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).

# Vertex AI client libraries

This page shows how to get started with the Cloud Client Libraries for the Vertex AI API. Client libraries make it easier to access Google Cloud APIs from a supported language. Although you can use Google Cloud APIs directly by making raw requests to the server, client libraries provide simplifications that significantly reduce the amount of code you need to write.

Read more about the Cloud Client Libraries and the older Google API Client Libraries in Client libraries explained (/apis/docs/client-libraries-explained).

## Install the client library

C#Go (#go)Java (#java)Node.js (#node.js) (#c)

```
Install-Package Google.Cloud.AIPlatform.V1 -Pre
```

For more information, see Setting Up a C# Development Environment (/dotnet/docs/setup).

## Set up authentication

To authenticate calls to Google Cloud APIs, client libraries support Application Default Credentials (ADC) (/docs/authentication/application-default-credentials); the libraries look for credentials in a set of defined locations and use those credentials to authenticate requests to the API. With ADC, you can make credentials available to your application in a variety of environments, such as local development or production, without needing to modify your application code.

For production environments, the way you set up ADC depends on the service and context. For more information, see Set up Application Default Credentials (/docs/authentication/provide-credentials-adc).

For a local development environment, you can set up ADC with the credentials that are associated with your Google Account:

1. After installing (/sdk/docs/install) the Google Cloud CLI, initialize (/sdk/docs/initializing) it by running the following command:

   ```
   gcloud init
   ```

   If you're using an external identity provider (IdP), you must first sign in to the gcloud CLI with your federated identity (/iam/docs/workforce-log-in-gcloud).

2. If you're using a local shell, then create local authentication credentials for your user account:

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

   You don't need to do this if you're using Cloud Shell.

   If an authentication error is returned, and you are using an external identity provider (IdP), confirm that you have signed in to the gcloud CLI with your federated identity (/iam/docs/workforce-log-in-gcloud).

   A sign-in screen appears. After you sign in, your credentials are stored in the local credential file used by ADC (/docs/authentication/application-default-credentials#personal).

## Use the client library

The following example shows how to use the client library.

C#Go (#go)Java (#java)Node.js (#node.js) (#c)

```
using Google.Api.Gax.Grpc;
using Google.Cloud.AIPlatform.V1;
```

```
using System.Text;
using System.Threading.Tasks;

public class GeminiQuickstart
{
    public async Task<string> GenerateContent(
        string projectId = "your-project-id",
        string location = "us-central1",
        string publisher = "google",
        string model = "gemini-2.0-flash-001"
    )
    {
        // Create client
        var predictionServiceClient = new PredictionServiceClientBuilder
        {
            Endpoint = $"{location}-aiplatform.googleapis.com"
        }.Build();

        // Initialize content request
        var generateContentRequest = new GenerateContentRequest
        {
            Model = $"projects/{projectId}/locations/{location}/publishers/{pub
            GenerationConfig = new GenerationConfig
            {
                Temperature = 0.4f,
                TopP = 1,
                TopK = 32,
                MaxOutputTokens = 2048
            },
            Contents =
            {
                new Content
                {
                    Role = "USER",
                    Parts =
                    {
                        new Part { Text = "What's in this photo?" },
                        new Part { FileData = new() { MimeType = "image/png", F
                    }
                }
            }
        };

        // Make the request, returning a streaming response
        using PredictionServiceClient.StreamGenerateContentStream response = pr

        StringBuilder fullText = new();
```

```
        // Read streaming responses from server until complete
        AsyncResponseStream<GenerateContentResponse> responseStream = response.
        await foreach (GenerateContentResponse responseItem in responseStream)
        {
            fullText.Append(responseItem.Candidates[0].Content.Parts[0].Text);
        }

        return fullText.ToString();
    }
}
```

## Additional resources

C#Go (#go)Java (#java)Node.js (#node.js) (#c)

The following list contains links to more resources related to the client library for C#:

- API reference (https://cloud.google.com/dotnet/docs/reference/Google.Cloud.AIPlatform.V1/latest)
- Client libraries best practices (/apis/docs/client-libraries-best-practices)
- Issue tracker (https://github.com/googleapis/google-cloud-dotnet/issues)
- on Stack Overflow (https://stackoverflow.com/search?q=%5B%5D+%5Bc%23%5D)
- Source code (https://github.com/googleapis/google-cloud-dotnet)



Last updated 2025-06-06 UTC.