

Function calling - Connect ChatGPT to the Internet

Ashish Tiwari
Senior Developer Advocate



Challenges for Developers

Challenges for Developers

LLMs Challenges

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)
- Security and Privacy

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)
- Security and Privacy
- Inconsistent response

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)
- Security and Privacy
- Inconsistent response

Common challenges

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)
- Security and Privacy
- Inconsistent response

Common challenges

- Decision making on Nantural Language Query

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)
- Security and Privacy
- Inconsistent response

Common challenges

- Decision making on Nantural Language Query
- Executing correct componenet / code / function according to NL Query

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)
- Security and Privacy
- Inconsistent response

Common challenges

- Decision making on Nantural Language Query
- Executing correct componenet / code / function according to NL Query

Challenges for Developers

LLMs Challenges

- Connecting private data (RAGs, Fine tuning)
- Security and Privacy
- Inconsistent response

Common challenges

- Decision making on Nantural Language Query
- Executing correct componenet / code / function according to NL Query

Function calling

Powered by  OpenAI

Function calling

Powered by  OpenAI

Function calling in OpenAI refers to the capability of AI models to interact with external functions or APIs, allowing them to perform tasks beyond text generation.

fetch_from_elasticsearch(nl_query):

- Accepts query in natural language (e.g. Average delay time of flights going to India?)
- Convert Query into Elasticsearch Query DSL
- Execute Query on Index

```
def fetch_from_elasticsearch(nl_query):  
  
    query_dsl = build_query(nl_query)  
  
    # Execute query_dsl on Elasticsearch  
    ...  
    ...  
    ...  
    json_resp = json.dumps(resp, indent=4)  
  
    return json_resp
```

fetch_from_elasticsearch(nl_query):

- Accepts query in natural language (e.g. Average delay time of flights going to India?)
- Convert Query into Elasticsearch Query DSL
- Execute Query on Index

```
def build_query(nl_query):
    index_mapping = get_index_mapping()
    ref_document = get_ref_document()
    prompt = f"""
        Use below index mapping and reference document to build Elasticsearch query:

        Index mapping:
        {index_mapping}

        Reference elasticsearch document:
        {ref_document}

        Return single line Elasticsearch Query DSL according to index mapping for the below search query:
        {nl_query}

        few example of Query DSL
        {few_shots_prompt}
    """
```

fetch_from_elasticsearch(nl_query):

- Accepts query in natural language (e.g. Average delay time of flights going to India?)
- Convert Query into Elasticsearch Query DSL
- Execute Query on Index

```
def get_index_mapping():  
  
    # Query on Elasticsearch to get mapping  
    ...  
    ...  
    ...  
    mapping = json.dumps(resp, indent=4)  
  
    return mapping
```


fetch_from_elasticsearch(nl_query):

- Accepts query in natural language (e.g. Average delay time of flights going to India?)
- Convert Query into Elasticsearch Query DSL
- Execute Query on Index

```
def get_ref_document():  
  
    # Query on Elasticsearch to one reference document from Index  
    ...  
    ...  
    ...  
  
    json_resp = json.dumps(resp["hits"]["hits"][0], indent=4)  
  
    return json_resp
```

weather_report(latitude, longitude)

- This function returns weather report in `json` .
- Accepts parameter latitude & longitude (e.g. "12.96","77.75")
- It calls Open-Meteo API to fetch weather report.

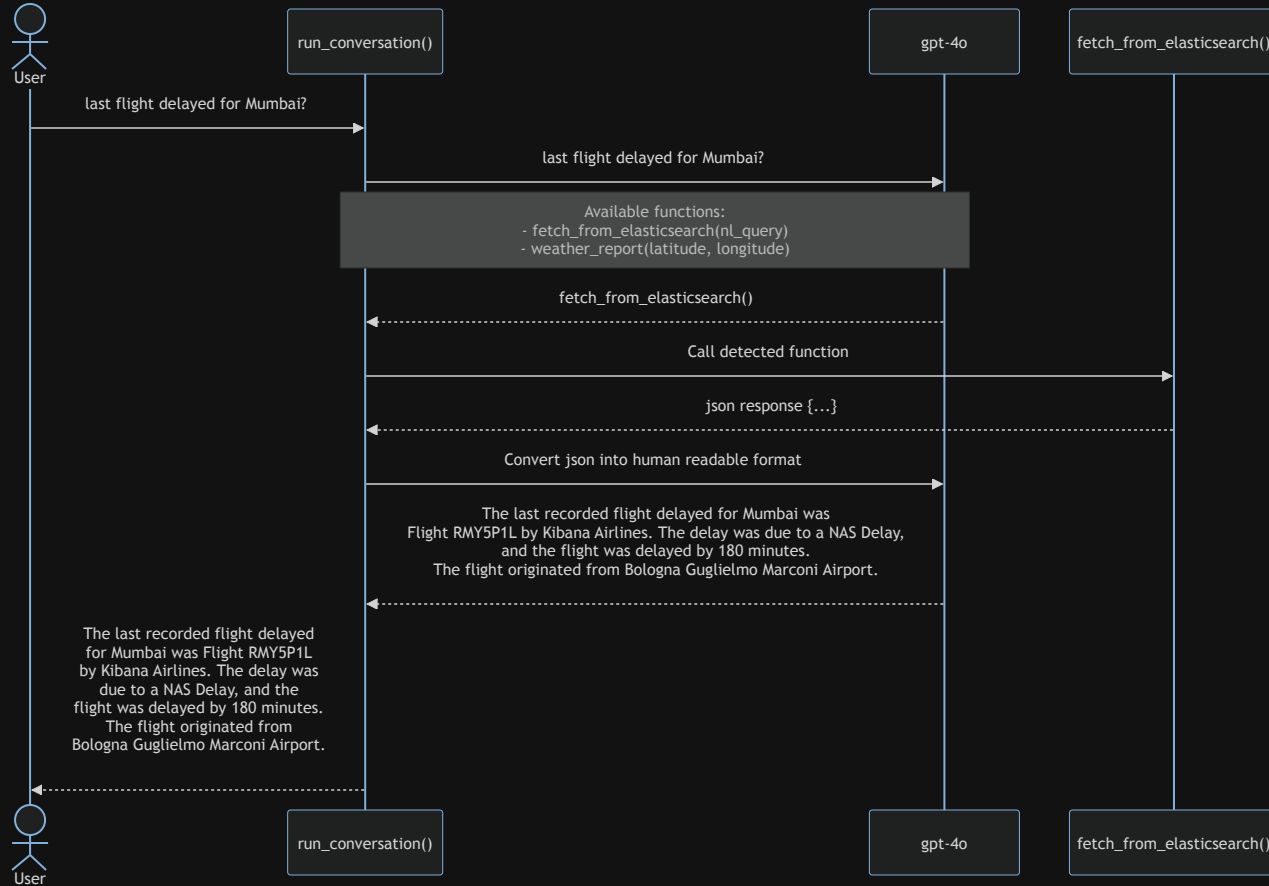
```
def weather_report(latitude, longitude):  
  
    url = f"{OPEN_METEO_ENDPOINT}?latitude={latitude}&longitude={longitude}&current=temperature_2m,precipitation,cloud_cover,"  
  
    resp = requests.request("GET", url)  
    resp = json.loads(resp.text)  
    json_resp = json.dumps(resp, indent=4)  
  
    return json_resp
```

weather_report(latitude, longitude)

- This function returns weather report in `json` .
- Accepts parameter latitude & longitude (e.g. "12.96","77.75")
- It calls Open-Meteo API to fetch weather report.

```
{
  "latitude": 19.125,
  "longitude": 72.875,
  .
  .
  .
},
"current": {
  "time": "2024-05-30T21:00",
  "interval": 900,
  "temperature_2m": 29.7,
  "precipitation": 0.0,
  "cloud_cover": 36,
  "visibility": 24140.0,
  "wind_speed_10m": 2.9
}
}
```

Flow



Parallel function calling

Parallel function calling is the model's ability to perform multiple function calls together

Only supported by latest OpenAI models - `gpt-4o` , `gpt-4-turbo` , and `gpt-3.5-turbo`

Query - How is the weather in Mumbai, and what about the last flight that got delayed there?

Demo (Notebook)



Function Calling Resources

Function calling <https://platform.openai.com/docs/guides/function-calling>

Notebook <https://github.com/elastic/elasticsearch-labs>


Gemini <https://ai.google.dev/gemini-api/docs/function-calling>

Mistral https://docs.mistral.ai/capabilities/function_calling/

Deck https://ashish.one/decks/function_calling.pdf

Thank You

 [in/ashishtiwari93](https://www.linkedin.com/in/ashishtiwari93)

 [@_ashish_tiwari](https://twitter.com/_ashish_tiwari)