

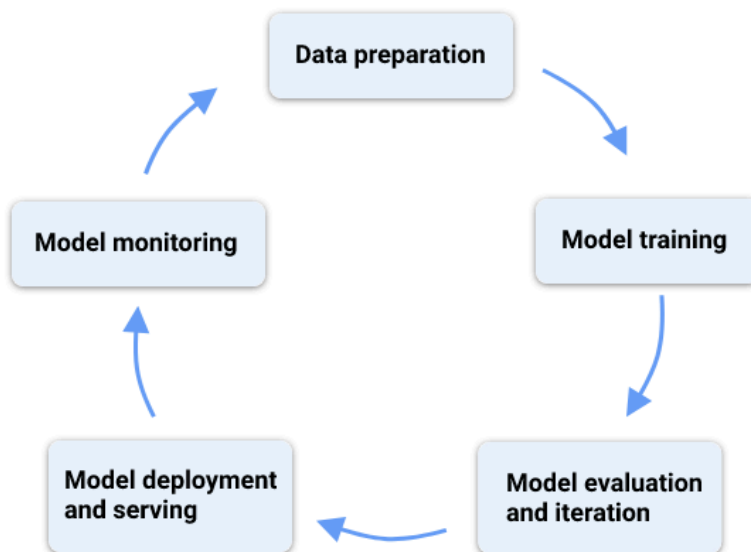
Agent Builder Advanced: Integrating External APIs

For section-specific guidelines and examples, see below

Vertex AI Overview

[Vertex AI](#) is a machine learning (ML) platform that lets you train and deploy ML models and AI applications. Vertex AI combines data engineering, data science, and ML engineering workflows, enabling your teams to collaborate using a common toolset. Agent Builder is a low-code, no-code platform that can help build things faster.

Machine learning workflow



What are LLMs

Large language models (LLMs) are deep learning models trained on massive datasets of text. LLMs can translate language, summarize text, generate creative writing, generate code, power chatbots and virtual assistants, and complement search engines and recommendation systems. Creating an LLM requires massive amounts of data, significant compute resources, and specialized skills. Because LLMs require a big investment to create, they target broad rather than specific use cases. On Vertex AI, you can customize a foundation model for more specific tasks or knowledge domains by using prompt design and model tuning.

Build powerful AI agents, no code required. For complex goals, you can easily stitch together multiple agents, with one agent functioning as the main agent and others as subagents. Train with your data, automate tasks, and iterate with ease. Launch and analyze - all within a user-friendly platform.

Objectives:

- ✓ Introduce Agent Builder
- ✓ Cloud Functions for Integrating with external apis
- ✓ Build a use-case with a no-code/low-code platform

Task 0. Setup and Requirements

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

What you need

To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.

Note: If you already have your own personal Google Cloud account or project, do not use it for this lab.

Note: If you are using a Chrome OS device, open an Incognito window to run this lab.

Alternatively, you can log out of all other Google / Gmail accounts before beginning the labs.





How to start your lab and sign in to the Google Cloud Console


1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.

[Open Google Console](#)

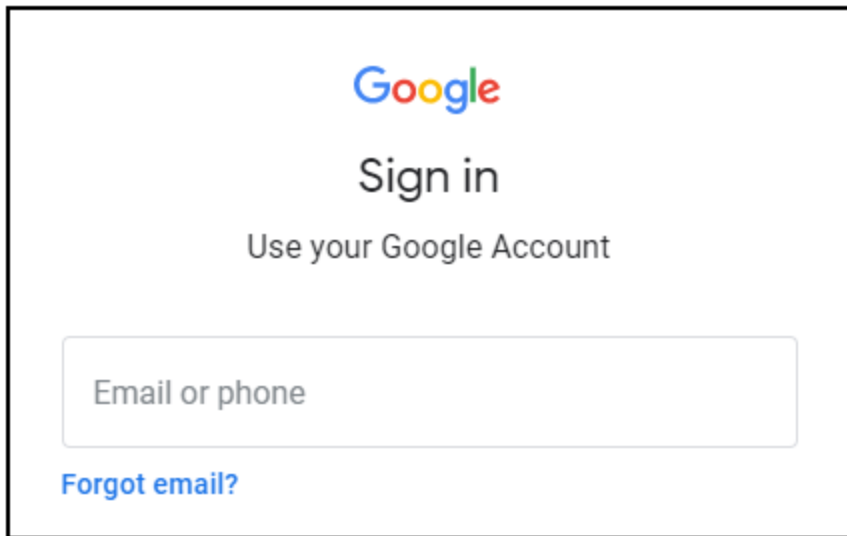
Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. [Learn more.](#)

Username
google2727032_student@qwiklabs.n 

Password
k68CZXsxMZ 

GCP Project ID
qwiklabs-gcp-4fbfecac8667e457 

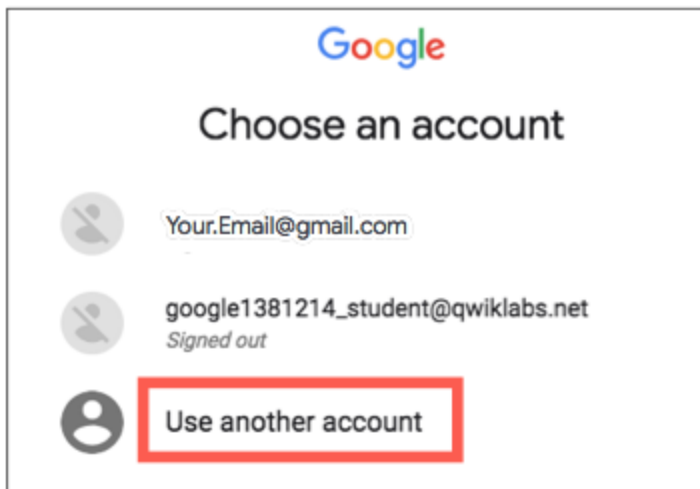
[New to labs? View our introductory video!](#)



2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.

Tip: Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another Account**.



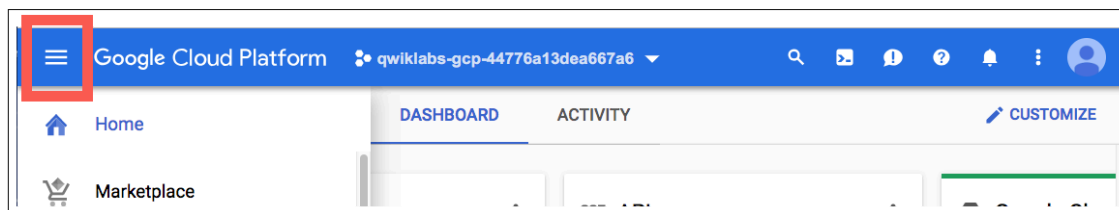
3. In the **Sign in** page, paste the username that you copied from the left panel. Then copy and paste the password.

Important: You must use the credentials from the left panel. Do not use your Google Cloud Training credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

4. Click through the subsequent pages:
 - Accept the terms and conditions.
 - Do not add recovery options or two-factor authentication (because this is a temporary account).
 - Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

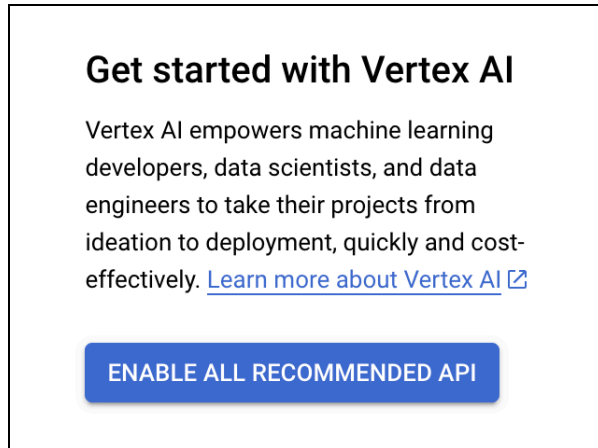
Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.



Task 1. Set up your environment

Enable the Vertex AI API

1. In the Google Cloud Console, on the **Navigation menu**, click **Vertex AI** or Utilize the search bar to find **Vertex AI**.
2. Within the "Get Started with Vertex AI" section, locate and click on the option that says



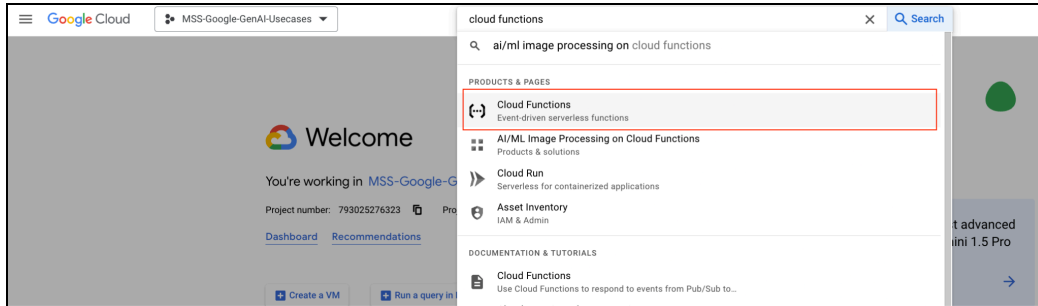
"ENABLE ALL RECOMMENDED API."

Task 2. Create a Cloud Function to Fetch Weather Data from Weatherbit.io

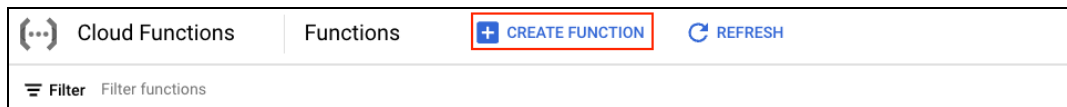
Open Google Cloud Console and create a new Cloud Function with name **get_current_weather**. Use Flask as the HTTP handler for the function's endpoint. Inside the function, make a call to the Weatherbit.io API to retrieve the current weather data.

Follow the steps below:

1. Search for the **Cloud Run Functions** in the GCP console, and select Cloud Functions from the search result



2. Create the Cloud Function
 - a. Click on CREATE FUNCTION



3. Update the parameters, and set the parameters as marked in **RED**.

Note: Function name is a name entered by the user

- a. Environment : **Cloud Run Function**
- b. Function name : **get_current_weather**
- c. Region : **us-central1 (Iowa)**
- d. Trigger type : **HTTPS**
- e. Select **“Allow unauthenticated invocations”**

Cloud Run functions

Create function

1 Configuration

2 Code

Basics

Environment

Cloud Run function

Function name *

get_current_weather

Region *

us-central1 (Iowa)

Trigger

Trigger type

HTTPS

URL

https://us-central1-fresh-span-400217.cloudfunctions.net/get_current_weather

Authentication

☒ Allow unauthenticated invocations

Check this if you are creating a public API or website.

☐ Require authentication

Manage authorized users with Cloud IAM.

Runtime, build, connections and security settings

NEXT

CANCEL

Note: Maximum numbers of instances should not exceed 5 (Refer below screenshot). This is the maximum number assigned for Qwiklabs.

Runtime, build, connections and security settings

< RUNTIME BUILD CONNECTIONS SECURITY AND >

Memory allocated * 256 MiB CPU * 0.167

Timeout * 60 seconds ?

Concurrency

Maximum concurrent requests per instance 1 ?

Autoscaling ?

Minimum number of instances 0

Maximum number of instances 5

4. Get the Weather API Key from Weatherbit.io

- Go to <https://www.weatherbit.io/>
- Click on Sign up to create a new account

Weatherbit

Features Pricing Weather APIs Resources Contact Us Log In Sign up

Weather API for Programmers

The High Performance Weather API for All of Your Data Needs.

Enhance your business with our global weather data API. Start with a free trial in just a few clicks!

Your email
gogheniak24@gmail.com

Choose a password

Create Account

Birmingham, MI, US

63°F

Wed, 10 mph
Precip: 0 in
Pressure: 1012 mb

Overcast Clouds

WED	THU	FRI	SAT	SUN	MON
67°F / 37°F	58°F / 41°F	59°F / 42°F	54°F / 34°F	55°F / 39°F	60°F / 56°F

- Enter the email id and password

Weatherbit

Features Pricing Weather API's Resources Contact Us Log in Sign up

Sign Up for the Weatherbit API

*Username:
ajaygogineni123

*Password:

*Repeat Password for Verification:

*Email:
g.ajaykumar7778@gmail.com

☒ Send me updates via email.

☐ I accept the [Terms and Conditions](#) and the [Privacy Policy](#).

☐ I'm not a robot

reCAPTCHA
Privacy - Terms

Submit

d. Accept the Terms and Conditions, Verify captcha and click on Submit

Weatherbit

Features Pricing Weather API's Resources Contact Us Log in Sign up

Sign Up for the Weatherbit API

*Username:
ajaygogineni123

*Password:

*Repeat Password for Verification:

*Email:
g.ajaykumar7778@gmail.com

☒ Send me updates via email.

☒ I accept the [Terms and Conditions](#) and the [Privacy Policy](#).

☒ I'm not a robot

reCAPTCHA
Privacy - Terms

Submit

e. Verify the email id by clicking on the link sent over email.

The screenshot shows the Weatherbit API User Login page. At the top is a dark blue navigation bar with the Weatherbit logo and links for Features, Pricing, Weather API's, Resources, Contact Us, Log in, and a green Sign up button. The main content area has a white background with a central card. The card is titled "API User Login" in bold. Below the title is a green checkmark icon followed by a message: "We have sent a confirmation link to your email. Please check your email, and click the link provided in it. If you did not receive that email, you can reset your password via the 'Forgot Password?' link below to access your account." Below this message are two input fields: "Username:" with a placeholder "Enter Email or Username" and "Password:" with a placeholder "Enter Password". Both fields are outlined with a red border. Below the password field is a green "Login" button. At the bottom of the card are two blue links: "Forgot Password" and "Forgot Username".

f. Login using the details after verifying the account

This screenshot shows the same Weatherbit API User Login page, but with the username field pre-filled with "ajaygogineni123" and the password field masked with "*****". The "Username:" label and the "Password:" label are still present. The green "Login" button remains below the password field. The "Forgot Password" and "Forgot Username" links are at the bottom of the card. The overall layout and navigation bar are identical to the previous screenshot.

g. You will be redirected to a page where you can select your plan

Welcome to Weatherbit.io!

Before you can access your free API key, we need just a few more things:

Company Name/Organization

Name*

Billing Currency*

Country*

API Usage Purpose*

API Tier* [Billing Terms and Conditions](#)

Plan Details

Plan: Free Trial (21 Days)
1,500 req/day
1,500 historical req/day
Current weather + alerts
25 years historical
16 day forecasts
240 hour forecasts
60 minute forecasts
+ Energy / Air Quality / Agweather / Climate Normals API
Non-Commercial use only
95.0% Uptime
Price: \$0/mo

[Proceed to Dashboard](#)

h. Fill the details. Select Free Trial for API Tier. Click on Proceed to Dashboard

Welcome to Weatherbit.io!

Before you can access your free API key, we need just a few more things:

Company Name/Organization

Name*

Billing Currency*

Country*

API Usage Purpose*

API Tier* [Billing Terms and Conditions](#)

Plan Details

Plan: Free Trial (21 Days)
1,500 req/day
1,500 historical req/day
Current weather + alerts
25 years historical
16 day forecasts
240 hour forecasts
60 minute forecasts
+ Energy / Air Quality / Agweather / Climate Normals API
Non-Commercial use only
95.0% Uptime
Price: \$0/mo

[Proceed to Dashboard](#)

i. You will be redirected to the Dashboard where you can get the API key

Weatherbit

Features Pricing Weather API's Resources Contact Us Dashboard

Dashboard

Manage Subscription

Change Email / Password

API Documentation

API Status

Support

Historical Forecast Order

Historical Data Order

Delete Account

Log Out

Plan	Details	Support Level
Business Trial (Expires 2024-11-13)	1,500 req/day 1,500 historical req/day 25 years historical Current weather + alerts 16 day forecasts 240 hour forecasts 60 minute forecasts + Energy / Air Quality / Agweather / Climate Normals API Non-Commercial use only	Basic Support Email Support 3 Business Day Response API Status Page Outage Notifications > 2 hours

API Key	Name
c7f1c97301dd46b98...	Master API Key

Contact Emails	Access
gogineniak24@gmail.com	Primary Login Email

Company Information

Business Name

Update

5. Add the API key as an environment variable in the Cloud Function

Expand Runtime, build, connections and security settings menu

Cloud Run functions

Edit function

1 Configuration

2 Code

✓ get_current_weather

Cloud Run function

us-central1

Trigger

Trigger type

HTTPS

URL

https://us-central1-fresh-span-400217.cloudfunctions.net/get_current_weather

Runtime, build, connections and security settings

NEXT

CANCEL

Go to Runtime environment variables

Add a variable with name WEATHER_API_KEY and value as the API KEY obtained from Weatherbit.io

Cloud Run functions

Edit function

Runtime, build, connections and security settings

<

RUNTIME

BUILD

CONNECTIONS

SECURITY AND

>

Memory allocated *

256 MiB

CPU *

0.167

Timeout *

60

seconds

?

Concurrency

Maximum concurrent requests per instance

1

?

Autoscaling

?

Minimum number of instances

0

Maximum number of instances

100

Runtime service account

?

Service account

Default compute service account

By default Cloud Functions uses the automatically created Default Compute Engine Service Account. [Learn more about service accounts.](#)

Runtime environment variables

?

Name 1 *

LOG_EXECUTION_ID

Value 1

true

Name 2 *

WEATHER_API_KEY

Value 2

c7f1c97301dd46b9883cdc31c04de

+ ADD VARIABLE

NEXT

CANCEL

6. Click on NEXT to open the Editor

Cloud Run functions

Create function

1 Configuration

2 Code

Basics

Environment

Cloud Run function

Function name *

get_current_weather

Region *

us-central1 (Iowa)

Trigger

Trigger type

HTTPS

URL

https://us-central1-fresh-span-400217.cloudfunctions.net/get_current_weather

Authentication

☒ Allow unauthenticated invocations

Check this if you are creating a public API or website.

☐ Require authentication

Manage authorized users with Cloud IAM.

Runtime, build, connections and security settings

NEXT

CANCEL

6. Update the code to point to a Flask endpoint in the Editor.
(Note that the Function name from Step 1, Entry point and the function name in main.py should be same)

- Select Python 3.12 in Runtime
- Update Entry point to get_current_weather
- Update the python code in main.py

This code gets weather through an API code

Code for [main.py](#):

Note: "Please verify code indentation after copying." You can copy the code in a text format from the github location: <https://github.com/themlguy-tf/generative-ai/tree/main/agent-builder>

```
from flask import Flask, request, make_response, jsonify
import requests
import os
app = Flask(__name__)

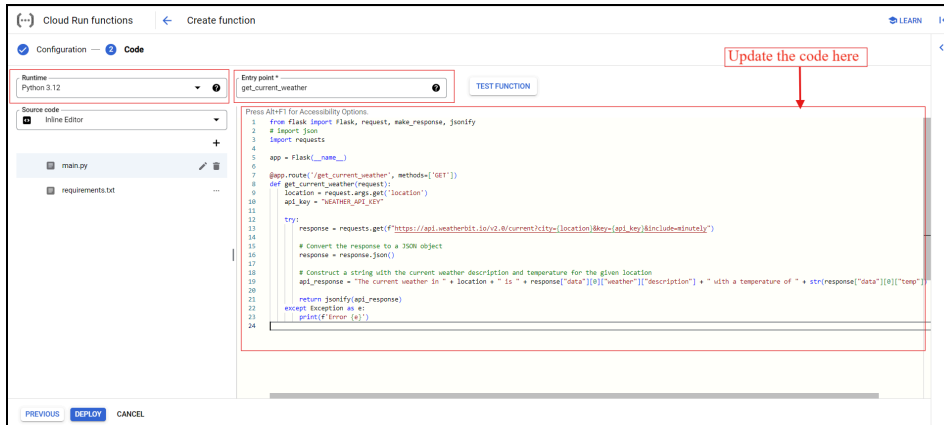
@app.route('/get_current_weather', methods=['GET'])
def get_current_weather(request):
    location = request.args.get('location')
    api_key = os.getenv('WEATHER_API_KEY')

    try:
        response =
requests.get(f"https://api.weatherbit.io/v2.0/current?city={location}&key={api_key}&include
=minutely")

        # Convert the response to a JSON object
        response = response.json()

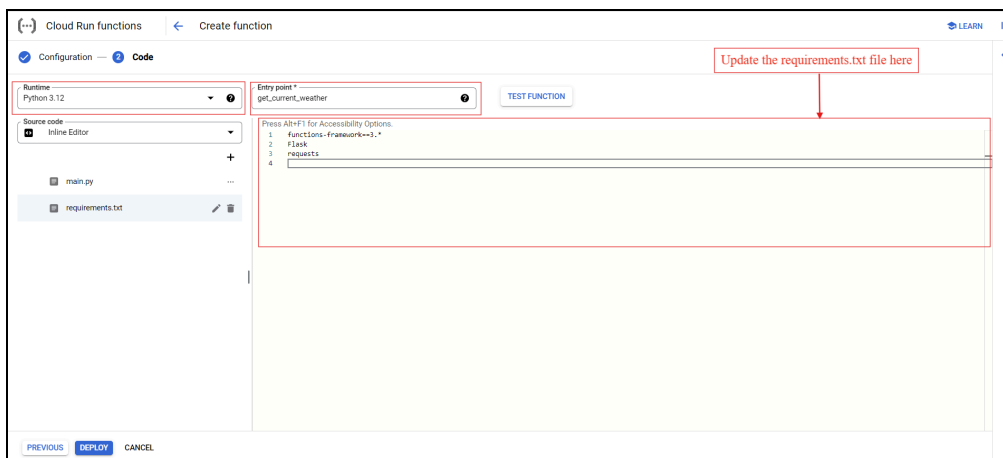
        # Construct a string with the current weather description and temperature for the given
location
        api_response = "The current weather in " + location + " is " +
response["data"][0]["weather"]["description"] + " with a temperature of " +
str(response["data"][0]["temp"]) + "°C."

        return jsonify(api_response)
    except Exception as e:
        print(f'Error {e}')
```

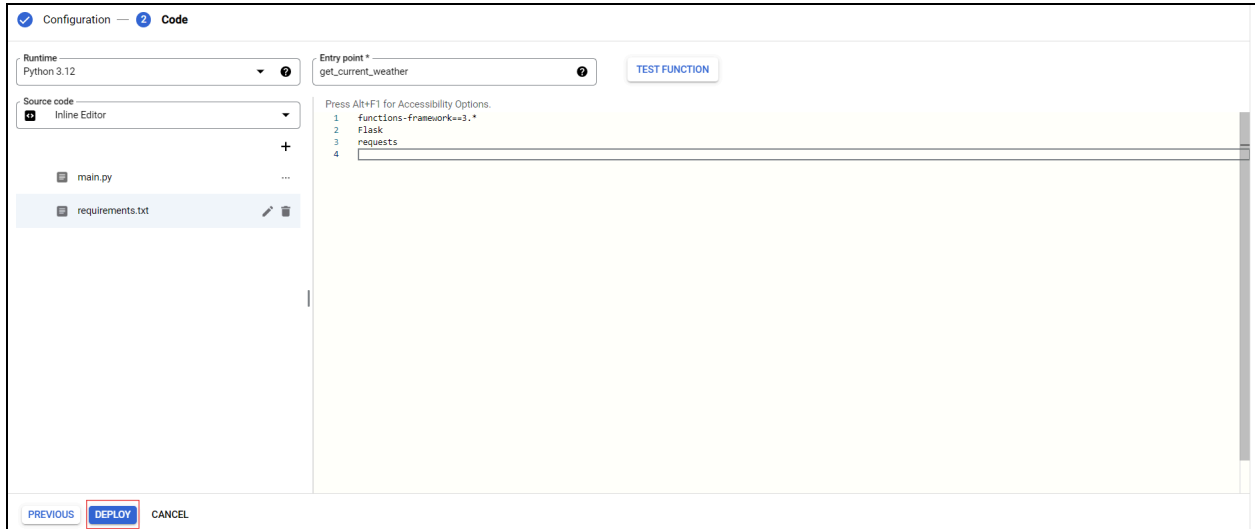


- d. Update the requirements.txt file
- e. requirements.txt

functions-framework==3.*
 Flask
 requests



6. Click on Deploy to deploy the cloud function

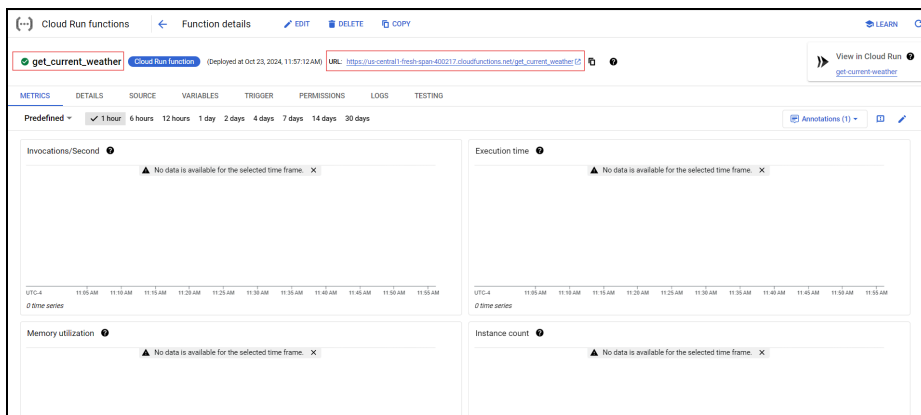


This will deploy the cloud function that can get weather from Weatherbit.io. The cloud function is deployed here:

https://us-central1-fresh-span-400217.cloudfunctions.net/get_current_weather

* Note: This url will be different for your application

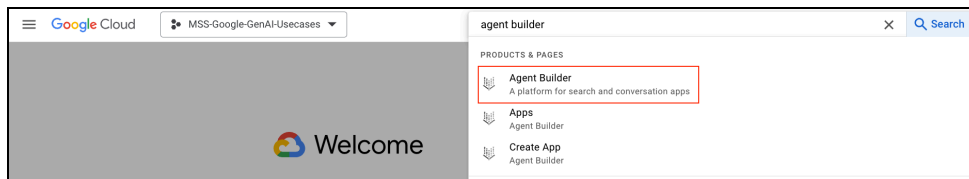
Debug Steps: Make sure the Entry point and function name are same



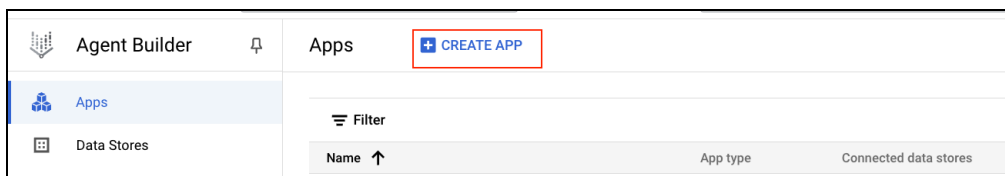
Task 3. Navigate to the Agent Builder service and set up an OpenAPI tool for the get_current_weather cloud function.

This tool will be used as an input to the agent. A description of the tool and the corresponding schema should be provided here. The output format of get_current_weather cloud function and the input of this tool should be the same. If there is a mismatch, there will be errors.

1. Go to Agent Builder on Google cloud console

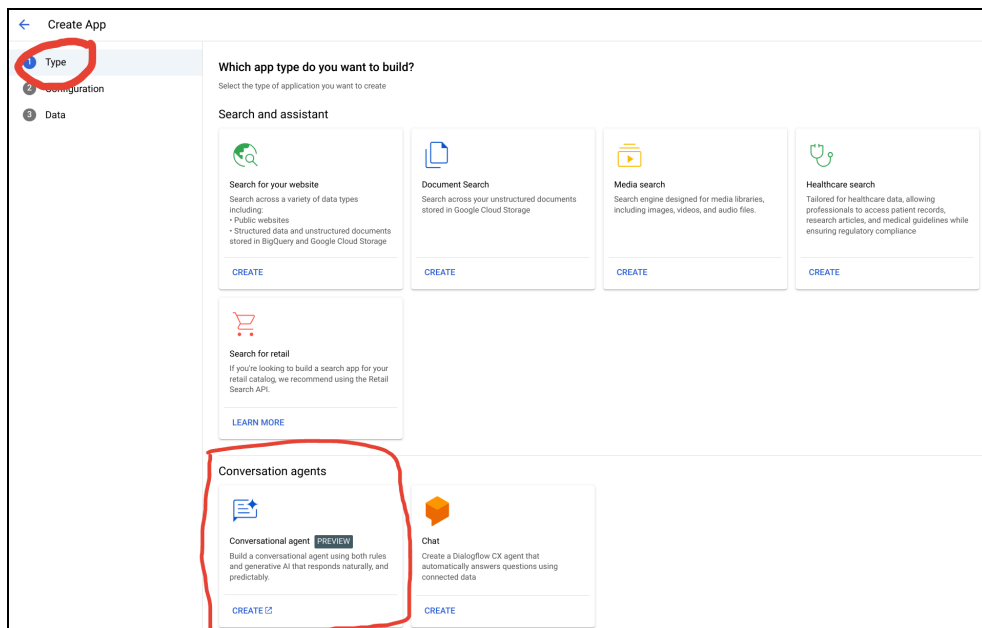


2. Click on Create App to start building a new Agent builder

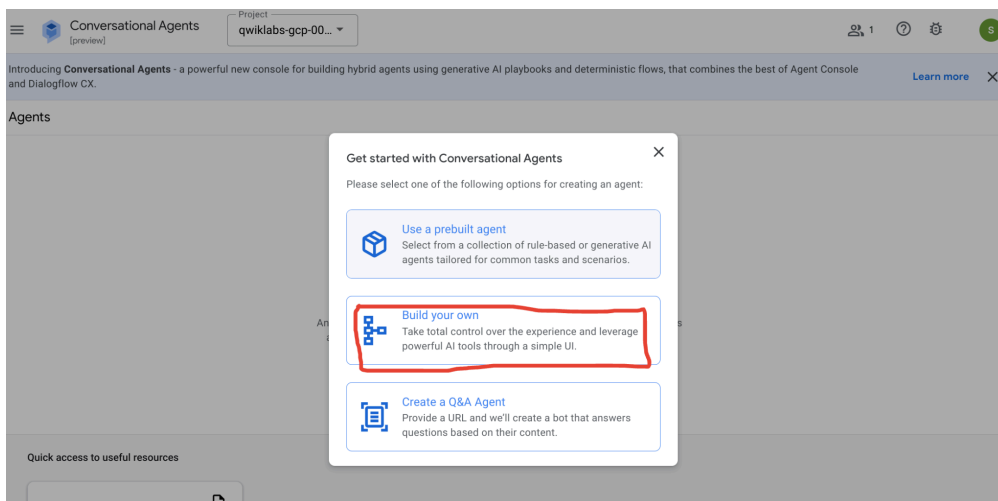


3. Select Apps > Type > Agent

In Create App, select the Conversation agent



Select “Build your own”



Click “Create”. Please note that it is a Playbook - Generative AI agent.

Update the Display name of the agent to **Weather Agent** and click on Create

The screenshot shows the 'Create agent' modal in the Google Cloud Conversational Agents console. The modal is titled 'Create agent' and contains the following fields and options:

- Display name***: A text input field with the value 'Agent'.
- Agent display name**: A label for the display name field.
- Location***: A dropdown menu showing 'us-central1 (Iowa, USA)' with a 'View' link.
- Time zone***: A dropdown menu showing '(GMT-8:00) America/Los_Angeles'.
- Default language***: A dropdown menu showing 'en - English'.
- Conversation start**: A section for defining the start of a conversation.

Below the fields, there is a note: 'You cannot change the location after creating the agent. Please use the global region for integration with Agent Assist or Insights, as other regions do not support this.' At the bottom right, there are 'Cancel' and 'Create' buttons.

The screenshot shows the 'Default Generative Playbook' configuration page in the Google Cloud Conversational Agents console. The page is titled 'Default Generative Playbook' and has tabs for 'Basics', 'Examples', and 'Settings'. The 'Basics' tab is active.

The page contains the following sections:

- High level description of the goal the playbook intends to accomplish**: A text input field with a 'Learn more' link.
- Instructions**: A section with a 'Sample' button and a list of instructions:
 - Greet the user, then ask how you can help them today.
 - Summarize the user's request and ask them to confirm that you understood correctly.
 - If necessary, seek clarifying details.
 - Use \${TOOL: Example tool name} to help the user with their task.
 - Use \${PLAYBOOK: Example playbook name} to help the user with a complex subtask.
 - Thank the user for their business and say goodbye.
- Available tools**: A section with a description: 'This playbook can use selected tools to generate responses. You can also call other tools, including 3P directly in steps. Create a data store tool to allow this playbook to answer questions using data store content.' Below this are '+ Data store' and 'Manage all tools' buttons.

4. Select Tools from the left panel

Agent Console

App
weather_agent

You are currently in the **upgraded version of the Agent Console (V2)**, which contains the latest bugs fixes and features. Feel free to report any bugs using the bug icon above. You can also switch back to V1 at any time via the banner link.

Agents

Tools

Conversation history

Integrations [preview]

Settings

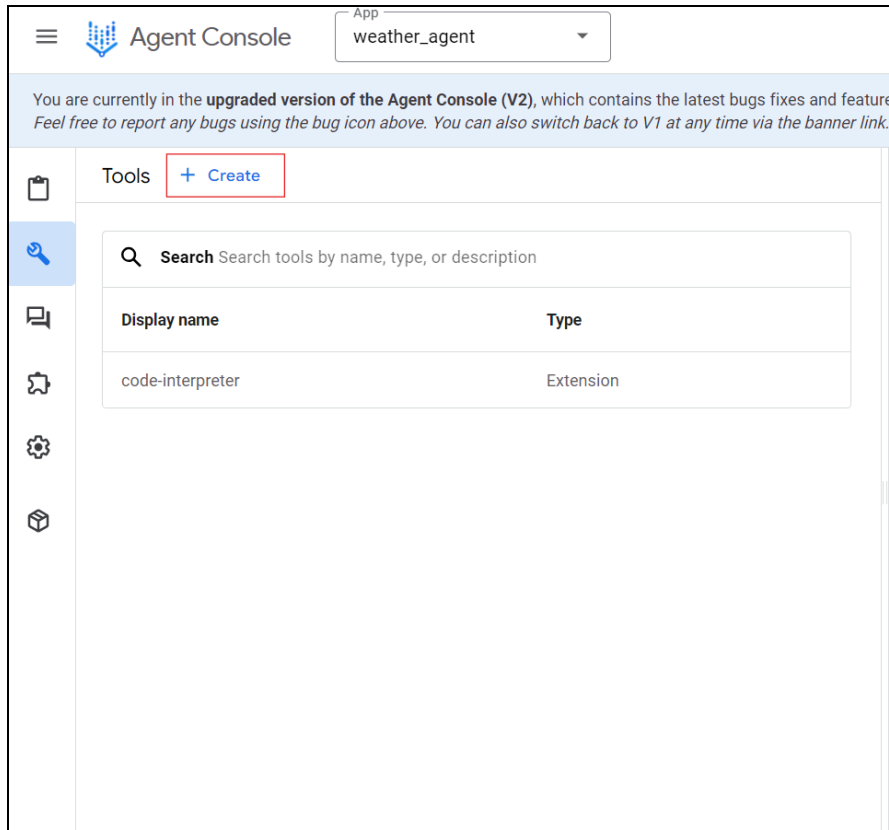
Prebuilt [preview]

by name, type, or description

Type

Extension

5. Click on Create



6. Update the following parameters:

- a. Tool name : get_current_weather
- b. Type : OpenAPI
- c. Update the Description to:
Use this tool to get the real time weather at a particular location. Pass the location as an input parameter to get the weather at that location.
- d. Select YAML under schema
- e. Update the schema to:
* Note: Use the url corresponding to get_current_weather cloud function created previously in the schema. Set the paths to get_current_weather.

Note: "Please verify code indentation after copying." You can copy the code in a text format from the github location: <https://github.com/themlquy-tf/generative-ai/tree/main/agent-builder>

openapi: 3.0.2

info:

title: Weather API

description: API to get the current weather based on location using Weatherbit.

version: 1.0.0

servers:

- url: 'https://us-central1-genaiinprogress.cloudfunctions.net'

description: Cloud Function Deployment

paths:

/get_current_weather:

get:

summary: Get Current Weather

description: Retrieve the current weather for a specified location.

parameters:

- in: query

name: location

schema:

type: string

required: true

description: The name of the city to get the weather for.

responses:

'200':

description: Successful response with weather information

content:

application/json:

schema:

type: object

properties:

weather:

type: string

example: >-

The current weather in London is Clear sky with a temperature of 13.3°C.

'400':

description: Bad Request (missing or invalid location parameter)

'500':

description: 'Internal Server Error (e.g., API issues or connectivity failure)'

components:

securitySchemes:

ApiKeyAuth:

type: apiKey

in: header

name: WEATHER_API_KEY
security:
- ApiKeyAuth: []

The screenshot shows the 'Tools' configuration page in the Agent Console. The tool name is 'get_current_weather' and the type is 'OpenAPI'. The description is 'Use this tool to get the real time weather at a particular location. Pass the location as an input parameter to get the weather at that location.' The schema is defined in YAML format, showing an OpenAPI 3.0.2 specification for a 'Weather API' that provides current weather based on location using Weatherbit. The schema is defined in YAML format, showing an OpenAPI 3.0.2 specification for a 'Weather API' that provides current weather based on location using Weatherbit. The schema is defined in YAML format, showing an OpenAPI 3.0.2 specification for a 'Weather API' that provides current weather based on location using Weatherbit.

Agent Console

App: weather_agent

You are currently in the **upgraded version of the Agent Console (V2)**, which contains the latest bugs fixes and features. Feel free to report any bugs using the bug icon above. You can also switch back to V1 at any time via the banner link.

Tools [Save](#)

Using tools, you can connect agents to external systems. These systems can augment the knowledge of agents and empower them to execute complex tasks efficiently. [Learn more](#)

Tool name*
get_current_weather

Type
OpenAPI

Connect to an external API using an OpenAPI tool.

Description
Provide a description of this tool. This description is provided to the model as context informing how the tool is used.

Description
Use this tool to get the real time weather at a particular location. Pass the location as an input parameter to get the weather at that location.

Schema
Provide the [OpenAPI schema](#) defining this tool's API. YAML and JSON are supported. You can use the [OpenAPI schema builder](#) to assist with building your OpenAPI schema specification.

☐ JSON ☒ YAML [Update the schema here](#)

Samples

```
1 openapi: 3.0.2
2 info:
3   title: Weather API
4   description: API to get the current weather based on location using Weatherbit.
5   version: 1.0.0
6 servers:
7   - url: 'https://us-central1-fresh-span-400217.cloudfunctions.net'
8     description: Cloud Function Deployment
9 paths:
10  /get_current_weather:
11    get:
```

7. Save the Tool

The screenshot shows the 'Tools' configuration page in the Agent Console. The tool name is 'get_current_weather' and the type is 'OpenAPI'. The 'Save' button is highlighted with a red box. The description is 'Use this tool to get the real time weather at a particular location. Pass the location as an input parameter to get the weather at that location.' The schema is defined in YAML format, showing an OpenAPI 3.0.2 specification for a 'Weather API' that provides current weather based on location using Weatherbit.

Agent Console

App: weather_agent

You are currently in the **upgraded version of the Agent Console (V2)**, which contains the latest bugs fixes and features. Feel free to report any bugs using the bug icon above. You can also switch back to V1 at any time via the banner link.

Tools [Save](#)

Using tools, you can connect agents to external systems. These systems can augment the knowledge of agents and empower them to execute complex tasks efficiently. [Learn more](#)

Tool name*
get_current_weather

Type
OpenAPI

Connect to an external API using an OpenAPI tool.

Description
Provide a description of this tool. This description is provided to the model as context informing how the tool is used.

Description
Use this tool to get the real time weather at a particular location. Pass the location as an input parameter to get the weather at that location.

Schema
Provide the [OpenAPI schema](#) defining this tool's API. YAML and JSON are supported. You can use the [OpenAPI schema builder](#) to assist with building your OpenAPI schema specification.

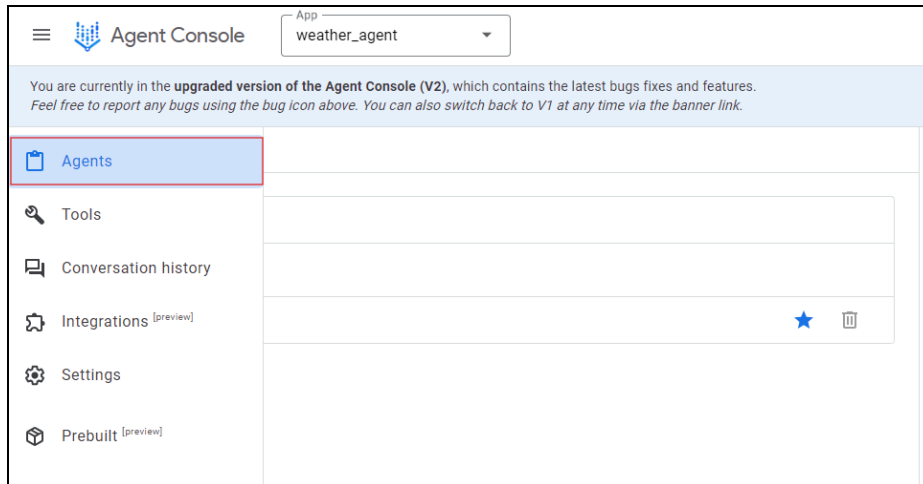
☐ JSON ☒ YAML

Samples

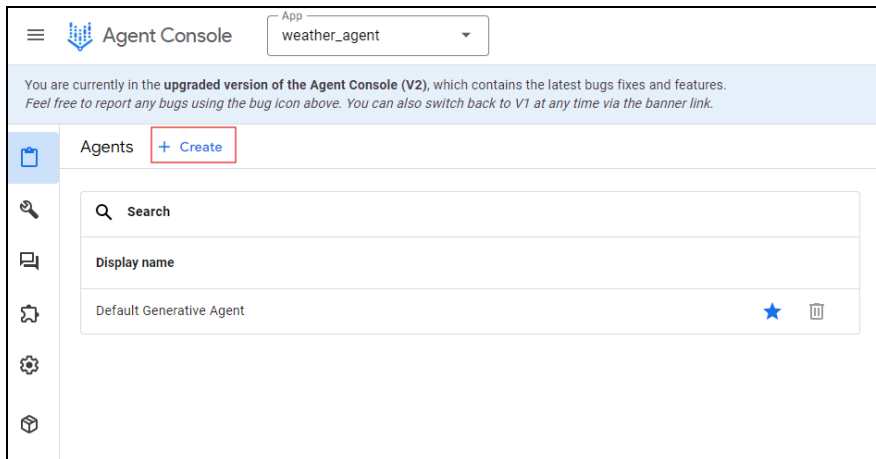
```
1 openapi: 3.0.2
2 info:
3   title: Weather API
4   description: API to get the current weather based on location using Weatherbit.
5   version: 1.0.0
6 servers:
7   - url: 'https://us-central1-fresh-span-400217.cloudfunctions.net'
8     description: Cloud Function Deployment
9 paths:
10  /get_current_weather:
11    get:
```

Step 3: Create an Agent using the tool created in Step 2 as one of the inputs

1. Click on “Playbook” in the left panel



2. Click on Create to create a new Agent



3. Update the following fields:

- a. Agent name : Weather Agent *Note: Change the name of the agent as per requirement
- b. Set the **Goal** of the Agent to :

Provide helpful and informative answers to the user's questions, using available tools like ``get_current_weather`` when appropriate. Use the ``get_current_weather`` tool to get the current real time weather at a location.

- c. Specify the **instructions** to be followed by the Agent. To reference a tool, use the syntax **\${TOOL: tool name}**:

- Determine the user's intent and what information they are seeking.
- If the user's question is about the current weather at a specific location, use the **\${TOOL:get_current_weather}** tool to retrieve the latest weather information.

The screenshot shows the 'Weather Agent' configuration interface in Google Cloud AI Studio. The 'Agent name' field is set to 'Weather Agent'. The 'Goal' field contains the text: 'Provide helpful and informative answers to the user's questions, using available tools like 'get_current_weather' when appropriate. Use the 'get_current_weather' tool to get the current real time weather at a location.' The 'Instructions' field contains a list of instructions: '- Determine the user's intent and what information they are seeking.' and '- If the user's question is about the current weather at a specific location, use the tool \${TOOL:get_current_weather} tool to retrieve the latest weather information.' Red boxes highlight these fields, and red arrows point to them with labels 'Set the Goal here' and 'Set the Instructions here'.

4. Save the Agent

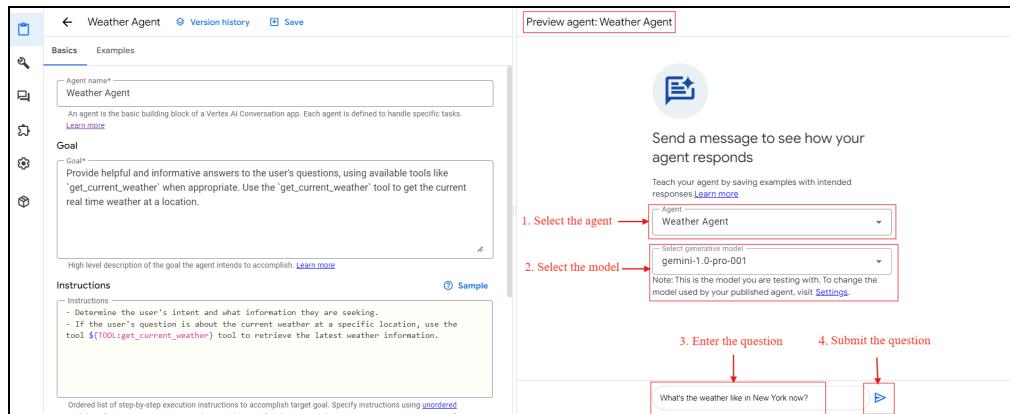
The screenshot shows the 'Weather Agent' configuration page in the Google Cloud AI Agent Builder. The interface includes a top navigation bar with a back arrow, the agent name 'Weather Agent', a 'Version history' link, and a 'Save' button (highlighted with a red box). Below the navigation bar are tabs for 'Basics' and 'Examples', with 'Basics' being the active tab. On the left side, there is a vertical toolbar with icons for a clipboard, search, chat, settings, and a cube. The main content area is divided into three sections: 'Agent name*' with a text input containing 'Weather Agent'; 'Goal' with a text area containing 'Provide helpful and informative answers to the user's questions, using available tools like `get_current_weather` when appropriate. Use the `get_current_weather` tool to get the current real time weather at a location.'; and 'Instructions' with a text area containing two bullet points: '- Determine the user's intent and what information they are seeking.' and '- If the user's question is about the current weather at a specific location, use the tool `tool ${TOOL: get_current_weather}` tool to retrieve the latest weather information.' Each section has a 'Learn more' link. At the bottom of the 'Instructions' section, there is a note: 'Ordered list of step-by-step execution instructions to accomplish target goal. Specify instructions using `unordered`'.

Step 4: Testing the Agent created

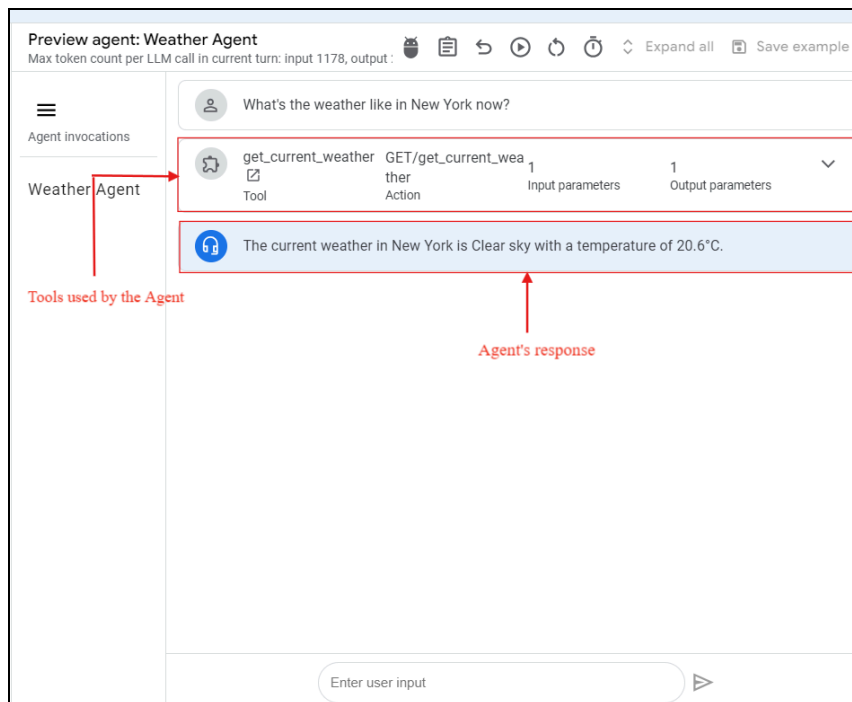
Follow these steps to test the Agent Builder:

Use the Preview agent: section to ask questions

1. Go to Conversation History from the tool bar on the left side
2. Select the agent created, Weather Agent. *Note: Select the correct agent
3. Select the GenAI model, **gemini-1.0-pro-001 Or the latest version of Gemini model**
4. Enter the question in the text box. Example question: 'What's the weather like in New York now?'
5. Click the **submit** button or press Enter



Get a list of tools used by the agent to come up with the final answer



Expand the tool to view its inputs and outputs in detail.

Preview agent: Weather Agent
Max token count per LLM call in current turn: input 1178, output 1178

Agent invocations

Weather Agent

What's the weather like in New York now?

get_current_weather GET/get_current_wea 1 1
Tool Action Input parameters Output parameters

Tool*
get_current_weather

Input to get_current_weather tool Output of get_current_weather tool

Action name
GET/get_current_weather

Name of the action to be called during the tool use.

Tool input

Input (location) *
"New York"

Tool output

Output (200) *
"The current weather in New York is Clear sky with a temperature of 20.6°C."

The current weather in New York is Clear sky with a temperature of 20.6°C.

Enter user input

Congratulations!

In this lab, you have successfully learnt about how to build an agent using the Agent Builder using external APIs

Lab Author: Naresh Jasotani(nareshjasotani@google.com)

