



Google Vertex AI Gemini

Vertex AI is a product built on top of the Google Cloud Platform that provides access to Google's large generative models, including the older generation (PaLM2) and the newer generation (Gemini).

To utilize Vertex AI, one must first create a Google Cloud Platform account.

Get started

Create Google Cloud Account

If you're new to Google Cloud, you can create a new account by clicking on the `[create an account]` button located under `Get set up on Google Cloud` dropdown menu on the following page:

[Create an account](#)

Create a project within your Google Cloud Platform account.

Within your Google Cloud Account create a new project and enable the Vertex AI APIs by following the steps outlined below:

[Create a new project](#)

Note your `PROJECT_ID` as it will be required for future API calls..

Select the Google Cloud authentication strategy

There are several ways on how your application authenticates to Google Cloud services and APIs. For example, you can create a [service account](#) and set up environment variable `GOOGLE_APPLICATION_CREDENTIALS` to the path of the JSON file that contains your credentials.

You can discover all the authentication strategies [here](#). But for simplicity of local testing we will be using authentication via `gcloud` utility.

Install Google Cloud CLI (Optional)

To access your cloud projects locally, you can install `gcloud` tool by following the [installation instructions](#). For GNU/Linux operating systems, the installation steps are as follows:

1. Download SDK:

```
curl -O  
https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-  
cloud-cli-467.0.0-linux-x86_64.tar.gz
```

2. Extract an archive:

```
tar -xf google-cloud-cli-467.0.0-linux-x86_64.tar.gz
```

3. Run an installation script:

```
cd google-cloud-sdk/  
./install.sh
```

4. Run the following command to set up a default project and authentication credentials:

```
gcloud auth application-default login
```

This authentication method is compatible with both the `vertex-ai` (Embedding models, PaLM2) and `vertex-ai-gemini` (Gemini) packages.

Add dependencies

To get started, add the following dependencies to your project's `pom.xml`:

```
<dependency>  
  <groupId>dev.langchain4j</groupId>  
  <artifactId>langchain4j-vertex-ai-gemini</artifactId>
```

```
<version>0.32.0</version>
</dependency>
```

or project's `build.gradle`:

```
implementation 'dev.langchain4j:langchain4j-vertex-ai-gemini:0.32.0'
```

Try out an example code:

Example of using chat model for text prediction

Gemini Pro Vision with Image input

The `PROJECT_ID` field represents the variable you set when creating a new Google Cloud project.

```
import dev.langchain4j.data.message.AiMessage;
import dev.langchain4j.data.message.ImageContent;
import dev.langchain4j.data.message.TextContent;
import dev.langchain4j.data.message.UserMessage;
import dev.langchain4j.model.chat.ChatLanguageModel;
import dev.langchain4j.model.output.Response;
import dev.langchain4j.model.vertexai.VertexAiGeminiChatModel;

public class GeminiProVisionWithImageInput {

    private static final String PROJECT_ID = "YOUR-PROJECT-ID";
    private static final String LOCATION = "us-central1";
    private static final String MODEL_NAME = "gemini-pro-vision";
    private static final String CAT_IMAGE_URL =
        "https://upload.wikimedia.org/" +
            "wikipedia/commons/e/e9/" +

        "Felis_silvestris_silvestris_small_gradual_decrease_of_quality.png";

    public static void main(String[] args) {
        ChatLanguageModel visionModel =
            VertexAiGeminiChatModel.builder()
                .project(PROJECT_ID)
                .location(LOCATION)
                .modelName(MODEL_NAME)
                .build();
```

```

        Response<AiMessage> response = visionModel.generate(
            UserMessage.from(
                ImageContent.from(CAT_IMAGE_URL),
                TextContent.from("What do you see?")
            )
        );

        System.out.println(response.content().text());
    }
}

```

Available models

Model name	Description	Properties
gemini-pro	Designed to handle natural language tasks, multiturn text and code chat, and code generation.	Max total tokens (input and output): 32,760. Max output tokens: 8,192
gemini-pro-vision	Supports multimodal prompts. You can include text, images, and video in your prompt requests and get text or code responses.	Max total tokens (input and output): 16,384. Max output tokens: 2,048
gemini-ultra	Google's most capable multimodal model, optimized for complex tasks including instruction, code, and reasoning, with support for multiple languages.	Max tokens input: 8,192. Max tokens output: 2,048
gemini-ultra-vision	Google's most capable multimodal vision model, optimized to support text, images, videos, and multi-turn chat.	Max tokens input: 8,192. Max tokens output: 2,048

Note that in March 2024, the Ultra version has private access with an allow list. Therefore, you may receive an exception similar to this:

```

Caused by: io.grpc.StatusRuntimeException:
  FAILED_PRECONDITION: Project `1234567890` is not allowed to use
  Publisher Model

```

```
`projects/{YOUR_PROJECT_ID}/locations/us-central1/publishers/google/models/gemini-ultra`
```

Warning

Please note that Gemini does not support `SystemMessage`s. If there are `SystemMessage`s provided to the `generate()` methods, they will be merged into the first `UserMessage` (before the content of the `UserMessage`).

Apply for early access

[Early access for Gemma](#)

[Early access for Gemini 1.5 Pro](#)

References

[Available locations](#)

[Multimodal capabilities](#)

Examples

- [Google Vertex AI Gemini Examples](#)

 [Edit this page](#)