Starting April 29, 2025, Gemini 1.5 Pro and Gemini 1.5 Flash models are not available in projects that have no prior usage of these models, including new projects. For details, see Model versions and lifecycle (/vertex-ai/generative-ai/docs/learn/model-versions#legacy-stable).

Function calling reference

Release Notes

Function calling improves the LLM's ability to provide relevant and contextual answers.

You can provide custom functions to a generative AI model with the Function Calling API. The model doesn't directly invoke these functions, but instead generates structured data output that specifies the function name and suggested arguments.

This output enables the calling of external APIs or information systems such as databases, customer relationship management systems, and document repositories. The resulting API output can be used by the LLM to improve response quality.

For more conceptual documentation on function calling, see Function calling (/vertex-ai/generative-ai/docs/multimodal/function-calling).

Supported models

- Gemini 2.5 Flash with Live API native Audio (/vertex-ai/generative-ai/docs/models/gemini/2-5-flash#live-api-native-audio)
- Gemini 2.0 Flash with Live API (/vertex-ai/generative-ai/docs/models/gemini/2-0-flash)
- Vertex AI Model Optimizer (/vertex-ai/generative-ai/docs/model-reference/vertex-ai-model-optimizer)
- Gemini 2.5 Pro (/vertex-ai/generative-ai/docs/models/gemini/2-5-pro)
- Gemini 2.5 Flash (/vertex-ai/generative-ai/docs/models/gemini/2-5-flash)
- Gemini 2.0 Flash (/vertex-ai/generative-ai/docs/models/gemini/2-0-flash)
- Gemini 2.0 Flash-Lite (/vertex-ai/generative-ai/docs/models/gemini/2-0-flash-lite)

Limitations:

- The maximum number of function declarations that can be provided with the request is 128.

Example syntax

Syntax to send a function call API request.

curl (#curl)

```
curl -X POST \
  -H "Authorization: Bearer $(gcloud auth print-access-token)" \
  -H "Content-Type: application/json" \

https://${LOCATION}-aiplatform.googleapis.com/v1/projects/${PROJECT_ID}/locatio
-d '{
  "contents": [{
    ...
  }],
  "tools": [{
    "function_declarations": [
      {
        ...
      }
    ]
  }]
}'
```

Parameter list

See examples (#examples) for implementation details.

`FunctionDeclaration`

Defines a function that the model can generate JSON inputs for based on OpenAPI 3.0 (https://spec.openapis.org/oas/v3.0.3) specifications.

Parameters

`name`	`string` The name of the function to call. Must start with a letter or an underscore. Must be a-z, A-Z, 0-9, or contains underscores, dots, or dashes, with a maximum length of 64.
`description`	Optional: `string` The description and purpose of the function. The model uses this to decide how and whether to call the function. For the best results, we recommend that you include a description.
`parameters`	Optional: `Schema` Describes the parameters of the function in the OpenAPI JSON Schema Object format: OpenAPI 3.0 specification (https://spec.openapis.org/oas/v3.0.3).
`response`	Optional: `Schema` Describes the output from the function in the OpenAPI JSON Schema Object format: OpenAPI 3.0 specification (https://spec.openapis.org/oas/v3.0.3).

For more information, see Function calling (/vertex-ai/generative-ai/docs/multimodal/function-calling)

`Schema`

Defines the format of the input and output data in a function call based on the OpenAPI 3.0 Schema (https://spec.openapis.org/oas/v3.0.3#schema) specification.

Parameters	
type	`string` Enum. The type of the data. Must be one of: • `STRING` • `INTEGER` • `BOOLEAN` • `NUMBER` • `ARRAY` • `OBJECT`

`description`	Optional: `string` Description of the data.
`enum`	Optional: `string[]` Possible values of the element of primitive type with enum format.
`items`	Optional: `Schema[]` Schema of the elements of `Type.ARRAY`
`properties`	Optional: `Schema` Schema of the properties of `Type.OBJECT`
`required`	Optional: `string[]` Required properties of `Type.OBJECT`.
`nullable`	Optional: `bool` Indicates if the value may be `null`.

`FunctionCallingConfig`

The `FunctionCallingConfig` controls the behavior of the model and determines what type of function to call.

Parameters	
`mode`	Optional: `enum/string[]` • `AUTO`: Default model behavior. The model can make predictions in either a function call form or a natural language response form. The model decides which form to use based on the context. • `NONE`: The model doesn't make any predictions in the form of function calls. • `ANY`: The model is constrained to always predict a function call. If `allowed_function_names` is not provided, the model picks from all of the available function declarations. If `allowed_function_names` is provided, the model picks from the set of allowed functions.
`allowed_function_names`	Optional: `string[]` Function names to call. Only set when the `mode` is `ANY`. Function names should match `[FunctionDeclaration.name]`. With mode set to `ANY`, the model will predict a function call from the set of function names provided.

`functionCall`

A predicted `functionCall` returned from the model that contains a string representing the `functionDeclaration.name` and a structured JSON object containing the parameters and their values.

Parameters	
`name`	`string` The name of the function to call.
`args`	`Struct` The function parameters and values in JSON object format. See Function calling (/vertex-ai/generative-ai/docs/model-reference/function-calling) for parameter details.

`functionResponse`

The resulting output from a `FunctionCall` that contains a string representing the `FunctionDeclaration.name`. Also contains a structured JSON object with the output from the function (and uses it as context for the model). This should contain the result of a `FunctionCall` made based on model prediction.

Parameters	
`name`	`string` The name of the function to call.
`response`	`Struct` The function response in JSON object format.

Examples

Send a function declaration

The following example is a basic example of sending a query and a function declaration to the model.

RESTGen AI SDK for Python... (#rest) Node.js (#node.js)Java (#java)Go (#go)REST (OpenAI) (#rest-openai)Pytho

Before using any of the request data, make the following replacements:

- *PROJECT_ID*: Your project ID (/resource-manager/docs/creating-managing-projects#identifiers).
- *MODEL_ID*: The ID of the model that's being processed.
- *ROLE*: The identity of the entity (/vertex-ai/generative-ai/docs/model-reference/inference#content) that creates the message.
- *TEXT*: The prompt to send to the model.
- *NAME*: The name of the function to call.
- *DESCRIPTION*: Description and purpose of the function.
- For other fields, see the Parameter list (#parameter-list) table.

HTTP method and URL:

```
POST https://aiplatform.googleapis.com/v1/projects/PROJECT_ID/locations/globa
```

Request JSON body:

```
{
  "contents": [{
    "role": "ROLE",
    "parts": [{
      "text": "TEXT"
    }]
  }],
  "tools": [{
    "function_declarations": [
      {
        "name": "NAME",
        "description": "DESCRIPTION",
          "parameters": {
            "type": "TYPE",
            "properties": {
              "location": {
                "type": "TYPE",
                "description": "DESCRIPTION"
              }
            },
            "required": [
              "location"
            ]
          }
        }
      ]
    }]
}
```

To send your request, choose one of these options:

curlPowerShell (#powershell)
(#curl)

★ **Note:** The following command assumes that you have logged in to the **`gcloud`** CLI with your user account by running **`gcloud init`** (/sdk/gcloud/reference/init) or **`gcloud auth login`** (/sdk/gcloud/reference/auth/login) , or by using Cloud Shell (/shell/docs), which automatically logs you into the **`gcloud`** CLI . You can check the currently active account by running **`gcloud auth list`** (/sdk/gcloud/reference/auth/list).

Save the request body in a file named `request.json`, and execute the following command:

```
curl -X POST \
     -H "Authorization: Bearer $(gcloud auth print-access-token)" \
     -H "Content-Type: application/json; charset=utf-8" \
     -d @request.json \
     "https://aiplatform.googleapis.com/v1/projects/PROJECT_ID/locatior
```

Example curl command

```
PROJECT_ID=myproject
LOCATION=us-central1
MODEL_ID=gemini-2.0-flash-001

curl -X POST \
  -H "Authorization: Bearer $(gcloud auth print-access-token)" \
  -H "Content-Type: application/json" \
  https://${LOCATION}-aiplatform.googleapis.com/v1/projects/${PROJECT_ID}/locat
  -d '{
    "contents": [{
      "role": "user",
      "parts": [{
        "text": "What is the weather in Boston?"
      }]
    }],
    "tools": [{
      "functionDeclarations": [
        {
          "name": "get_current_weather",
          "description": "Get the current weather in a given location",
          "parameters": {
            "type": "object",
            "properties": {
              "location": {
                "type": "string",
                "description": "The city and state, e.g. San Francisco, CA or a
              }
            },
            "required": [
              "location"
            ]
          }
        }
      ]
    }]
  }'
```

Send a function declaration with `FunctionCallingConfig`

The following example demonstrates how to pass a `FunctionCallingConfig` to the model.

The `functionCallingConfig` ensures that the model output is always a specific function call. To configure:

- Set the function calling `mode` to `ANY`.
- Specify the function names that you want to use in `allowed_function_names`. If `allowed_function_names` is empty, any of the provided functions can be returned.

REST (#rest) Gen AI SDK for Python... Node.js (#node.js) Go (#go) REST (OpenAI) (#rest-openai) Python (OpenAI).

```
PROJECT_ID=myproject
LOCATION=us-central1
MODEL_ID=gemini-2.0-flash-001

curl -X POST \
  -H "Authorization: Bearer $(gcloud auth print-access-token)" \
  -H "Content-Type: application/json" \
  https://${LOCATION}-aiplatform.googleapis.com/v1beta1/projects/${PROJECT_ID}/
  -d '{
    "contents": [{
      "role": "user",
      "parts": [{
        "text": "Do you have the White Pixel 8 Pro 128GB in stock in the US?"
      }]
    }],
    "tools": [{
      "functionDeclarations": [
        {
          "name": "get_product_sku",
          "description": "Get the available inventory for a Google products, e.
          "parameters": {
            "type": "object",
            "properties": {
              "product_name": {"type": "string", "description": "Product name"}
            }
          }
        },
        {
          "name": "get_store_location",
          "description": "Get the location of the closest store",
          "parameters": {
            "type": "object",
            "properties": {
              "location": {"type": "string", "description": "Location"}
            },
          }
        }
      ]
    }],
    "toolConfig": {
        "functionCallingConfig": {
            "mode":"ANY",
            "allowedFunctionNames": ["get_product_sku"]
      }
    },
    "generationConfig": {
      "temperature": 0.95,
      "topP": 1.0,
      "maxOutputTokens": 8192
    }
  }'
```

What's next

For detailed documentation, see the following:

- Function calling (/vertex-ai/generative-ai/docs/multimodal/function-calling)

Last updated 2025-06-06 UTC.