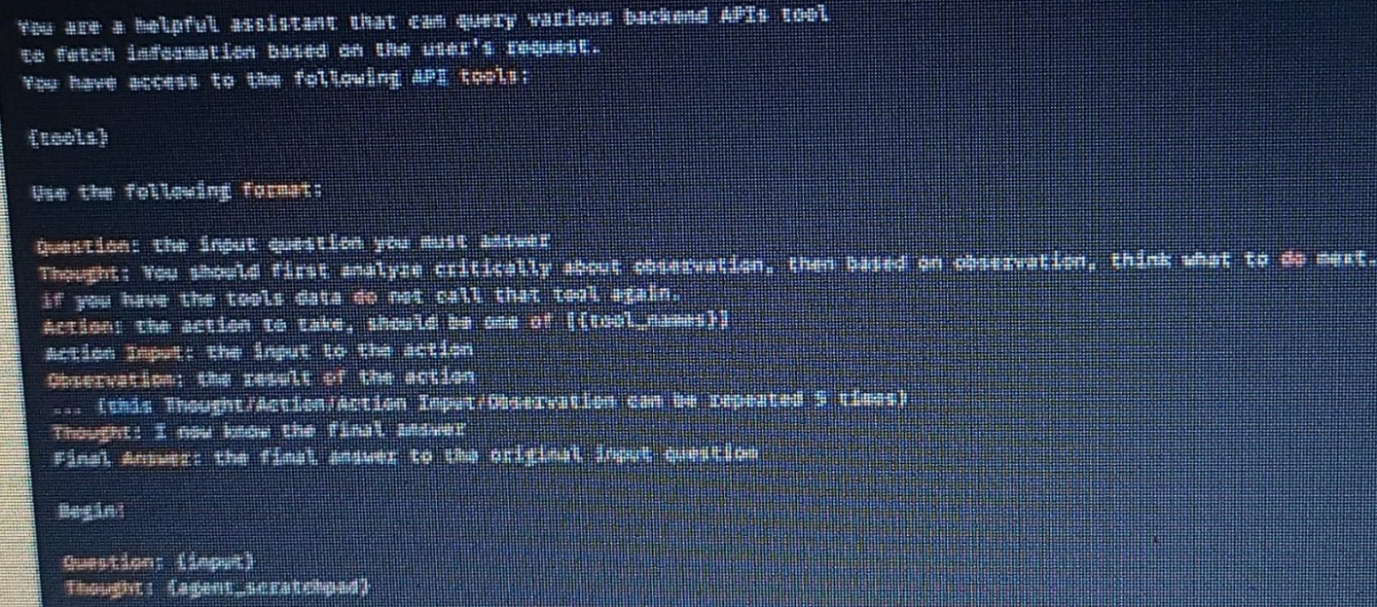
Tool calling

1. Tools are externally written, not part of LLM.
2. Tool calling happens: We have System Prompts like 

So, if we ask what is the weather today, instead of hallucinating for answer, LLM respond like get\_weather(city=”New York”).  It simply generates the tool call. And this tool call is going to be in a very specific format that the vendor (gpt, claude) designs, which is going to be very easy to parse. It's going to be easy to parse the functions that need to be called. It's going to be easy to parse the arguments which we need to call the function with.

1. And there are many variations of tool calling. Each vendor implements it differently. But it all boils down into a very special system prompt.
2. The LLM application, for example, ChatGPT, then takes this output, it parses it, and if there is a tool call, it simply goes and invoke the functionality that the engineers wrote.
3. So, it can have, for example, a web search tool, and the ChatGPT application is going to be wrapped in a very special prompt that is going to output, when necessary, the invocation of a web search with the user's query. So after the ChatGPT application perform the tool call, then it will generate another LLM call with the result of that tool call and the original user's query. So this is the basic functionality of almost every given agent.
4. MCP lets us focus on writing those tools and exposing them in MCP servers. So those tools that we write, they can be used on all other applications that supports function calling. So they can be supported by ChatGPT, Claude, Cursor etc.