Quick tip: Review the prerequisites before you run the lab

Start Lab 01:00:00

Getting Started with Google Generative AI Using the Gen AI SDK

Lab 1 hour No cost Introductory

★★★★½

This lab may incorporate AI tools to support your learning.

GSP1209

Google Cloud Self-Paced Labs

Previous Next

Overview

The Google Gen AI SDK provides a unified interface to Google's generative AI API services. This SDK simplifies the process of integrating generative AI capabilities into applications and services, enabling developers to leverage Google's advanced AI models for various tasks. In this lab, you explore the Google Gen AI SDK, learning to connect to AI services, send diverse prompts, and fine-tune responses from Gemini. You also get hands-on experience with more advanced techniques to prepare you to leverage the power of generative AI for your own projects.

Objectives

- Installing the Gen AI SDK.
- Connecting to an API service.
- Sending text and multimodal prompts.
- Setting system instructions.
- Configuring model parameters and safety filters.
- Managing model interactions (multi-turn chat, content streaming, asynchronous requests).
- Using advanced features (token counting, context caching, function calling, batch prediction, text embeddings).

Prerequisites

Before starting this lab, you should be familiar with:

- Basic Python programming.
- General API concepts.
- Running Python code in a Jupyter notebook on Vertex AI Workbench.

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 are made available to you.

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

To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).

Note: Use an Incognito (recommended) or private browser window to run this lab. This prevents conflicts between your personal account and the student account, which may cause extra charges incurred to your personal account.

Thanks for reviewing this lab.

Note: Use only the student account for this lab. If you use a different Google Cloud account, you may incur charges to that account.

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 dialog opens for you to select your payment method. On the left is the Lab Details pane with the following:
    - The Open Google Cloud console button
    - Time remaining
    - The temporary credentials that you must use for this lab
    - Other information, if needed, to step through this lab
2. Click **Open Google Cloud console** (or right-click and select **Open Link in**

Thanks for reviewing this lab.

   The lab spins up resources, and then opens another tab that shows the Sign in page.

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

   Note: If you see the **Choose an account** dialog, click **Use Another Account**.

3. If necessary, copy the **Username** below and paste it into the **Sign in** dialog.

   ```
   "Username"
   ```

   You can also find the Username in the Lab Details pane.

4. Click **Next**.
5. Copy the **Password** below and paste it into the **Welcome** dialog.

   ```
   "Password"
   ```

   You can also find the Password in the Lab Details pane.

Thanks for reviewing this lab.

6. Click **Next**.

   Important: You must use the credentials the lab provides you. Do not use your Google Cloud account credentials.

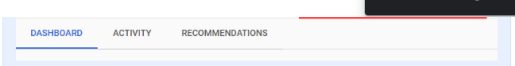
   Note: Using your own Google Cloud account for this lab may incur extra charges.

7. 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 Google Cloud console opens in this tab.

Note: To access Google Cloud products and services, click the **Navigation menu** or type the service or product name in the **Search** field.

Thanks for reviewing this lab.



Task 1. Open the notebook in Vertex AI Workbench

1. In the Google Cloud console, on the **Navigation menu** (☰), click **Vertex AI > Workbench**.

2. Find the `Workbench instance name` instance and click on the **Open JupyterLab** button.

The JupyterLab interface for your Workbench instance opens in a new browser tab.

Task 2. Set up the notebook

Open your notebook file, import your libraries, and choose your model.

1. Open the `notebook name` file.
2. In the **Select Kernel** dialog, choose **Python 3** from the list of available kernels.
3. Run through these sections of the notebook:
    - **Get started**
    - **Use Google Gen AI SDK**
    - **Connect to a generative AI API service**

For **Project ID**, use `Project ID`, and for **Location**, use `Region`.

Note: Skip any notebook cells that are noted *Colab only*. If you experience a 429 response from any of the notebook cell executions, wait 1 minute before running the

Click **Check my progress** to verify the objective.

Import libraries and set up the notebook

Check my progress

Task 3. Interact with the model

For more information about all AI models and APIs on Vertex AI, refer to Google Models and Model Garden.

Choose a model

- Run the **Choose a model** section of the notebook.

Send text prompts

Use the `generate_content` method to generate responses to your prompts. You can pass text to `generate_content`, and use the `.text` property to get the text content of the response.

- Run the **Send text prompts** section of the notebook.

Send multimodal prompts

You can include text, PDF documents, images, audio and video in your prompt requests and get text or code responses.

You can also pass the file URL in `Part.from_uri` in the request to the model directly.

- Run the **Send multimodal prompts** section of the notebook.

Set the system instructions

responses, and adhere to guidelines over the user interaction.

- Run the **Set system instruction** section of the notebook.

Click **Check my progress** to verify the objective.

Interact with the model

Check my progress

Task 4. Configure and control the model

Configure model parameters

You can include parameter values in each call that you send to a model to control how

values.

Thanks for reviewing this lab.

- Run the **Configure model parameters** section of the notebook.

Configure safety filters

The Gemini API provides safety filters that you can adjust across multiple filter categories to restrict or allow certain types of content. You can use these filters to adjust what's appropriate for your use case. Refer to the Configure safety filters page for details.

When you make a request to the model, the content is analyzed and assigned a safety rating. You can inspect the safety ratings of the generated content by printing out the model responses, as in this example:

- Run the **Configure safety filters** section of the notebook.

Start a multi-turn chat

The Gemini API enables you to have freeform conversations across multiple turns.

Control generated output

The controlled generation capability in Gemini API allows you to constrain the model output to a structured format. You can provide the schemas as Pydantic Models or a JSON string.

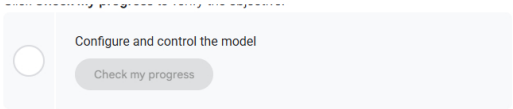
You also can define a response schema in a Python dictionary. You can use only the fields below. All other fields are ignored.

- `enum`
- `items`
- `maxItems`
- `nullable`
- `properties`
- `required`

In this example, you instruct the model to analyze product review data, extract key entities, perform sentiment classification (multiple choices), provide additional explanation, and output the results in JSON format.

- Run the **Control generated output** section of the notebook

Click Check my progress to verify the objective.



Task 5. Manage the model interaction

Generate content stream

By default, the model returns a response after completing the entire generation process. You can also use the `generate_content_stream` method to stream the response as it is being generated. The model returns chunks of the response as they are generated.

- Run the **Generate content stream** section of the notebook.

Send asynchronous requests

You can send asynchronous requests using the `client.aio` module. This module exposes all the analogous async methods that are available on `client`.

For example, `client.aio.models.generate_content` is the async version of `client.models.generate_content`.

- Run the **Send asynchronous requests** section of the notebook.

Count tokens and compute tokens

You can use the `count_tokens` method to calculate the number of input tokens before sending a request to the Gemini API. Refer to the List and count tokens page for details.

Count tokens

- Run the **Count tokens** section of the notebook.

Compute tokens

- Run the **Compute tokens** section of the notebook.

Manage the model interaction

Check my progress

Task 6. Advanced features

Function calling

Function calling lets you provide a set of tools that it can use to respond to the user's prompt. You create a description of a function in your code, then pass that description to a language model in a request. The response from the model includes the name of a function that matches the description and the arguments to call it with.

- Run the **Function calling** section of the notebook.

Click **Check my progress** to verify the objective.

Function calling

Check my progress

Use context caching

Context caching lets you store frequently used input tokens in a dedicated cache and reference them for subsequent requests. This eliminates the need to repeatedly pass the same set of tokens to a model.

Note: Context caching is only available for stable models with fixed versions (for example, `gemini-2.0-flash-001`). You must include the version postfix (for example, the `-001`).

Create a cache

- Run the **Create a cache** section of the notebook.

- Run the **Use a cache** section of the notebook.

Delete a cache

- Run the **Delete a cache** section of the notebook.

Click **Check my progress** to verify the objective.

Use context caching

Check my progress

Batch prediction

Different from getting online (synchronous) responses, where you are limited to one input request at a time, batch predictions for the Gemini API in Vertex AI allow you to send a large number of requests to Gemini in a single batch request. Then, the model responses asynchronously populate to your storage output location in Cloud Storage or BigQuery.

Prepare batch inputs

The input for batch requests specifies the items to send to your model for prediction.

Batch requests for Gemini accept BigQuery storage sources and Cloud Storage sources. You can learn more about the batch input formats in the Batch text generation page.

This lab uses Cloud Storage as an example. The requirements for Cloud Storage input are:

- File format: JSON Lines (JSONL)
- Located in `us-central1`
- Appropriate read permissions for the service account

Each request that you send to a model can include parameters that control how the model generates a response. Learn more about Gemini parameters in the Experiment

with parameter values page.

This is one of the example requests in the input JSONL file `batch_requests_for_multimodal_input_2.jsonl`:


```
{"request":{"contents": [{"role": "user", "parts": [{"text": "List
```

```
"image/jpeg"}}]}],"generationConfig":{"temperature": 0.4}}}
```


- Run the **Prepare batch inputs** section of the notebook.

Prepare batch output location

When a batch prediction task completes, the output is stored in the location specified in your request.

- The location is in the form of a Cloud Storage or BigQuery URI prefix, for example: `gs://path/to/output/data` or `bq://projectId.bqDatasetId`.
- If not specified, `gs://STAGING_BUCKET/gen-ai-batch-prediction` is used for Cloud Storage source and `bq://PROJECT_ID.gen_ai_batch_prediction.predictions_TIMESTAMP` is used for BigQuery source.

This lab uses a Cloud Storage bucket as an example for the output location.

You can specify the URI of your Cloud Storage bucket in `BUCKET_URI`, or, if it is not specified, a new Cloud Storage bucket in the form of `gs://PROJECT_ID-TIMESTAMP` is be created for you.

Send a batch prediction request

To make a batch prediction request, you specify a source model ID, an input source and an output location where Vertex AI stores the batch prediction results.

For more, see the Batch prediction API page. You can also check the status in the console at https://console.cloud.google.com/vertex-ai/batch-predictions

- Run the **Send a batch prediction request** section of the notebook.

> **Note:** it may take a few minutes for your batch prediction to complete.

Retrieve batch prediction results

When a batch prediction task is complete, the output of the prediction is stored in the location specified in your request. It is also available in `batch_job.dest.bigquery_uri` or `batch_job.dest.gcs_uri`.

Example output:


```
{"status": "", "processed_time": "2024-11-13T14:04:28.376+00:00",
```

```
"image/jpeg"}, "text": null}], "role": "user"}], "generationConfig":
{"temperature": 0.4}}, "response": {"candidates": [{"avgLogprobs":
-0.10394711927934126, "content": {"parts": [{"text": "Here's a list of
the objects in the image:\n\n* **Watering can:** A green plastic
watering can with a white rose head.\n* **Plant:** A small plant
(possibly oregano) in a terracotta pot.\n* **Terracotta pots:** Two
terracotta pots, one containing the plant and another empty, stacked on
top of each other.\n* **Gardening gloves:** A pair of striped gardening
gloves.\n* **Gardening tools:** A small trowel and a hand cultivator
(hoe).  Both are green with black handles."}], "role": "model"},
"finishReason": "STOP"}], "modelVersion": "gemini-2.0-flash-
001@default", "usageMetadata": {"candidatesTokenCount": 110,
"promptTokenCount": 264, "totalTokenCount": 374}}
```


- Run the **Retrieve batch prediction results** section of the notebook.

Click **Check my progress** to verify the objective.

Retrieve batch prediction results

Check my progress

Get text embeddings

You can get text embeddings for a snippet of text by using `embed_content` method. All models produce an output with 768 dimensions by default. However, some models give users the option to choose an output dimensionality between `1` and `768`. See Vertex AI text embeddings API for details.

- Run the **Get text embeddings** section of the notebook.

Click **Check my progress** to verify the objective.

Get text embeddings

Check my progress

Congratulations!

connect to AI services, send diverse prompts, and fine-tune responses from the Gemini model. You've also got hands-on experience with more advanced techniques like managing interactions, using context caching, and even working with embeddings! Now you're well-equipped to leverage the power of generative AI for your own projects.

Next steps / learn more

Check out the following resources to learn more about Gemini:

- Gemini Overview
- Generative AI on Vertex AI Documentation
- Generative AI on YouTube
- Explore the Vertex AI Cookbook for a curated, searchable gallery of notebooks for Generative AI.
- Explore other notebooks and samples in the Google Cloud Generative AI repository.

Google Cloud training and certification

...helps you make the most of Google Cloud technologies. Our classes include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. Certifications help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated May 05, 2025

Lab Last Tested May 05, 2025