

Documentation

Tool Use

Introduction to Tool Use

Tool use is a powerful feature that allows Large Language Models (LLMs) to interact with external resources, such as APIs, databases, and the web, to gather dynamic data they wouldn't otherwise have access to in their pre-trained (or static) state and perform actions beyond simple text generation.

Tool use bridges the gap between the data that the LLMs were trained on with dynamic data and real-world actions, which opens up a wide array of realtime use cases for us to build powerful applications with, especially with Groq's insanely fast inference speed. 🚀

How Tool Use Works

Groq API tool use structure is compatible with OpenAI's tool use structure, which allows for easy integration. See the following cURL example of a tool use request:

```
curl https://api.groq.com/openai/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $GROQ_API_KEY" \
-d '{
  "model": "llama-3.3-70b-versatile",
  "messages": [
    {
      "role": "user",
      "content": "What's the weather like in Boston today?"
    }
  ],
  "tools": [
    {
      "type": "function",
      "function": {
        "name": "get_current_weather",
        "description": "Get the current weather in a given location",
        "parameters": {
          "type": "object",
            "location": {
              "type": "string",
              "description": "The city and state, e.g. San Francisco, CA"
            },
            "unit": {
              "type": "string",
              "description": "The unit of measurement, e.g. Celsius, Fahrenheit"
            }
          }
        }
      }
    ]
  }
}
```

To integrate tools with the Groq API, follow these steps:

1. Provide tools (or predefined functions) to the LLM for performing actions and accessing external data in real-time in addition to your user prompt within your Groq API request
2. Define how the tools should be used to teach the LLM how to use them effectively (e.g. by defining input and output formats)
3. Let the LLM autonomously decide whether or not the provided tools are needed for a user query by evaluating the user query, determining whether the tools can enhance its response, and utilizing the tools accordingly
4. Extract tool input, execute the tool code, and return results
5. Let the LLM use the tool result to formulate a response to the original prompt

This process allows the LLM to perform tasks such as real-time data retrieval, complex calculations, and external API interaction, all while maintaining a natural conversation with our end user.

Tool Use with Groq

Groq API endpoints support tool use to almost instantly deliver structured JSON output that can be used to directly invoke functions from desired external resources.

Supported Models

The following models are recommended for tool use due to their versatility and performance:

- **deepseek-r1-distill-llama-70b**
- **llama-3.3-70b-versatile**
- **llama-3.1-8b-instant**

Other Supported Models

The following models powered by Groq also support tool use:

- **mixtral-8x7b-32768** (parallel tool use not supported)
- **gemma2-9b-it** (parallel tool use not supported)

Tools Specifications

Tool use is part of the [Groq API chat completion request payload](#).

Tool Call and Tool Response Structure

Tool Call Structure

Groq API tool calls are structured to be OpenAI-compatible. The following is an example tool call structure:

```
{
  "model": "llama-3.3-70b-versatile",
  "messages": [
    {
      "role": "system",
      "content": "You are a weather assistant. Use the get_weather function to retrieve weather information for a location."
    },
    {
      "role": "user",
      "content": "What's the weather like in New York today?"
    }
  ],
  "tools": [
    {
      "type": "function",
      "function": {
        "name": "get_weather",
        "description": "Get the current weather for a location",
        "parameters": {
          "type": "object",
          "properties": {
            "location": {
              "type": "string",
              "description": "The city and state, e.g. San Francisco, CA"
            },
            "unit": {
              "type": "string",
              "enum": ["celsius", "fahrenheit"],
              "description": "The unit of temperature to use. Defaults to fahrenheit."
            }
          },
          "required": ["location"]
        }
      }
    }
  ]
}
```

Tool Call Response

The following is an example tool call response based on the above:

```
{
  "model": "llama-3.3-70b-versatile",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "The current weather in New York is 68 degrees Fahrenheit.",
        "tool_calls": [
          {
            "id": "call_d5wg",
            "type": "function",
            "function": {
              "name": "get_weather",
              "arguments": "{\"location\": \"New York, NY\"}"
            }
          }
        ]
      },
      "finish_reason": "tool_calls"
    }
  ],
  "tool_choice": "auto",
  "max_completion_tokens": 4096,
  "logprobs": null
}
```

When a model decides to use a tool, it returns a response with a `tool_calls` object containing:

- `id`: a unique identifier for the tool call
- `type`: the type of tool call, i.e. function
- `name`: the name of the tool being used
- `parameters`: an object containing the input being passed to the tool

Setting Up Tools

To get started, let's go through an example of tool use with Groq API that you can use as a base to build more tools on your own.

Step 1: Create Tool

Let's install Groq SDK, set up our Groq client, and create a function called `calculate` to evaluate a mathematical expression that we will represent as a tool.

Note: In this example, we're defining a function as our tool, but your tool can be any function or an external resource (e.g. database, web search engine, external API).

Python JavaScript

```
pip install groq
```

```
1 from groq import Groq
2 import json
3
4 # Initialize the Groq client
5 client = Groq()
6 # Specify the model to be used (we recommend Llama 3.3 70B)
7 MODEL = 'llama-3.3-70b-versatile'
8
9 def calculate(expression):
10     """Evaluate a mathematical expression"""
11     try:
12         # Attempt to evaluate the math expression
13         result = eval(expression)
14         return json.dumps({"result": result})
15     except:
16         # Return an error message if the math expression is invalid
17         return json.dumps({"error": "Invalid expression"})
```

Step 2: Pass Tool Definition and Messages to Model

Next, we'll define our `calculate` tool within an array of available tools and call our Groq API chat completion. You can read more about tool schema and supported required and optional fields above in **Tool Specifications**.

By defining our tool, we'll inform our model about what our tool does and have the model decide whether or not to use the tool. We should be as descriptive and specific as possible for our model to be able to make the correct tool use decisions.

In addition to our tools array, we will provide our messages array (e.g. containing system prompt, assistant prompt, and/or user prompt).

Step 3: Receive and Handle Tool Results

After executing our chat completion, we'll extract our model's response and check for tool calls.

If the model decides that no tools should be used and does not generate a tool or function call, then the response will be a normal chat completion (i.e. `response_message = response.choices[0].message`) with a direct model reply to the user query.

If the model decides that tools should be used and generates a tool or function call, we will:

1. Define available tool or function,
2. Add the model's response to the conversation by appending our message
3. Process the tool call and add the tool response to our message
4. Make a second Groq API call with the updated conversation
5. Return the final response

Python JavaScript

```
1 # imports calculate function from step 1
2 def run_conversation(user_prompt):
3     # Initialize the conversation with system and user messages
4     messages=[
```

```

5      {
6          "role": "system",
7          "content": "You are a calculator assistant. Use the calculate function to perform mathematical operations",
8      },
9      {
10         "role": "user",
11         "content": user_prompt,
12     }
13 ]
14 # Define the available tools (i.e. functions) for our model to use
15 tools = [
16     {
17         "type": "function",
18         "function": {

```

Routing System

If you use our models fine-tuned for tool use, we recommended to use them as part of a routing system:

- Query Analysis:** Implement a routing system that analyzes incoming user queries to determine their nature and requirements.
- Model Selection:** Based on the query analysis, route the request to the most appropriate model:
 - For queries involving function calling, API interactions, or structured data manipulation, use the Llama 3 Groq Tool Use models.
 - For general knowledge, open-ended conversations, or tasks not specifically related to tool use, route to a general-purpose language model, such as Llama 3.70B.

The following is the calculate tool we built in the above steps enhanced to include a routing system that routes our request to Llama 3.70B if the user query does not require the tool:

Python JavaScript

```

35 response = client.chat.completions.create(
36     model=MODEL, # LLM to use
37     messages=messages, # Conversation history
38     stream=False,
39     from_groq_endpoint=True, # Available tools (i.e. functions) for our LLM to use
40     import json, choice="auto", # Let our LLM decide when to use tools
41     max_completion_tokens=4096 # Maximum number of tokens to allow in our response
42 ) # Initialize the Groq client
43 # Extract the response and any tool call responses
44 response_message = response.choices[0].message
45 # Define models
46 ROUTING_MODEL = "llama3-70b-8192"
47 TOOL_USE_MODEL = "llama3-70b-versatile"
48 GENERAL_MODEL = "llama3-70b-8192"
49 # Define the available tools that can be called by the LLM
50 "calculate": calculate,
51 def calculate(expression):
52     """To add to evaluate a mathematical expression"""
53     try:
54         messages.append(response_message)
55         result = eval(expression)
56         # Append each tool result
57         return [{"tool_call": "error", "invalid_expression"}]
58     except:
59         tool_call in tool_calls:
60             function_to_call = available_functions[function_name]
61 def route_query(query_args = json.loads(tool_call.function.arguments)
62     """To decide which tool to use if needed"""
63     routing_prompt = f"""
64     We learned about tool use and built single-turn tool use examples above. Now let's take tool use a step further and
65     imagine a workflow where multiple tools can be called simultaneously, enabling more efficient and effective
66     responses.
67     If a calculation tool is needed, respond with 'TOOL: CALCULATE'.
68     If no tool is needed, respond with 'NO TOOL'.
69     """
70     messages.append(
71         {"role": "user", "content": routing_prompt}
72     )
73     response = client.chat.completions.create(
74         model=ROUTING_MODEL,
75         messages=messages,
76         max_completion_tokens=20 # We only need a short response

```

Parallel Tool Use

We learned about tool use and built single-turn tool use examples above. Now let's take tool use a step further and imagine a workflow where multiple tools can be called simultaneously, enabling more efficient and effective responses.

If no tool is needed, respond with 'NO TOOL'.

This concept is known as **parallel tool use** and is key for building agentic workflows that can deal with complex queries, which is a great example of where inference speed becomes increasingly important (and thankfully we can access fast inference speed with Groq API).

Note: Parallel tool use is natively enabled for all Llama 3 and Llama 3.1 models!

Here's an example of parallel tool use with a tool for getting the temperature and the tool for getting the weather condition to show parallel tool use with Groq API in action:

```

32 response = client.chat.completions.create(
33     model=ROUTING_MODEL,
34     messages=messages,
35     max_completion_tokens=20 # We only need a short response

```

```

39 ) # Return the final response
40 return second_response.choices[0].message.content
41 # From the response, get the tool call
42 # From the response, get the tool call
43 prompt = "What is 25 * 4 + 10?"
44 print(f"From the Groq API, the tool needed is: {decision}")
45 # Initialize the Groq client
46 client = Groq()
47 model = "llama-3.1-70b-instruct"
48 # Define the tools
49 # Define the tools
50 def get_weather_condition(location: str):
51     # This is a mock tool/function. In a real scenario, you would call a weather API.
52     temperatures = {"New York": 22, "London": 18, "Tokyo": 26, "Sydney": 20}
53     return temperatures.get(location, "Temperature data not available")
54     "content": "You are a calculator assistant. Use the calculate function to perform mathematical operations."
55 def get_weather_condition(location: str):
56     # This is a mock tool/function. In a real scenario, you would call a weather API.
57     condition = {"New York": "Sunny", "London": "Rainy", "Tokyo": "Cloudy", "Sydney": "Clear"}
58     return condition.get(location, "Weather condition data not available")
59 }
60 # Define system messages and tools
61 messages = [

```

Error Handling

Groq API tool use is designed to verify whether a model generates a valid tool call. When a model fails to generate a valid tool call object, Groq API will return a 400 error with an explanation in the "failed_generation" field of the JSON body that is returned.

Next Steps

For more information and examples of working with multiple tools in parallel using Groq API and Instructor, see our [Groq API Cookbook](#).

Tool Use with Structured Outputs (Python)

Groq API offers best-effort matching for parameters, which means the model could occasionally miss parameters or misinterpret types for more complex tool calls. We recommend the [Instructor](#) library to simplify the process of working with structured data and to ensure that the model's output adheres to a predefined schema.

Here's an example of how to implement tool use using the [Instructor](#) library with Groq API:

```

38 }
39 ]
40 response = client.chat.completions.create(
41     model=model,
42     messages=messages,
43     tools=tools,
44     { tool_choice="auto",
45       type="function",
46     }
47 )
48 # Define the tool schema
49 # Define the tool schema
50 tool_schema = {
51     "name": "get_weather_condition",
52     "description": "Get the weather information for a given location",
53     "parameters": {
54         "type": "object",
55         "properties": {
56             "location": {
57                 "type": "string",
58             },
59         },
60     },
61 }
62 # Define the tool schema
63 # Define the tool schema
64 tool_schema = {
65     "name": "calculate",
66     "description": "Calculate the result of a mathematical expression",
67     "parameters": {
68         "type": "object",
69         "properties": {
70             "expression": {
71                 "type": "string",
72             },
73         },
74     },
75 }
76 # Define the tool schema
77 # Define the tool schema
78 tool_schema = {
79     "name": "get_weather_condition",
80     "description": "Get the weather information for a given location",
81     "parameters": {
82         "type": "object",
83         "properties": {
84             "location": {
85                 "type": "string",
86             },
87         },
88     },
89 }
90 # Define the tool schema
91 # Define the tool schema
92 tool_schema = {
93     "name": "calculate",
94     "description": "Calculate the result of a mathematical expression",
95     "parameters": {
96         "type": "object",
97         "properties": {
98             "expression": {
99                 "type": "string",
100             },
101         },
102     },
103 }
104 # Define the tool schema
105 # Define the tool schema
106 tool_schema = {
107     "name": "get_weather_condition",
108     "description": "Get the weather information for a given location",
109     "parameters": {
110         "type": "object",
111         "properties": {
112             "location": {
113                 "type": "string",
114             },
115         },
116     },
117 }
118 # Define the tool schema
119 # Define the tool schema
120 tool_schema = {
121     "name": "calculate",
122     "description": "Calculate the result of a mathematical expression",
123     "parameters": {
124         "type": "object",
125         "properties": {
126             "expression": {
127                 "type": "string",
128             },
129         },
130     },
131 }
132 # Define the tool schema
133 # Define the tool schema
134 tool_schema = {
135     "name": "get_weather_condition",
136     "description": "Get the weather information for a given location",
137     "parameters": {
138         "type": "object",
139         "properties": {
140             "location": {
141                 "type": "string",
142             },
143         },
144     },
145 }
146 # Define the tool schema
147 # Define the tool schema
148 tool_schema = {
149     "name": "calculate",
150     "description": "Calculate the result of a mathematical expression",
151     "parameters": {
152         "type": "object",
153         "properties": {
154             "expression": {
155                 "type": "string",
156             },
157         },
158     },
159 }
160 # Define the tool schema
161 # Define the tool schema
162 tool_schema = {
163     "name": "get_weather_condition",
164     "description": "Get the weather information for a given location",
165     "parameters": {
166         "type": "object",
167         "properties": {
168             "location": {
169                 "type": "string",
170             },
171         },
172     },
173 }
174 # Define the tool schema
175 # Define the tool schema
176 tool_schema = {
177     "name": "calculate",
178     "description": "Calculate the result of a mathematical expression",
179     "parameters": {
180         "type": "object",
181         "properties": {
182             "expression": {
183                 "type": "string",
184             },
185         },
186     },
187 }
188 # Define the tool schema
189 # Define the tool schema
190 tool_schema = {
191     "name": "get_weather_condition",
192     "description": "Get the weather information for a given location",
193     "parameters": {
194         "type": "object",
195         "properties": {
196             "location": {
197                 "type": "string",
198             },
199         },
200     },
201 }
202 # Define the tool schema
203 # Define the tool schema
204 tool_schema = {
205     "name": "calculate",
206     "description": "Calculate the result of a mathematical expression",
207     "parameters": {
208         "type": "object",
209         "properties": {
210             "expression": {
211                 "type": "string",
212             },
213         },
214     },
215 }
216 # Define the tool schema
217 # Define the tool schema
218 tool_schema = {
219     "name": "get_weather_condition",
220     "description": "Get the weather information for a given location",
221     "parameters": {
222         "type": "object",
223         "properties": {
224             "location": {
225                 "type": "string",
226             },
227         },
228     },
229 }
230 # Define the tool schema
231 # Define the tool schema
232 tool_schema = {
233     "name": "calculate",
234     "description": "Calculate the result of a mathematical expression",
235     "parameters": {
236         "type": "object",
237         "properties": {
238             "expression": {
239                 "type": "string",
240             },
241         },
242     },
243 }
244 # Define the tool schema
245 # Define the tool schema
246 tool_schema = {
247     "name": "get_weather_condition",
248     "description": "Get the weather information for a given location",
249     "parameters": {
250         "type": "object",
251         "properties": {
252             "location": {
253                 "type": "string",
254             },
255         },
256     },
257 }
258 # Define the tool schema
259 # Define the tool schema
260 tool_schema = {
261     "name": "calculate",
262     "description": "Calculate the result of a mathematical expression",
263     "parameters": {
264         "type": "object",
265         "properties": {
266             "expression": {
267                 "type": "string",
268             },
269         },
270     },
271 }
272 # Define the tool schema
273 # Define the tool schema
274 tool_schema = {
275     "name": "get_weather_condition",
276     "description": "Get the weather information for a given location",
277     "parameters": {
278         "type": "object",
279         "properties": {
280             "location": {
281                 "type": "string",
282             },
283         },
284     },
285 }
286 # Define the tool schema
287 # Define the tool schema
288 tool_schema = {
289     "name": "calculate",
290     "description": "Calculate the result of a mathematical expression",
291     "parameters": {
292         "type": "object",
293         "properties": {
294             "expression": {
295                 "type": "string",
296             },
297         },
298     },
299 }
300 # Define the tool schema
301 # Define the tool schema
302 tool_schema = {
303     "name": "get_weather_condition",
304     "description": "Get the weather information for a given location",
305     "parameters": {
306         "type": "object",
307         "properties": {
308             "location": {
309                 "type": "string",
310             },
311         },
312     },
313 }
314 # Define the tool schema
315 # Define the tool schema
316 tool_schema = {
317     "name": "calculate",
318     "description": "Calculate the result of a mathematical expression",
319     "parameters": {
320         "type": "object",
321         "properties": {
322             "expression": {
323                 "type": "string",
324             },
325         },
326     },
327 }
328 # Define the tool schema
329 # Define the tool schema
330 tool_schema = {
331     "name": "get_weather_condition",
332     "description": "Get the weather information for a given location",
333     "parameters": {
334         "type": "object",
335         "properties": {
336             "location": {
337                 "type": "string",
338             },
339         },
340     },
341 }
342 # Define the tool schema
343 # Define the tool schema
344 tool_schema = {
345     "name": "calculate",
346     "description": "Calculate the result of a mathematical expression",
347     "parameters": {
348         "type": "object",
349         "properties": {
350             "expression": {
351                 "type": "string",
352             },
353         },
354     },
355 }
356 # Define the tool schema
357 # Define the tool schema
358 tool_schema = {
359     "name": "get_weather_condition",
360     "description": "Get the weather information for a given location",
361     "parameters": {
362         "type": "object",
363         "properties": {
364             "location": {
365                 "type": "string",
366             },
367         },
368     },
369 }
370 # Define the tool schema
371 # Define the tool schema
372 tool_schema = {
373     "name": "calculate",
374     "description": "Calculate the result of a mathematical expression",
375     "parameters": {
376         "type": "object",
377         "properties": {
378             "expression": {
379                 "type": "string",
380             },
381         },
382     },
383 }
384 # Define the tool schema
385 # Define the tool schema
386 tool_schema = {
387     "name": "get_weather_condition",
388     "description": "Get the weather information for a given location",
389     "parameters": {
390         "type": "object",
391         "properties": {
392             "location": {
393                 "type": "string",
394             },
395         },
396     },
397 }
398 # Define the tool schema
399 # Define the tool schema
400 tool_schema = {
401     "name": "calculate",
402     "description": "Calculate the result of a mathematical expression",
403     "parameters": {
404         "type": "object",
405         "properties": {
406             "expression": {
407                 "type": "string",
408             },
409         },
410     },
411 }
412 # Define the tool schema
413 # Define the tool schema
414 tool_schema = {
415     "name": "get_weather_condition",
416     "description": "Get the weather information for a given location",
417     "parameters": {
418         "type": "object",
419         "properties": {
420             "location": {
421                 "type": "string",
422             },
423         },
424     },
425 }
426 # Define the tool schema
427 # Define the tool schema
428 tool_schema = {
429     "name": "calculate",
430     "description": "Calculate the result of a mathematical expression",
431     "parameters": {
432         "type": "object",
433         "properties": {
434             "expression": {
435                 "type": "string",
436             },
437         },
438     },
439 }
440 # Define the tool schema
441 # Define the tool schema
442 tool_schema = {
443     "name": "get_weather_condition",
444     "description": "Get the weather information for a given location",
445     "parameters": {
446         "type": "object",
447         "properties": {
448             "location": {
449                 "type": "string",
450             },
451         },
452     },
453 }
454 # Define the tool schema
455 # Define the tool schema
456 tool_schema = {
457     "name": "calculate",
458     "description": "Calculate the result of a mathematical expression",
459     "parameters": {
460         "type": "object",
461         "properties": {
462             "expression": {
463                 "type": "string",
464             },
465         },
466     },
467 }
468 # Define the tool schema
469 # Define the tool schema
470 tool_schema = {
471     "name": "get_weather_condition",
472     "description": "Get the weather information for a given location",
473     "parameters": {
474         "type": "object",
475         "properties": {
476             "location": {
477                 "type": "string",
478             },
479         },
480     },
481 }
482 # Define the tool schema
483 # Define the tool schema
484 tool_schema = {
485     "name": "calculate",
486     "description": "Calculate the result of a mathematical expression",
487     "parameters": {
488         "type": "object",
489         "properties": {
490             "expression": {
491                 "type": "string",
492             },
493         },
494     },
495 }
496 # Define the tool schema
497 # Define the tool schema
498 tool_schema = {
499     "name": "get_weather_condition",
500     "description": "Get the weather information for a given location",
501     "parameters": {
502         "type": "object",
503         "properties": {
504             "location": {
505                 "type": "string",
506             },
507         },
508     },
509 }
510 # Define the tool schema
511 # Define the tool schema
512 tool_schema = {
513     "name": "calculate",
514     "description": "Calculate the result of a mathematical expression",
515     "parameters": {
516         "type": "object",
517         "properties": {
518             "expression": {
519                 "type": "string",
520             },
521         },
522     },
523 }
524 # Define the tool schema
525 # Define the tool schema
526 tool_schema = {
527     "name": "get_weather_condition",
528     "description": "Get the weather information for a given location",
529     "parameters": {
530         "type": "object",
531         "properties": {
532             "location": {
533                 "type": "string",
534             },
535         },
536     },
537 }
538 # Define the tool schema
539 # Define the tool schema
540 tool_schema = {
541     "name": "calculate",
542     "description": "Calculate the result of a mathematical expression",
543     "parameters": {
544         "type": "object",
545         "properties": {
546             "expression": {
547                 "type": "string",
548             },
549         },
550     },
551 }
552 # Define the tool schema
553 # Define the tool schema
554 tool_schema = {
555     "name": "get_weather_condition",
556     "description": "Get the weather information for a given location",
557     "parameters": {
558         "type": "object",
559         "properties": {
560             "location": {
561                 "type": "string",
562             },
563         },
564     },
565 }
566 # Define the tool schema
567 # Define the tool schema
568 tool_schema = {
569     "name": "calculate",
570     "description": "Calculate the result of a mathematical expression",
571     "parameters": {
572         "type": "object",
573         "properties": {
574             "expression": {
575                 "type": "string",
576             },
577         },
578     },
579 }
580 # Define the tool schema
581 # Define the tool schema
582 tool_schema = {
583     "name": "get_weather_condition",
584     "description": "Get the weather information for a given location",
585     "parameters": {
586         "type": "object",
587         "properties": {
588             "location": {
589                 "type": "string",
590             },
591         },
592     },
593 }
594 # Define the tool schema
595 # Define the tool schema
596 tool_schema = {
597     "name": "calculate",
598     "description": "Calculate the result of a mathematical expression",
599     "parameters": {
600         "type": "object",
601         "properties": {
602             "expression": {
603                 "type": "string",
604             },
605         },
606     },
607 }
608 # Define the tool schema
609 # Define the tool schema
610 tool_schema = {
611     "name": "get_weather_condition",
612     "description": "Get the weather information for a given location",
613     "parameters": {
614         "type": "object",
615         "properties": {
616             "location": {
617                 "type": "string",
618             },
619         },
620     },
621 }
622 # Define the tool schema
623 # Define the tool schema
624 tool_schema = {
625     "name": "calculate",
626     "description": "Calculate the result of a mathematical expression",
627     "parameters": {
628         "type": "object",
629         "properties": {
630             "expression": {
631                 "type": "string",
632             },
633         },
634     },
635 }
636 # Define the tool schema
637 # Define the tool schema
638 tool_schema = {
639     "name": "get_weather_condition",
640     "description": "Get the weather information for a given location",
641     "parameters": {
642         "type": "object",
643         "properties": {
644             "location": {
645                 "type": "string",
646             },
647         },
648     },
649 }
650 # Define the tool schema
651 # Define the tool schema
652 tool_schema = {
653     "name": "calculate",
654     "description": "Calculate the result of a mathematical expression",
655     "parameters": {
656         "type": "object",
657         "properties": {
658             "expression": {
659                 "type": "string",
660             },
661         },
662     },
663 }
664 # Define the tool schema
665 # Define the tool schema
666 tool_schema = {
667     "name": "get_weather_condition",
668     "description": "Get the weather information for a given location",
669     "parameters": {
670         "type": "object",
671         "properties": {
672             "location": {
673                 "type": "string",
674             },
675         },
676     },
677 }
678 # Define the tool schema
679 # Define the tool schema
680 tool_schema = {
681     "name": "calculate",
682     "description": "Calculate the result of a mathematical expression",
683     "parameters": {
684         "type": "object",
685         "properties": {
686             "expression": {
687                 "type": "string",
688             },
689         },
690     },
691 }
692 # Define the tool schema
693 # Define the tool schema
694 tool_schema = {
695     "name": "get_weather_condition",
696     "description": "Get the weather information for a given location",
697     "parameters": {
698         "type": "object",
699         "properties": {
700             "location": {
701                 "type": "string",
702             },
703         },
704     },
705 }
706 # Define the tool schema
707 # Define the tool schema
708 tool_schema = {
709     "name": "calculate",
710     "description": "Calculate the result of a mathematical expression",
711     "parameters": {
712         "type": "object",
713         "properties": {
714             "expression": {
715                 "type": "string",
716             },
717         },
718     },
719 }
720 # Define the tool schema
721 # Define the tool schema
722 tool_schema = {
723     "name": "get_weather_condition",
724     "description": "Get the weather information for a given location",
725     "parameters": {
726         "type": "object",
727         "properties": {
728             "location": {
729                 "type": "string",
730             },
731         },
732     },
733 }
734 # Define the tool schema
735 # Define the tool schema
736 tool_schema = {
737     "name": "calculate",
738     "description": "Calculate the result of a mathematical expression",
739     "parameters": {
740         "type": "object",
741         "properties": {
742             "expression": {
743                 "type": "string",
744             },
745         },
746     },
747 }
748 # Define the tool schema
749 # Define the tool schema
750 tool_schema = {
751     "name": "get_weather_condition",
752     "description": "Get the weather information for a given location",
753     "parameters": {
754         "type": "object",
755         "properties": {
756             "location": {
757                 "type": "string",
758             },
759         },
760     },
761 }
762 # Define the tool schema
763 # Define the tool schema
764 tool_schema = {
765     "name": "calculate",
766     "description": "Calculate the result of a mathematical expression",
767     "parameters": {
768         "type": "object",
769         "properties": {
770             "expression": {
771                 "type": "string",
772             },
773         },
774     },
775 }
776 # Define the tool schema
777 # Define the tool schema
778 tool_schema = {
779     "name": "get_weather_condition",
780     "description": "Get the weather information for a given location",
781     "parameters": {
782         "type": "object",
783         "properties": {
784             "location": {
785                 "type": "string",
786             },
787         },
788     },
789 }
790 # Define the tool schema
791 # Define the tool schema
792 tool_schema = {
793     "name": "calculate",
794     "description": "Calculate the result of a mathematical expression",
795     "parameters": {
796         "type": "object",
797         "properties": {
798             "expression": {
799                 "type": "string",
800             },
801         },
802     },
803 }
804 # Define the tool schema
805 # Define the tool schema
806 tool_schema = {
807     "name": "get_weather_condition",
808     "description": "Get the weather information for a given location",
809     "parameters": {
810         "type": "object",
811         "properties": {
812             "location": {
813                 "type": "string",
814             },
815         },
816     },
817 }
818 # Define the tool schema
819 # Define the tool schema
820 tool_schema = {
821     "name": "calculate",
822     "description": "Calculate the result of a mathematical expression",
823     "parameters": {
824         "type": "object",
825         "properties": {
826             "expression": {
827                 "type": "string",
828             },
829         },
830     },
831 }
832 # Define the tool schema
833 # Define the tool schema
834 tool_schema = {
835     "name": "get_weather_condition",
836     "description": "Get the weather information for a given location",
837     "parameters": {
838         "type": "object",
839         "properties": {
840             "location": {
841                 "type": "string",
842             },
843         },
844     },
845 }
846 # Define the tool schema
847 # Define the tool schema
848 tool_schema = {
849     "name": "calculate",
850     "description": "Calculate the result of a mathematical expression",
851     "parameters": {
852         "type": "object",
853         "properties": {
854             "expression": {
855                 "type": "string",
856             },
857         },
858     },
859 }
860 # Define the tool schema
861 # Define the tool schema
862 tool_schema = {
863     "name": "get_weather_condition",
864     "description": "Get the weather information for a given location",
865     "parameters": {
866         "type": "object",
867         "properties": {
868             "location": {
869                 "type": "string",
870             },
871         },
872     },
873 }
874 # Define the tool schema
875 # Define the tool schema
876 tool_schema = {
877     "name": "calculate",
878     "description": "Calculate the result of a mathematical expression",
879     "parameters": {
880         "type": "object",
881         "properties": {
882             "expression": {
883                 "type": "string",
884             },
885         },
886     },
887 }
888 # Define the tool schema
889 # Define the tool schema
890 tool_schema = {
891     "name": "get_weather_condition",
892     "description": "Get the weather information for a given location",
893     "parameters": {
894         "type": "object",
895         "properties": {
896             "location": {
897                 "type": "string",
898             },
899         },
900     },
901 }
902 # Define the tool schema
903 # Define the tool schema
904 tool_schema = {
905     "name": "calculate",
906     "description": "Calculate the result of a mathematical expression",
907     "parameters": {
908         "type": "object",
909         "properties": {
910             "expression": {
911                 "type": "string",
912             },
913         },
914     },
915 }
916 # Define the tool schema
917 # Define the tool schema
918 tool_schema = {
919     "name": "get_weather_condition",
920     "description": "Get the weather information for a given location",
921     "parameters": {
922         "type": "object",
923         "properties": {
924             "location": {
925                 "type": "string",
926             },
927         },
928     },
929 }
930 # Define the tool schema
931 # Define the tool schema
932 tool_schema = {
933     "name": "calculate",
934     "description": "Calculate the result of a mathematical expression",
935     "parameters": {
936         "type": "object",
937         "properties": {
938             "expression": {
939                 "type": "string",
940             },
941         },
942     },
943 }
944 # Define the tool schema
945 # Define the tool schema
946 tool_schema = {
947     "name": "get_weather_condition",
948     "description": "Get the weather information for a given location",
949     "parameters": {
950         "type": "object",
951         "properties": {
952             "location": {
953                 "type": "string",
954             },
955         },
956     },
957 }
958 # Define the tool schema
959 # Define the tool schema
960 tool_schema = {
961     "name": "calculate",
962     "description": "Calculate the result of a mathematical expression",
963     "parameters": {
964         "type": "object",
965         "properties": {
966             "expression": {
967                 "type": "string",
968             },
969         },
970     },
971 }
972 # Define the tool schema
973 # Define the tool schema
974 tool_schema = {
975     "name": "get_weather_condition",
976     "description": "Get the weather information for a given location",
977     "parameters": {
978         "type": "object",
979         "properties": {
980             "location": {
981                 "type": "string",
982             },
983         },
984     },
985 }
986 # Define the tool schema
987 # Define the tool schema
988 tool_schema = {
989     "name": "calculate",
990     "description": "Calculate the result of a mathematical expression",
991     "parameters": {
992         "type": "object",
993         "properties": {
994             "expression": {
995                 "type": "string",
996             },
997         },
998     },
999 }
1000 # Define the tool schema
1001 # Define the tool schema
1002 tool_schema = {
1003     "name": "get_weather_condition",
1004     "description": "Get the weather information for a given location",
1005     "parameters": {
1006         "type": "object",
1007         "properties": {
1008             "location": {
1009                 "type": "string",
1010             },
1011         },
1012     },
1013 }
1014 # Define the tool schema
1015 # Define the tool schema
1016 tool_schema = {
1017     "name": "calculate",
1018     "description": "Calculate the result of a mathematical expression",
1019     "parameters": {
1020         "type": "object",
1021         "properties": {
1022             "expression": {
1023                 "type": "string",
1024             },
1025         },
1026     },
1027 }
1028 # Define the tool schema
1029 # Define the tool schema
1030 tool_schema = {
1031     "name": "get_weather_condition",
1032     "description": "Get the weather information for a given location",
1033     "parameters": {
1034         "type": "object",
1035         "properties": {
1036             "location": {
1037                 "type": "string",
1038             },
1039         },
1040     },
1041 }
1042 # Define the tool schema
1043 # Define the tool schema
1044 tool_schema = {
1045     "name": "calculate",
1046     "description": "Calculate the result of a mathematical expression",
1047     "parameters": {
1048         "type": "object",
1049         "properties": {
1050             "expression": {
1051                 "type": "string",
1052             },
1053         },
1054     },
1055 }
1056 # Define the tool schema
1057 # Define the tool schema
1058 tool_schema = {
1059     "name": "get_weather_condition",
1060     "description": "Get the weather information for a given location",
1061     "parameters": {
1062         "type": "object",
1063         "properties": {
1064             "location": {
1065                 "type": "string",
1066             },
1067         },
1068     },
1069 }
1070 # Define the tool schema
1071 # Define the tool schema
1072 tool_schema = {
1073     "name": "calculate",
1074     "description": "Calculate the result of a mathematical expression",
1075     "parameters": {
1076         "type": "object",
1077         "properties": {
1078             "expression": {
1079                 "type": "string",
1080             },
1081         },
1082     },
1083 }
1084 # Define the tool schema
1085 # Define the tool schema
1086 tool_schema = {
1087     "name": "get_weather_condition",
1088     "description": "Get the weather information for a given location",
1089     "parameters": {
1090         "type": "object",
1091         "properties": {
1092             "location": {
1093                 "type": "string",
1094             },
1095         },
1096     },
1097 }
1098 # Define the tool schema
1099 # Define the tool schema
1100 tool_schema = {
1101     "name": "calculate",
1102     "description": "Calculate the result of a mathematical expression",
1103     "parameters": {
1104         "type": "object",
1105         "properties": {
1106             "expression": {
1107                 "type": "string",
1108             },
1109         },
1110     },
1111 }
1112 # Define the tool schema
1113 # Define the tool schema
1114 tool_schema = {
1115     "name": "get_weather_condition",
1116     "description": "Get the weather information for a given location",
1117     "parameters": {
1118         "type": "object",
1119         "properties": {
1120             "location": {
1121                 "type": "string",
1122             },
1123         },
1124     },
1125 }
1126 # Define the tool schema
1127 # Define the tool schema
1128 tool_schema = {
1129     "name": "calculate",
1130     "description": "Calculate the result of a mathematical expression",
1131     "parameters": {
1132         "type": "object",
1133         "properties": {
1134             "expression": {
1135                 "type": "string",
1136             },
1137         },
1138     },
1139 }
1140 # Define the tool schema
1141 # Define the tool schema
1142 tool_schema = {
1143     "name": "get_weather_condition",
1144     "description": "Get the weather information for a given location",
1145     "parameters": {
1146         "type": "object",
1147         "properties": {
1148             "location": {
1149                 "type": "string",
1150             },
1151         },
1152     },
1153 }
1154 # Define the tool schema
1155 # Define the tool schema
1156 tool_schema = {
1157     "name": "calculate",
1158     "description": "Calculate the result of a mathematical expression",
1159     "parameters": {
1160         "type": "object",
1161         "properties": {
1162             "expression": {
1163                 "type": "string",
1164             },
1165         },
1166     },
1167 }
1168 # Define the tool schema
1169 # Define the tool schema
1170 tool_schema = {
1171     "name": "get_weather_condition",
1172     "description": "Get the weather information for a given location",
1173     "parameters": {
1174         "type": "object",
1175         "properties": {
1176             "location": {
1177                 "type": "string",
1178             },
1179         },
1180     },
1181 }
1182 # Define the tool schema
1183 # Define the tool schema
1184 tool_schema = {
1185     "name": "calculate",
1186     "description": "Calculate the result of a mathematical expression",
1187     "parameters": {
1188         "type": "object",
1189         "properties": {
1190             "expression": {
1191                 "type": "string",
1192             },
1193         },
1194     },
1195 }
1196 # Define the tool schema
1197 # Define the tool schema
1198 tool_schema = {
1199     "name": "get_weather_condition",
1200     "description": "Get the weather information for a given location",
1201     "parameters": {
1202         "type": "object",
1203         "properties": {
1204             "location": {
1205                 "type": "string",
1206             },
1207         },
1208     },
1209 }
1210 # Define the tool schema
1211 # Define the tool schema
1212 tool_schema = {
1213     "name": "calculate",
1214     "description": "Calculate the result of a mathematical expression",
1215     "parameters": {
1216         "type": "object",
1217         "properties": {
1218             "expression": {
1219                 "type": "string",
1220             },
1221         },
1222     },
1223 }
1224 # Define the tool schema
1225 # Define the tool schema
1226 tool_schema = {
1227     "name": "get_weather_condition",
1228     "description": "Get the weather information for a given location",
1229     "parameters": {
1230         "type": "object",
1231         "properties": {
1232             "location": {
1233                 "type": "string",
1234             },
1235         },
1236     },
1237 }
1238 # Define the tool schema
1239 # Define the tool schema
1240 tool_schema = {
1241     "name": "calculate",
1242     "description": "Calculate the result of a mathematical expression",
1243     "parameters": {
1244         "type": "object",
1245         "properties": {
1246             "expression": {
1247                 "type": "string",
1248             },
1249         },
1250     },
1251 }
1252 # Define the tool schema
1253 # Define the tool schema
1254 tool_schema = {
1255     "name": "get_weather_condition",
1256     "description": "Get the weather information for a given location",
1257     "parameters": {
1258         "type": "object",
1259         "properties": {
1260             "location": {
1261                 "type": "string",
1262             },
1263         },
1264     },
1265 }
1266 # Define the tool schema
1267 # Define the tool schema
1268 tool_schema = {
1269     "name": "calculate",
1270     "description": "Calculate
```

For more information and examples of working with structured outputs using Groq API and Instructor, see our Groq API Cookbook tutorial [here](#).

```
203 available_function = {
204     "role": "system",
205     "content": "You are a helpful assistant.",
206 }
207 # Add the tool to the system prompt
208 available_function["content"] += "\n\n" + tool_description
209 # Add the tool to the system prompt
210 available_function["content"] += "\n\n" + tool_description
211 # Add the tool to the system prompt
212 available_function["content"] += "\n\n" + tool_description
213 # Add the tool to the system prompt
214 available_function["content"] += "\n\n" + tool_description
215 client = OpenAI(api_key="sk-1234567890", base_url="https://api.groq.com", mode="structured_output")
```

Best Practices

- Provide detailed tool descriptions for optimal performance.
 - We recommend tool use with the Instructor library for structured outputs.
 - Use the fine-tuned Llama 3 models by Groq or the Llama 3.1 models for your applications that require tool use.
 - Implement a routing system when using fine-tuned models in your workflow.
 - Handle tool execution errors by returning error messages with "is_error": true.
- ```
216 def process_query(query):
217 """Process the query and return the response using the appropriate model"""
218 # Route the query to the appropriate tool
219 route = route_query(query)
220 # Execute the tool
221 function_response = run_with_tool(query)
222 # Append the tool response to the messages
223 messages.append(
224 {
225 "role": "assistant",
226 "content": tool_response,
227 "tool_call_id": tool_call_id,
228 "route": route,
229 }
230)
231 # Make the final response
232 response = client.chat.completions.create(
233 model="llama3-70b-8192",
234 messages=messages,
235 tools=tools,
236 tool_choice="auto",
237 max_completion_tokens=4096,
238 temperature=0.7,
239)
240 # Extract the response
241 response_text = response.choices[0].message.content
242 # Print the response
243 print(f"Response: {response_text}")
244 # Print the tool calls
245 for call in tool_calls:
246 print(f"Input: {call.input_text}")
247 print(f"Tool: {call.tool_name}")
248 print(f"Parameters: {call.tool_parameters}")
249 print()
```