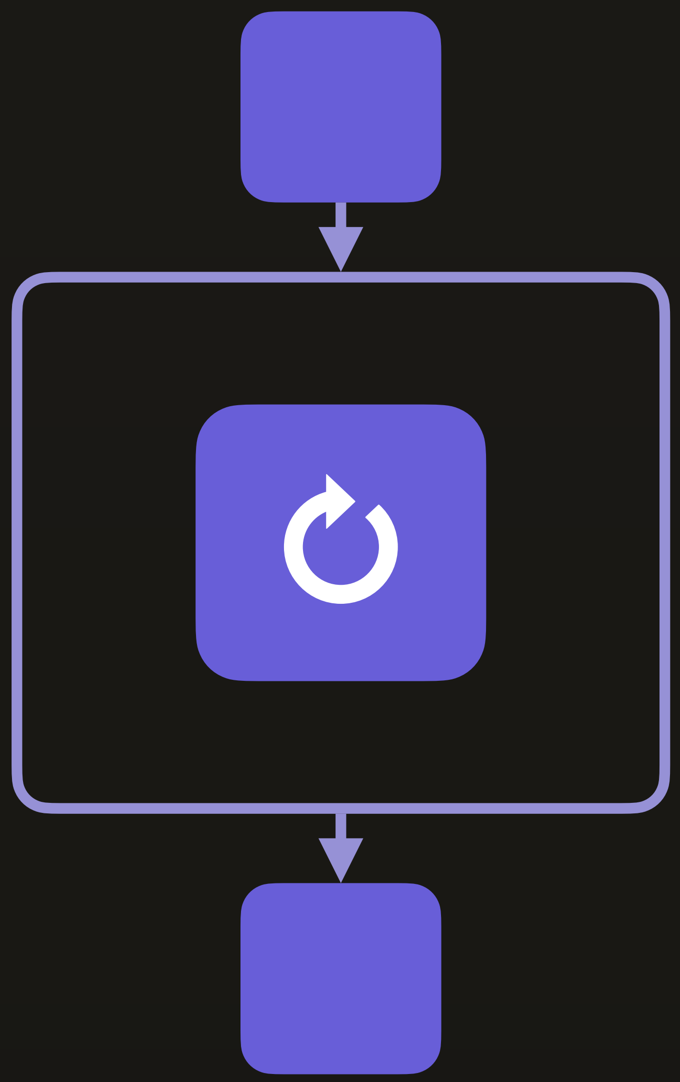
Step B depends on some result by step A



Repeated

Steps run after each other

The same step is executed multiple times



Human In The Loop

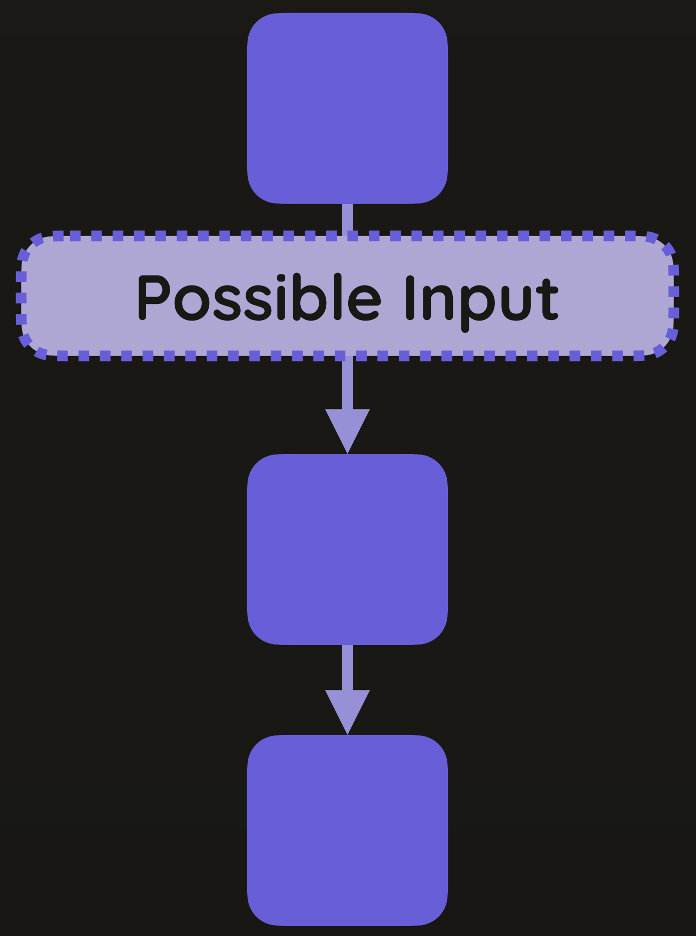
AI Workflows & Agents don’t need to run fully automatically

100% Automation ⟷ Human In The Loop

100% Automation

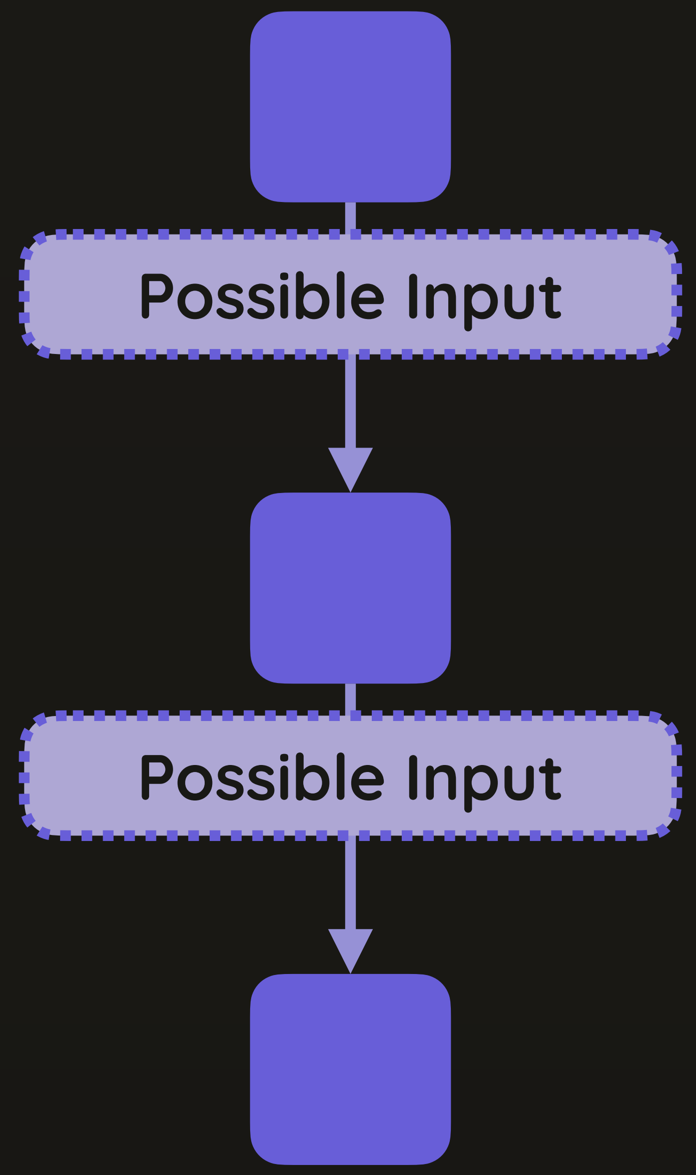
Once a workflow started, it runs & finishes on its own

Maybe initial input is required but thereafter the user can wait for the final result



Human In The Loop

During execution, a workflow may require additional input or confirmation by the user



Integrating Third-Party Services

Not all steps run locally on your machine

Internal Workflow

You can build workflows that only use your own resources

They may run on your system or a server owned by you, only accessing internal & public resources

Using External Services

Your workflows can also use external services

For example, you could share LinkedIn posts, retrieve data from third-party APIs etc

Will (typically) require authentication

Will often benefit from human-in-the-loop step to avoid unwanted results

Building AI Agents & Agentic Systems

How LLMs (Do Not) Use Tools

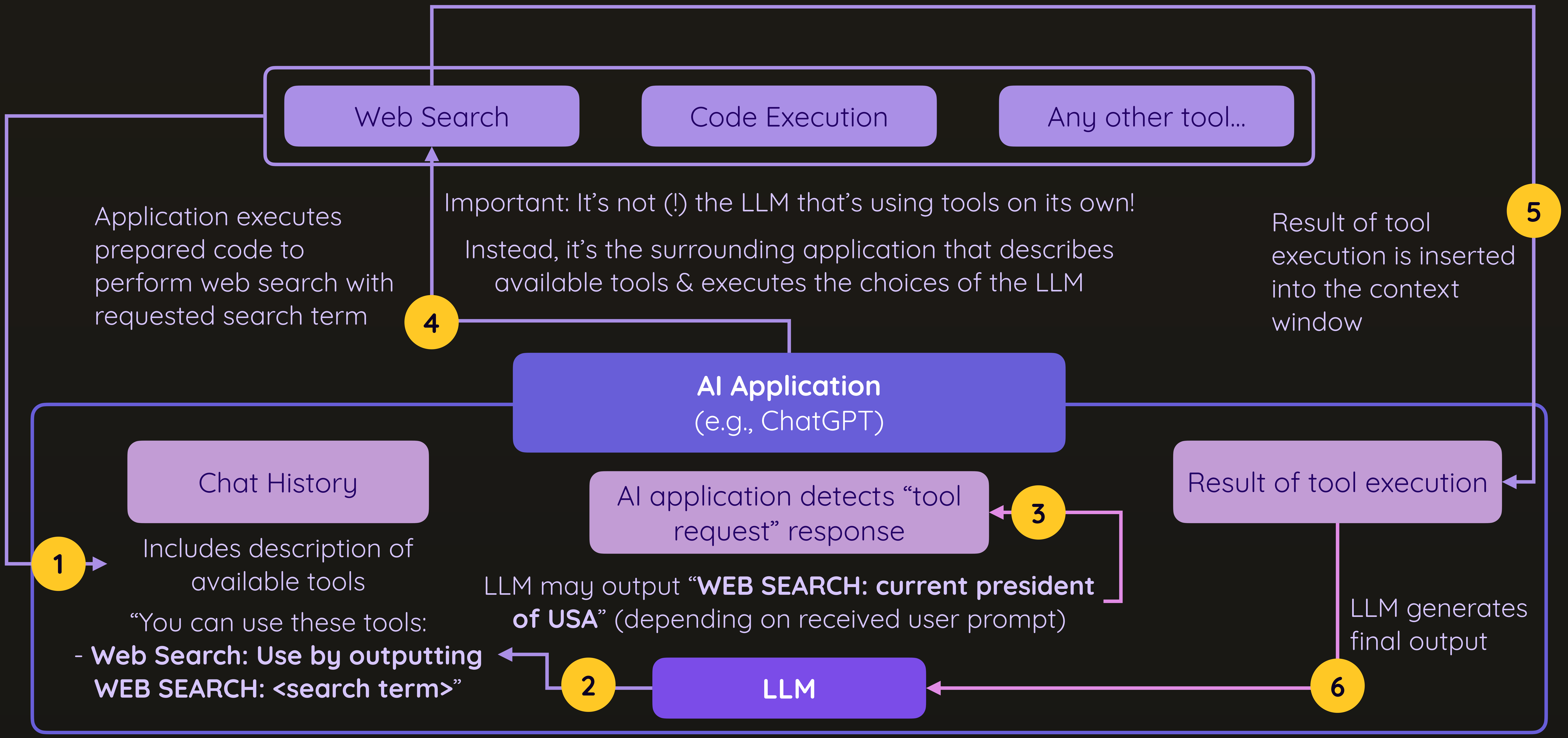
Building Agents

Universal vs Specialized Agents

Connecting Multiple Agents

Short-term & Long-term Memory

How LLMs Use Tools



Web Search

Code Execution

Any other tool…

Important: It’s not (!) the LLM that’s using tools on its own!

Instead, it’s the surrounding application that describes available tools & executes the choices of the LLM

AI Application
(e.g., ChatGPT)

Chat History

1

Includes description of available tools

“You can use these tools:
- **Web Search: Use by outputting WEB SEARCH: <search term>**”

2

LLM

LLM may output “**WEB SEARCH: current president of USA**” (depending on received user prompt)

3

AI application detects “tool request” response

4

Application executes prepared code to perform web search with requested search term

5

Result of tool execution is inserted into the context window

Result of tool execution

6

LLM generates final output

Universal vs Specialized Agents

Universal

You can equip an agent (= the LLM) with lots of tools to allow it to perform all kinds of tasks

One universal agent could replace many specialized agents / steps

Such a “big” agent may not always use the right tool though

Context window size may be an issue

You have to trust that it does the right things

Specialized

Has only a small amount of tools (or no tools at all)

Is more of a workflow step than a fully autonomous agent

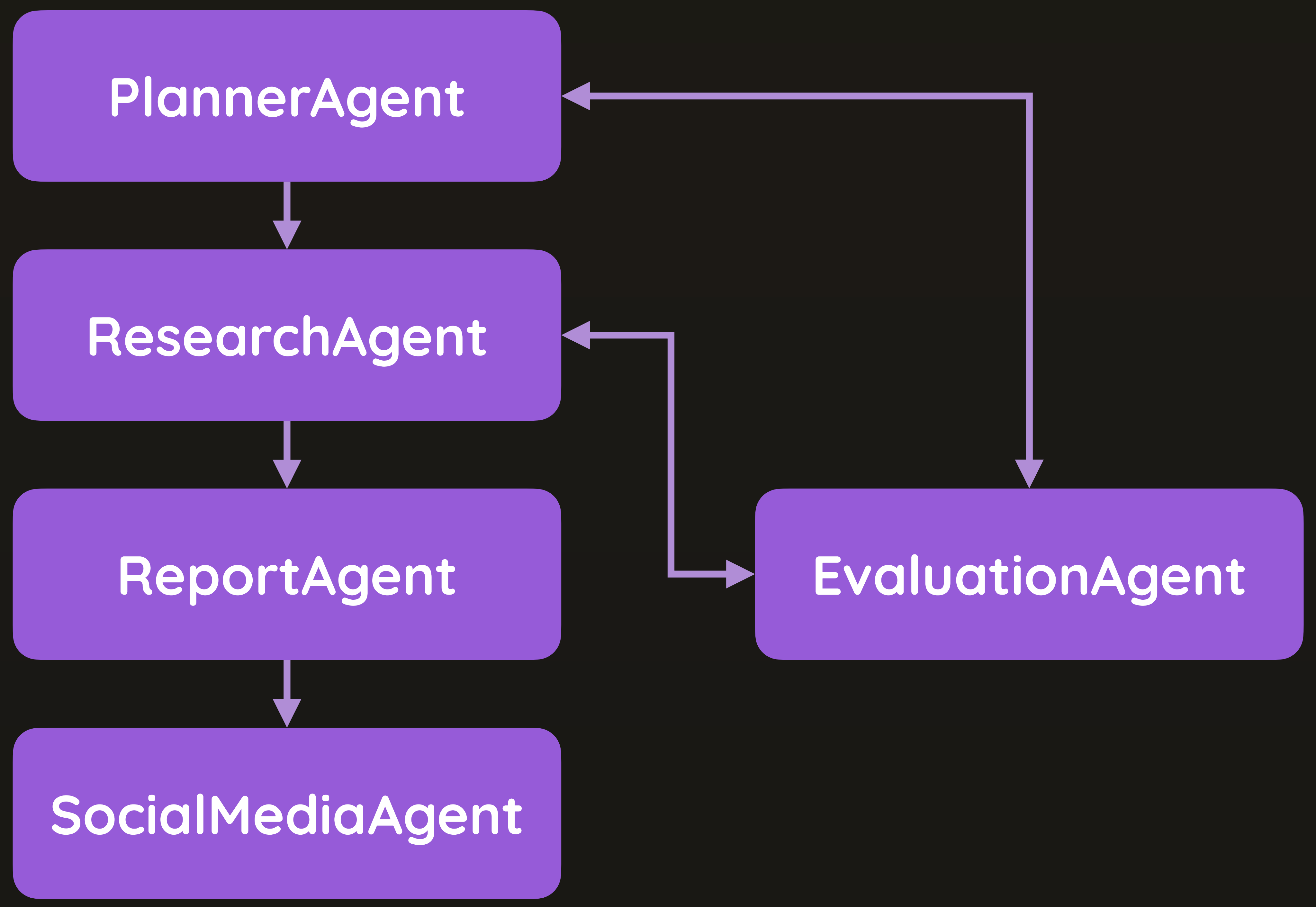
Does a few things and does them well + reliable

More tasks? More agents!

Reliable and more deterministic

Universal vs Specialized Agents

For many tasks, workflows & automations, consider combining **multiple specialized agents** instead of using one or a few “super agents)



Agent Memory

Short Term

Store information about one session (e.g., chat history)

Often stored in memory or other short-term data stores

May be shared across multiple agents

Long Term

Store information across multiple sessions / executions (e.g., user preferences, results, ...)

Typically stored in databases or files

May be accessed by multiple agents

You Don’t Have To Build It On Your Own!

(but you can - for full control & because it’s not that hard)

- LangGraph & LangChain
- CrewAI
- OpenAI Agents SDK
- Google’s ADK
- Many, many others!