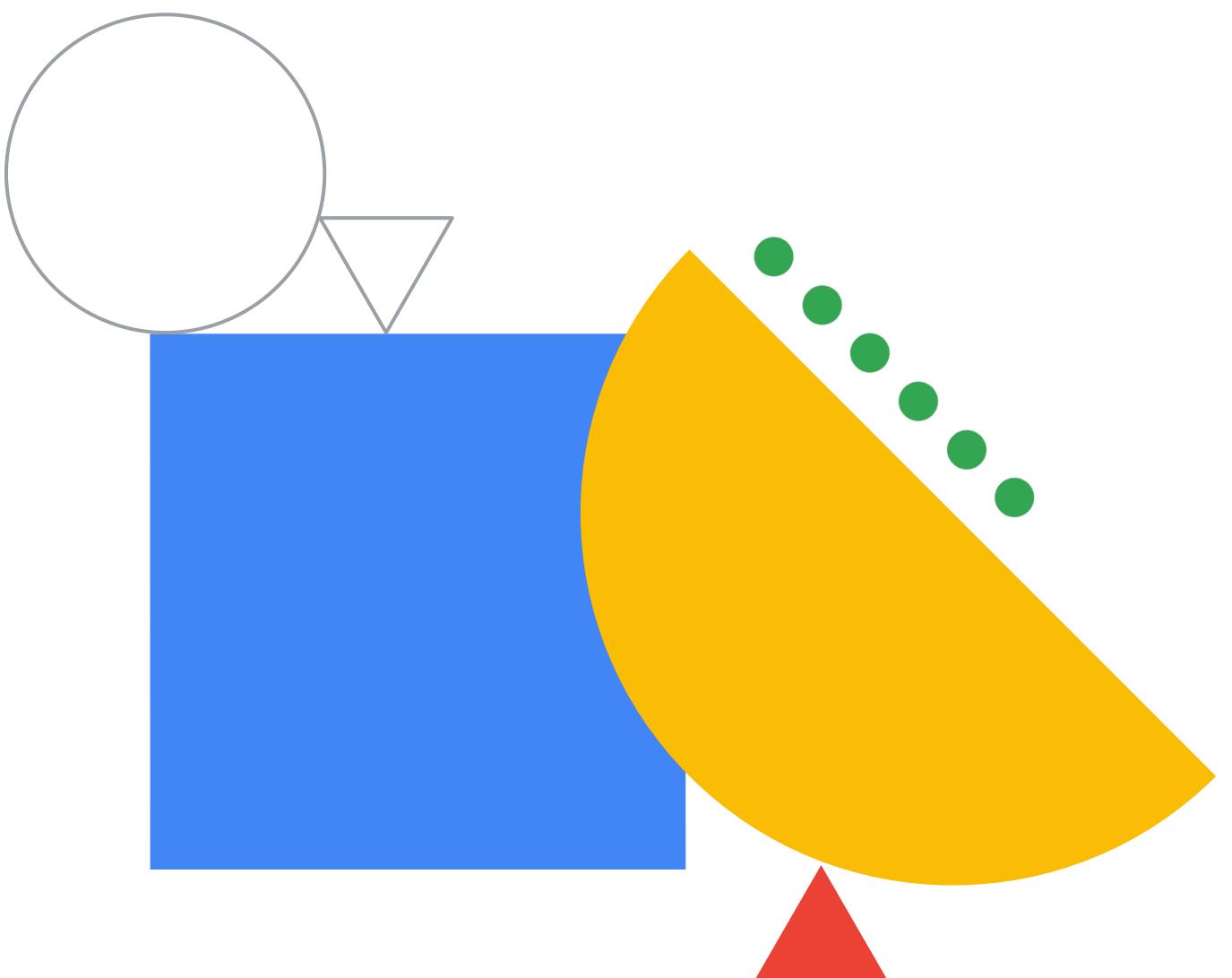
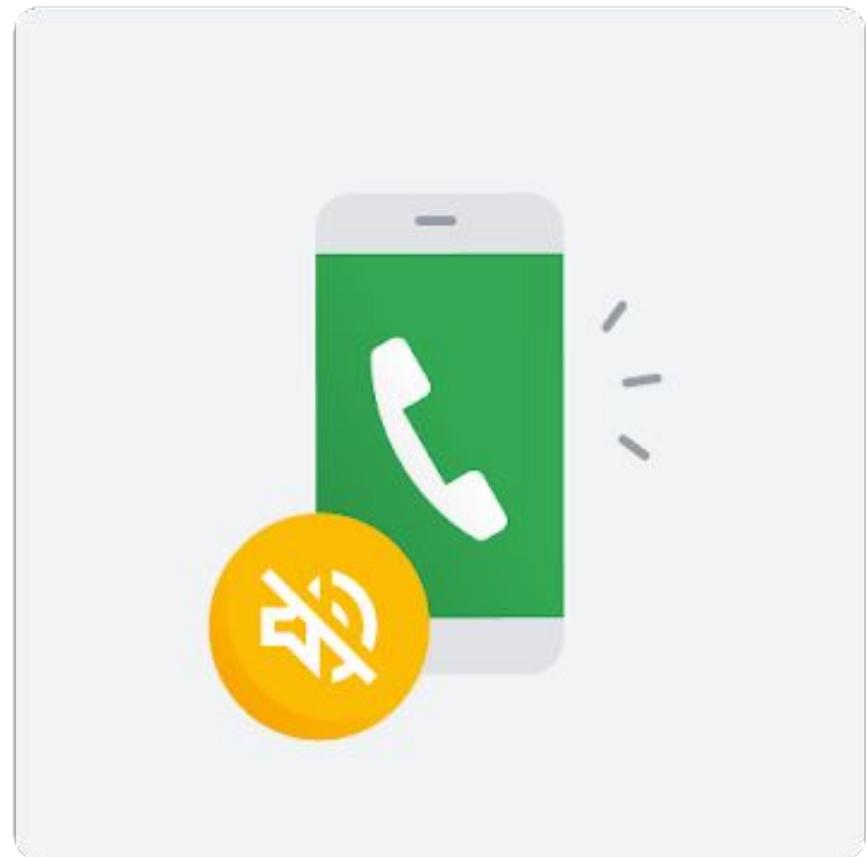


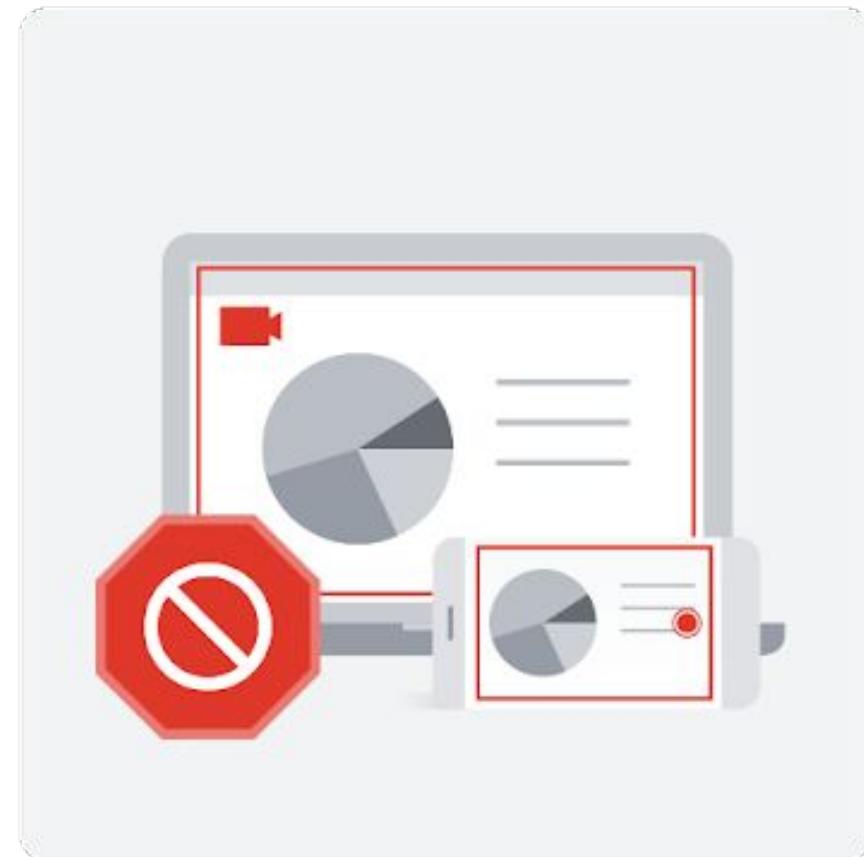
# Interactive Chat for Applications using Gen AI Studio



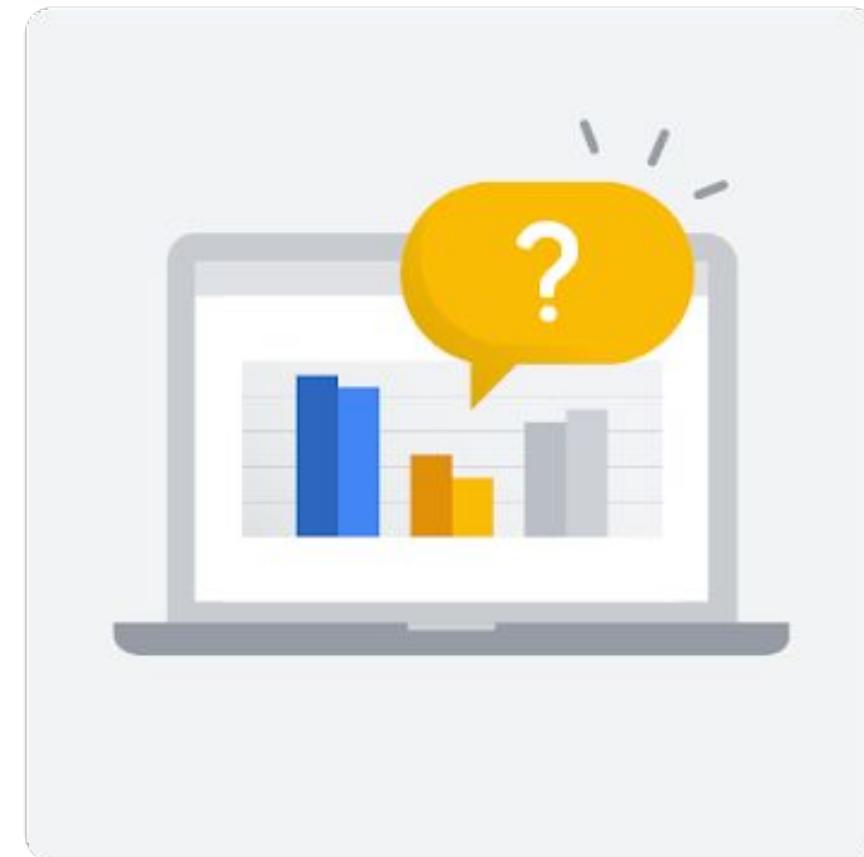
# Etiquette



No calls

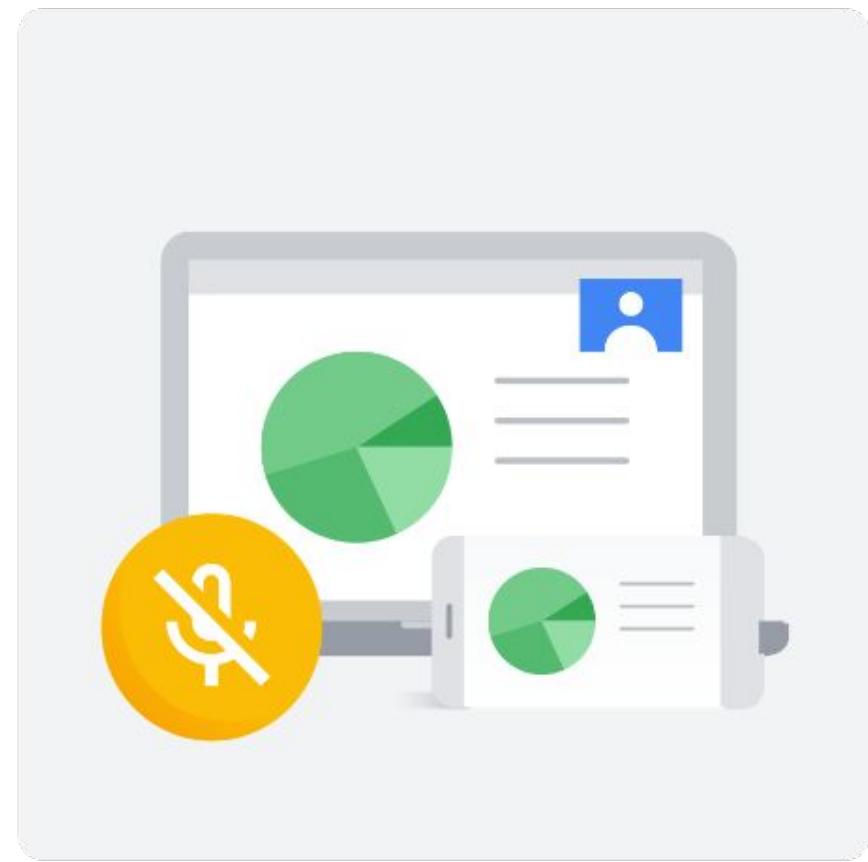


No recording

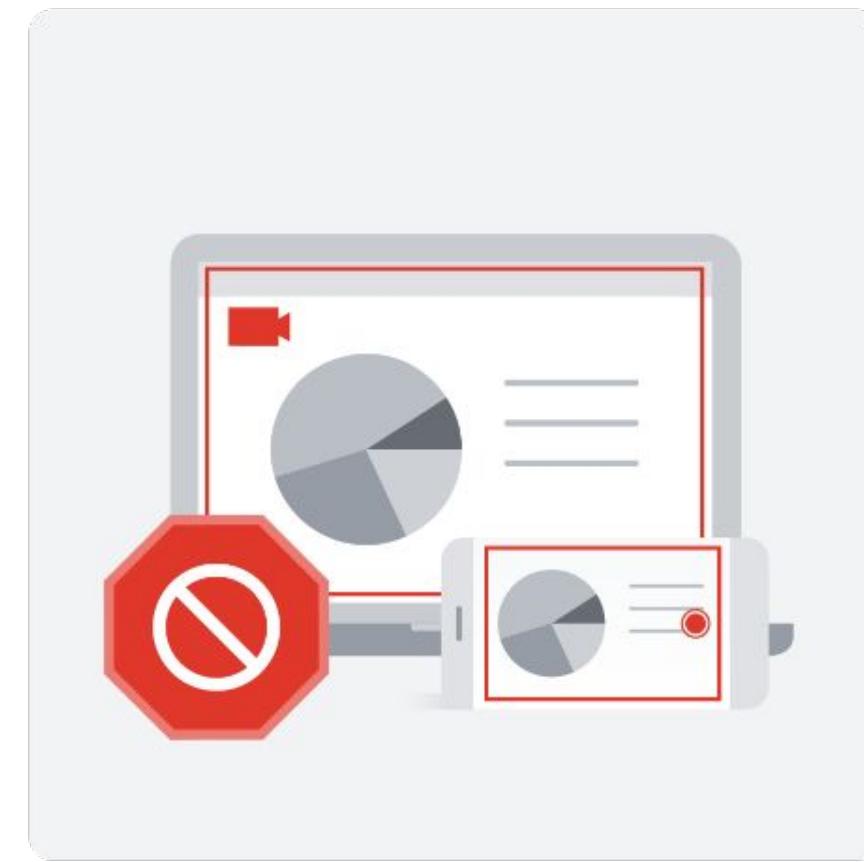


Ask questions

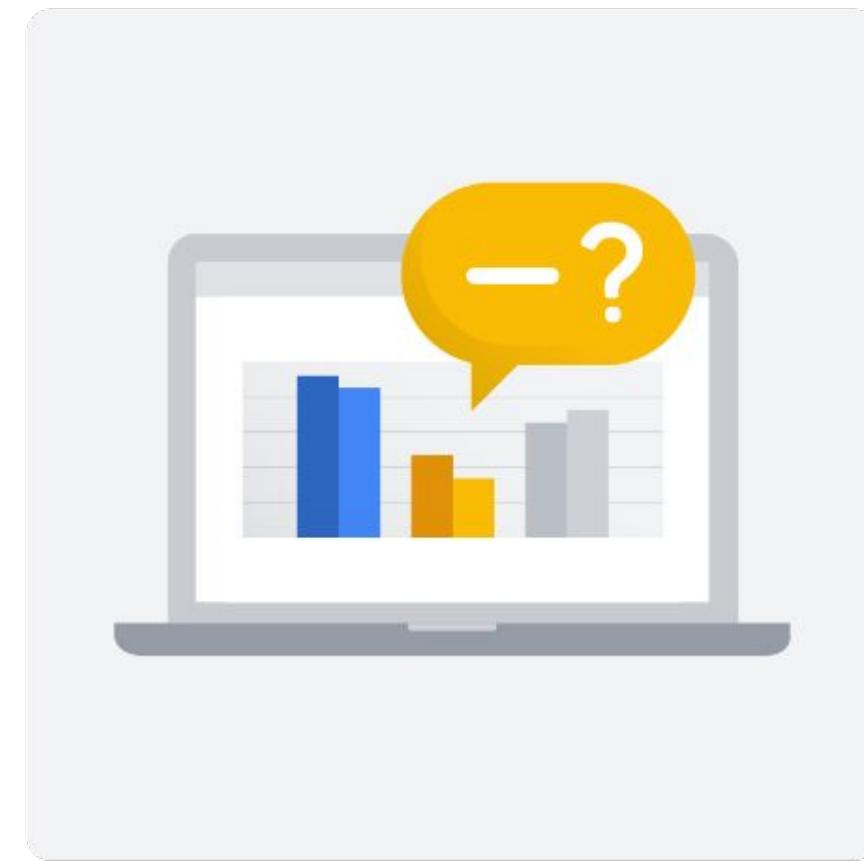
# Etiquette



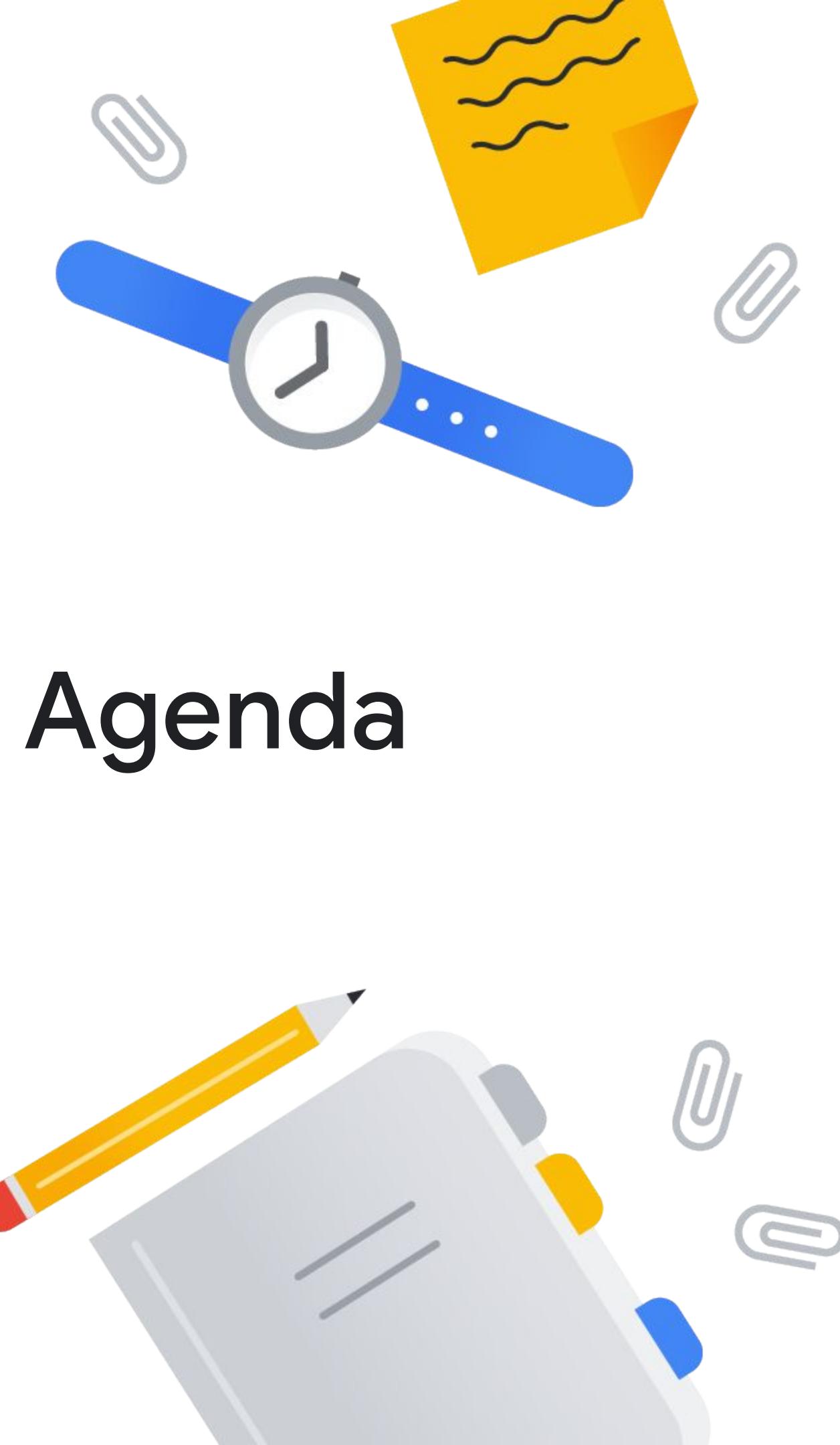
Mute microphone



No recording



Ask questions



# Agenda

- 01 Generative AI on Vertex AI
- 02 Gen AI Studio
- 03 Prompt Design
- 04 Implementing the PaLM API
- 05 Best Practices for Interactive Chat Applications

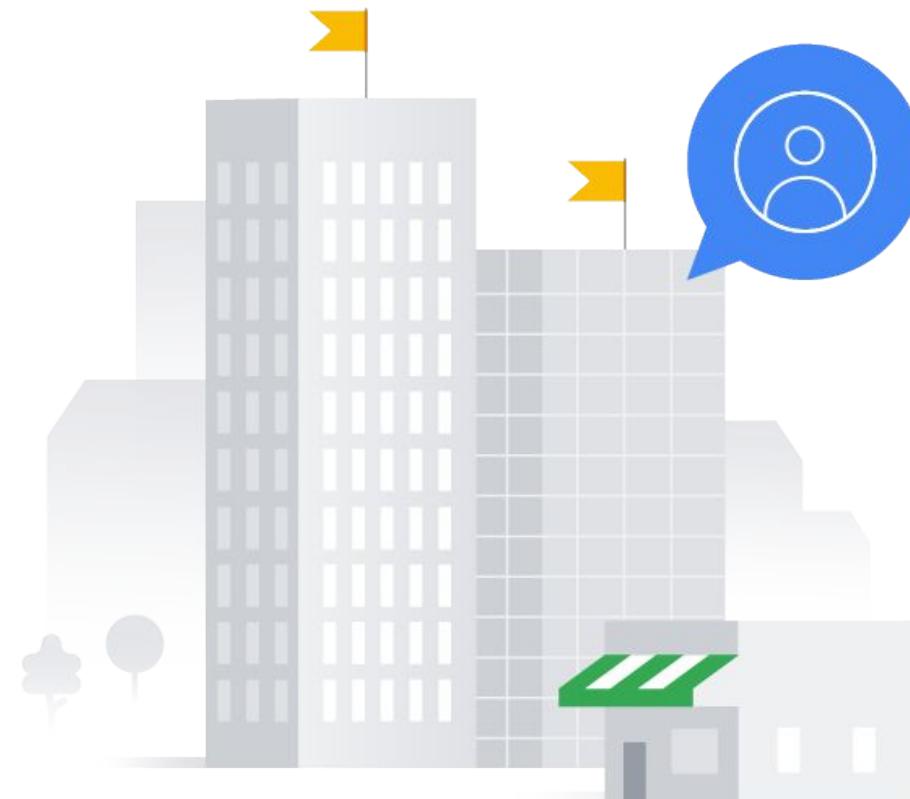
# Objectives

- 01 Understand Vertex AI generative AI options for your applications
- 02 Explore Gen AI Studio to interact with foundation models
- 03 Design and tune chat prompts for your Generative AI use cases
- 04 Understand the effect of changing PaLM API parameters on model responses
- 05 Implement the PaLM API into your applications using the Python SDK



# Target audience

- Those developing using Generative AI features on Google Cloud's Vertex AI platform



ML researchers



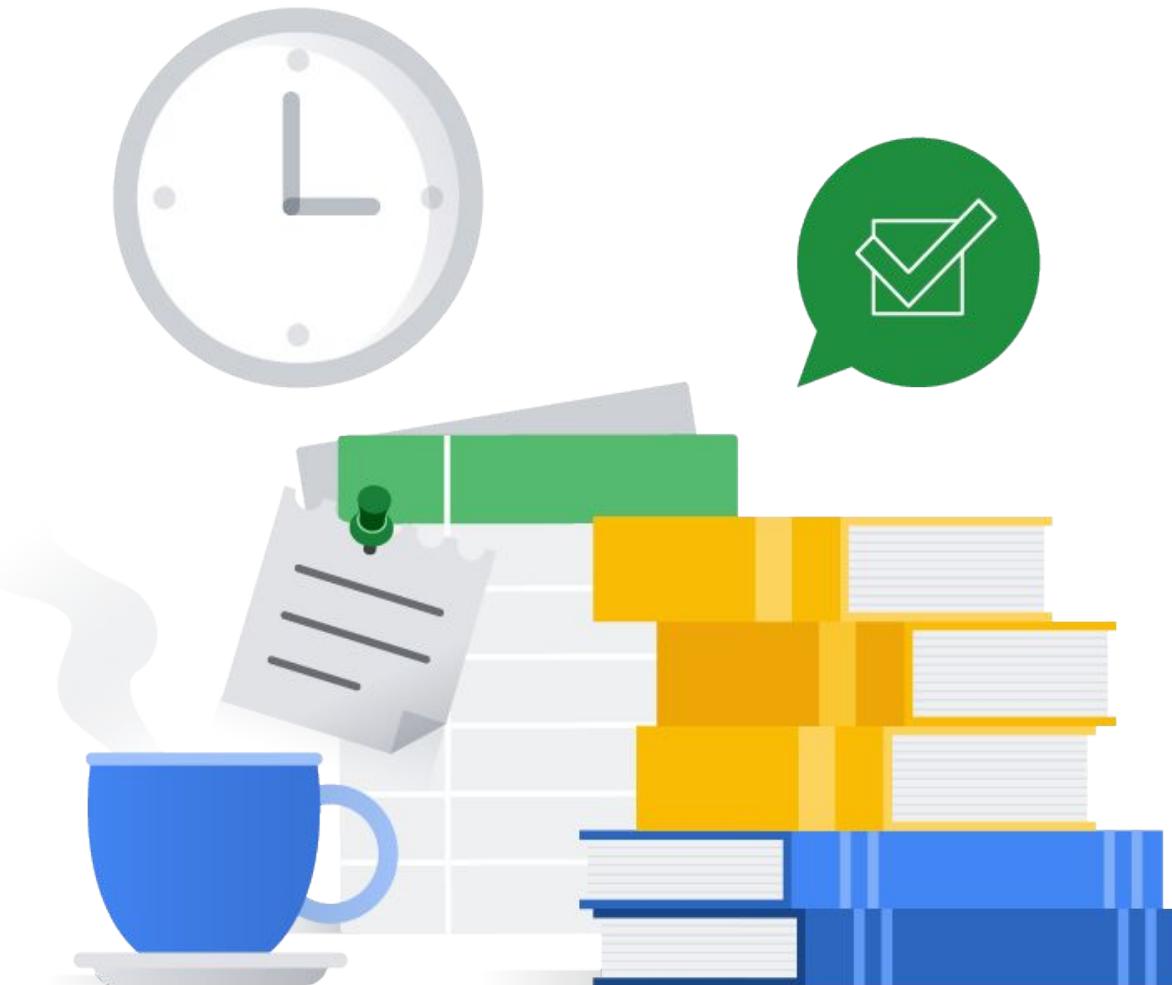
Programmers



App developers

Google Cloud

# Helpful knowledge



Google Cloud basics



Helpful to be familiar with



Python programming



Machine learning basics



Leveraging APIs in applications

# Lab environment

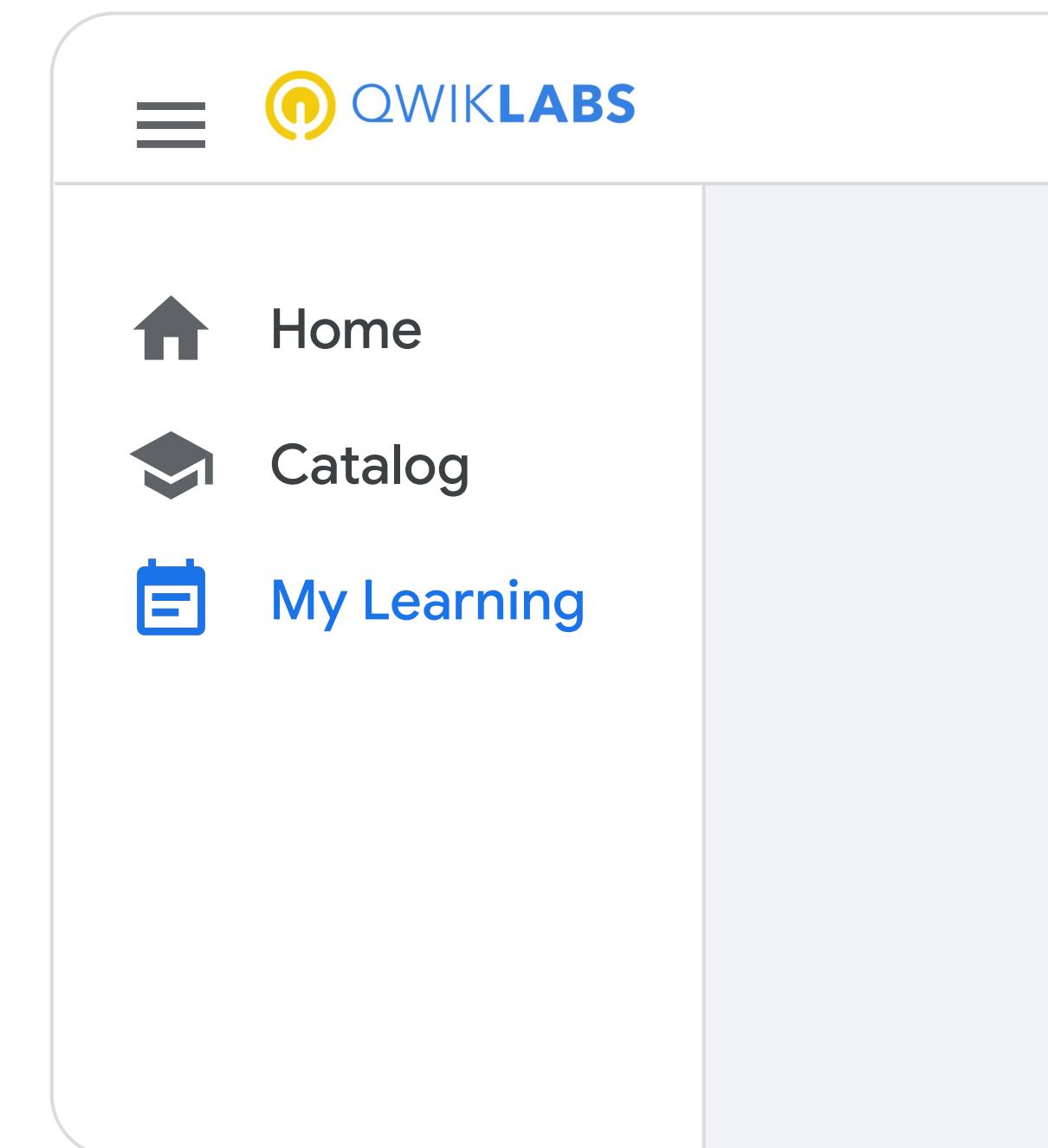
For each lab, Qwiklabs offers:

- A free set of resources for a fixed amount of time
- A clean environment with permissions



# Open Qwiklabs

- 1 Open an incognito window (or private/anonymous window).
- 2 Go to the Qwiklabs URL your instructor provides.
- 3 Sign In with existing account or Join with new account (with email you used to register for the course).
- 4 Launch the course from My Learning.

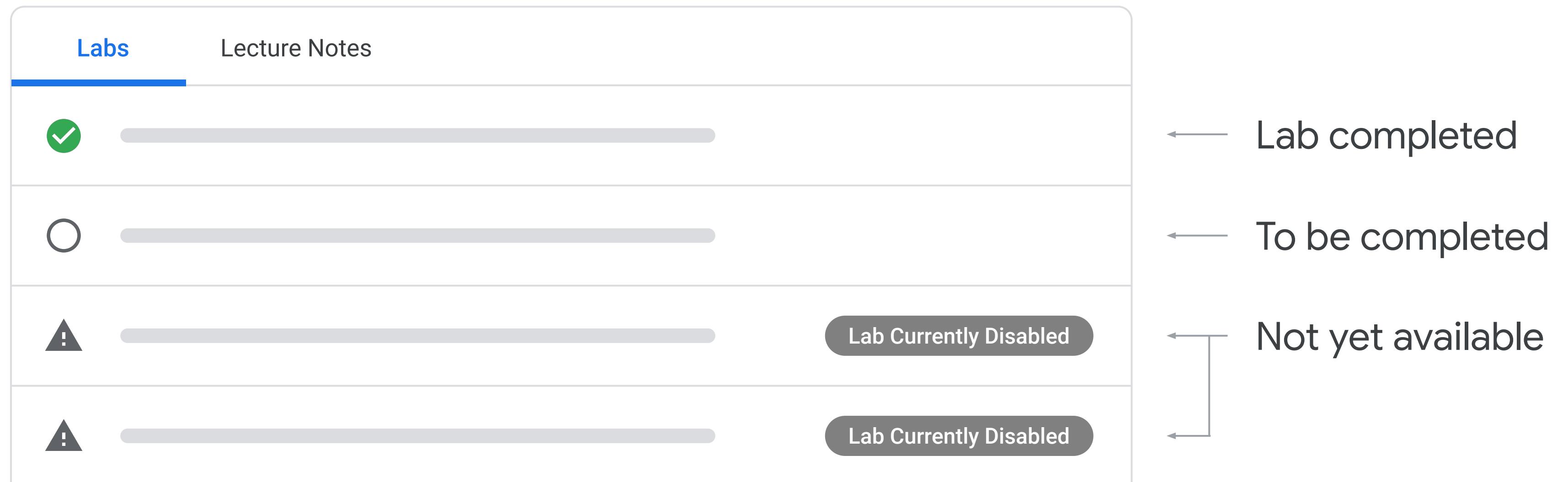


## Access issues

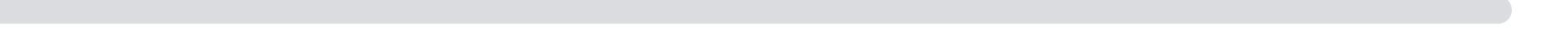
The process to open Qwiklabs can differ based on credentials used. Please reach out to your trainer if you have any access issues.

# View your labs

Do **NOT** launch a lab until instructed to do so!



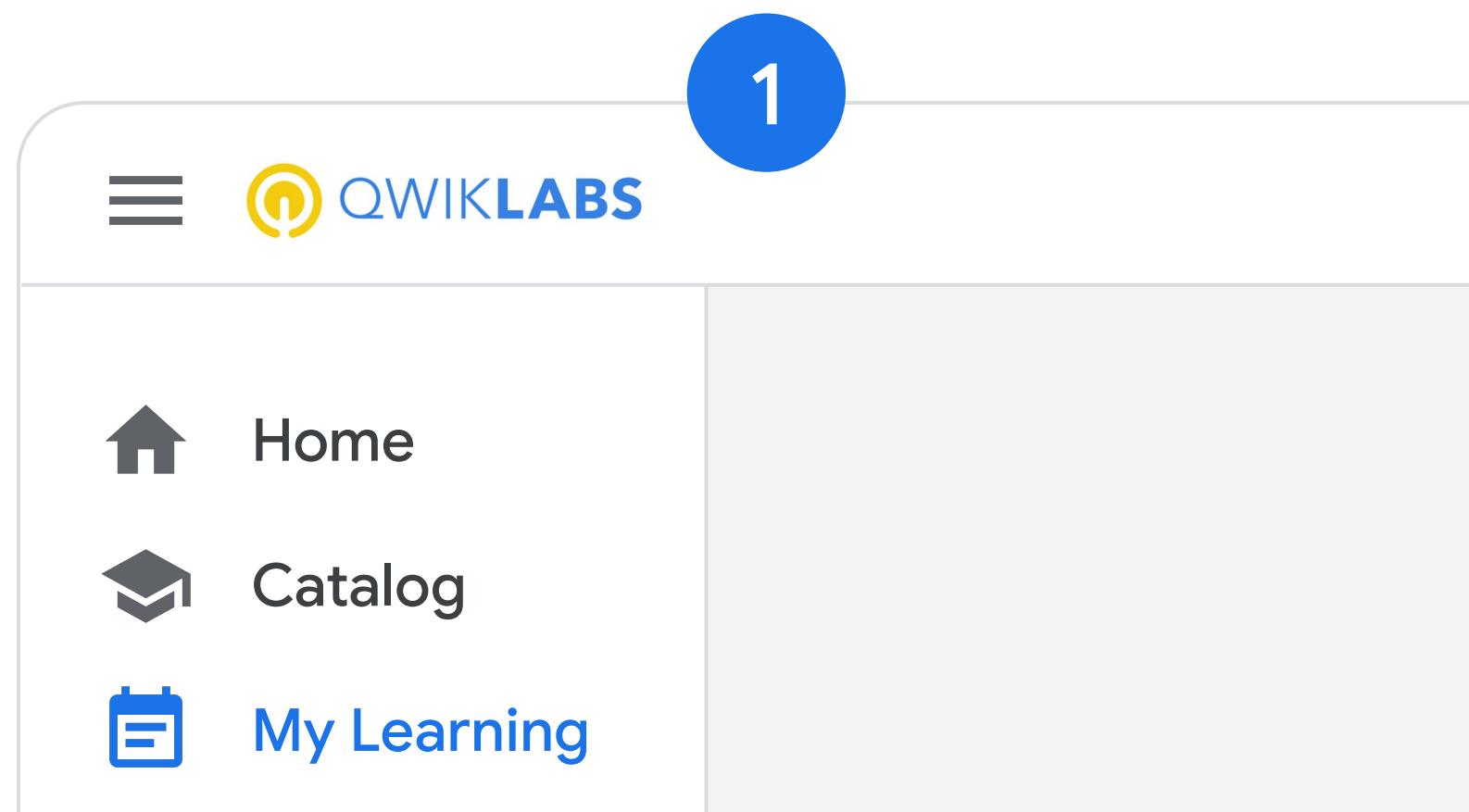
# View lecture notes

Labs	Lecture Notes
01	 <a href="#"></a>
02	 <a href="#"></a>
03	 <a href="#"></a>
04	 <a href="#"></a>

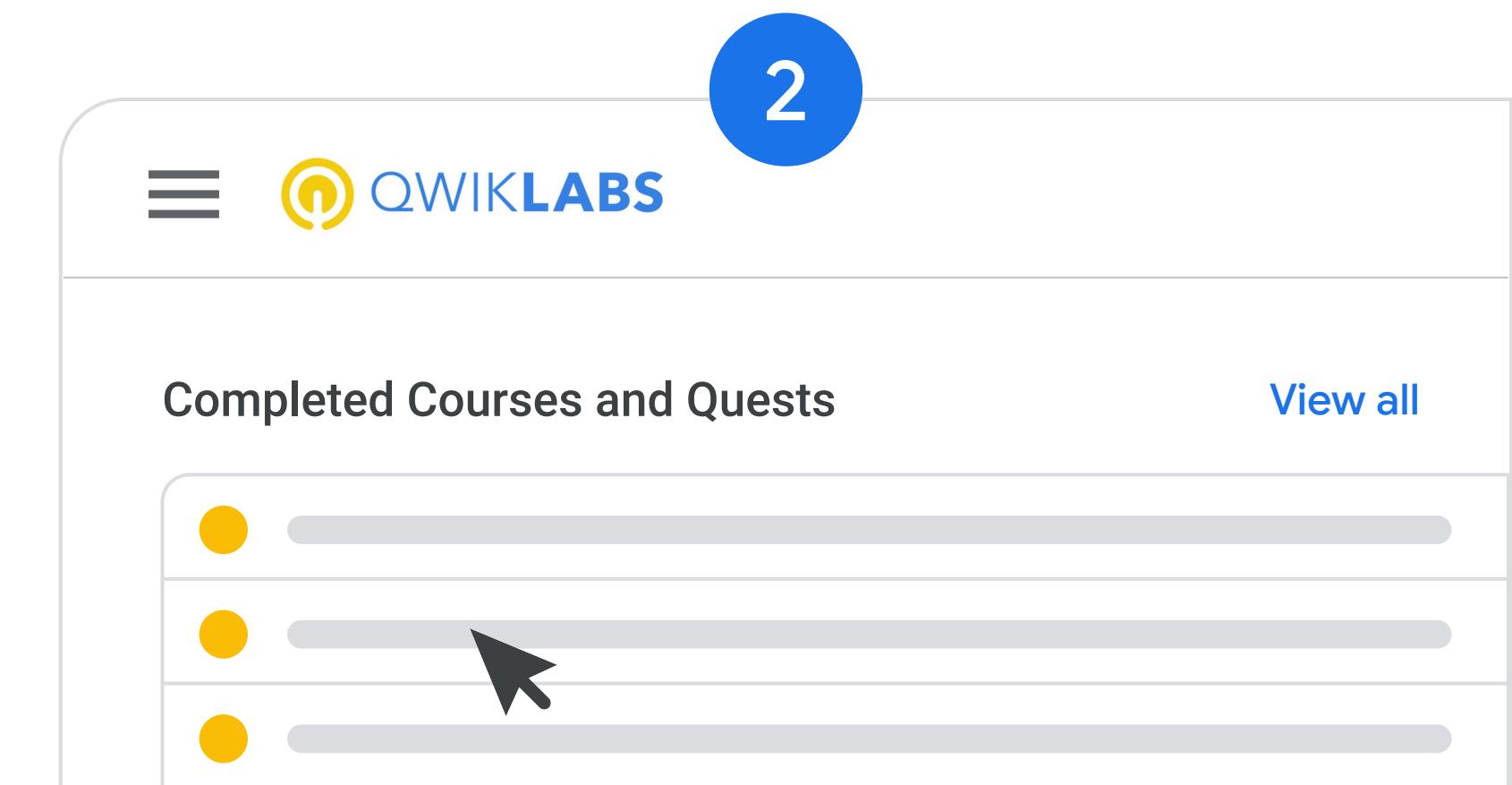
You can download  
these as PDF files

# End of class - Materials

Materials are available for 2 years



Click on My Learning in the left-hand navigation bar



Select the class from the **Completed Courses** list



# Generative AI on Vertex AI

# In this module, you learn to ...

- 01 Differentiate between machine learning in general and generative AI
- 02 Automate ML tasks using Vertex AI on Google Cloud
- 03 Choose from the available Generative AI options on Google Cloud
- 04 Explore text and code Chat AI use cases

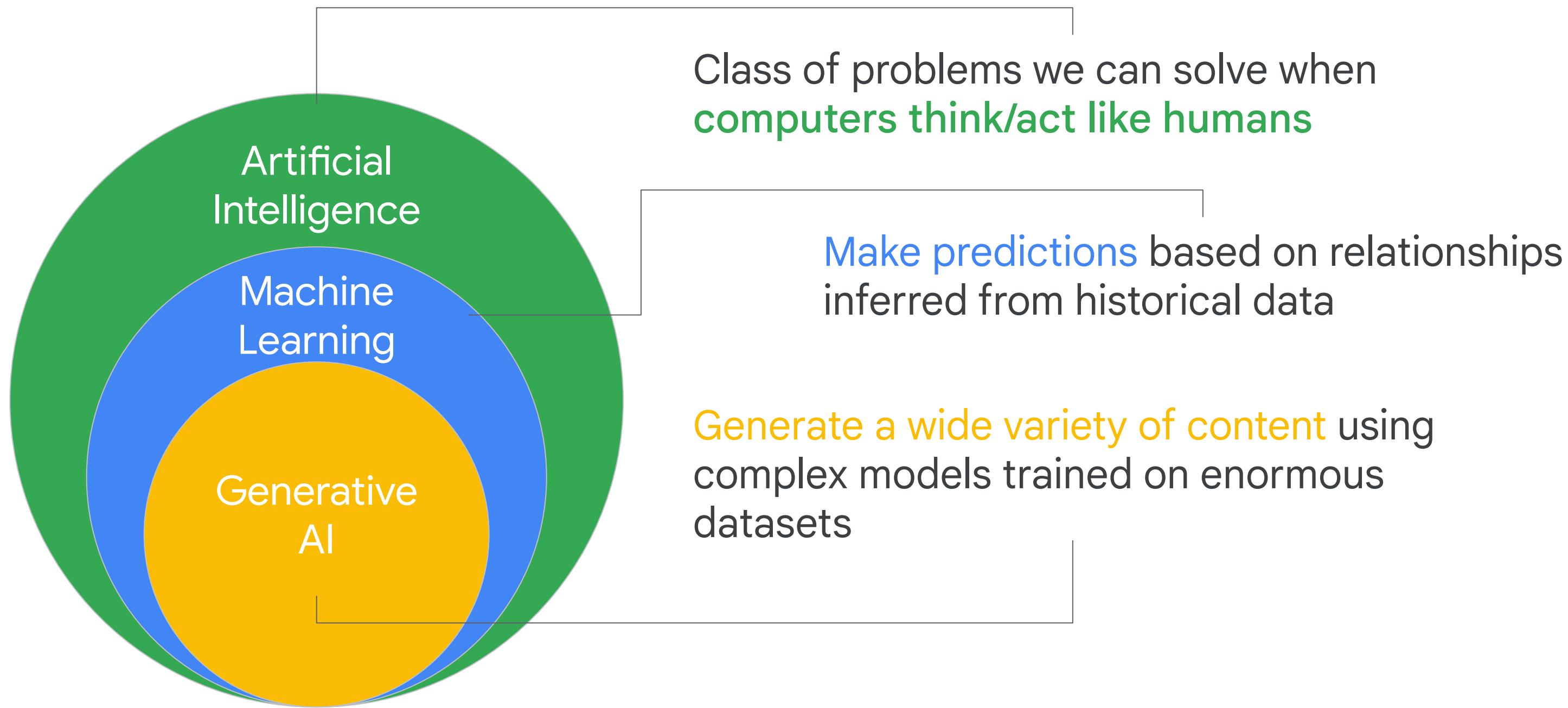


# Topics

- 01 What is Generative AI
- 02 Vertex AI on Google Cloud
- 03 Generative AI Options on Google Cloud
- 04 Introduction to the Course Use Case



# Machine Learning is a type of AI, and Generative AI (GenAI) is a type of machine learning



# Machine learning allows computers to learn without explicit programming

- In traditional programming, the programmer must define the steps that the computer should take to perform a task
- In machine learning, algorithms are trained to make predictions using historical data
  - Computers iterate over the algorithm making adjustments to find the best solution

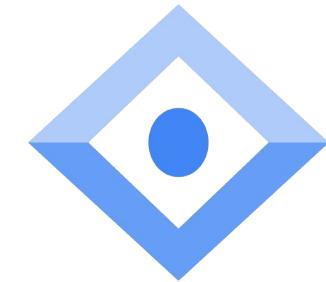


# Machine learning use cases include:

- Image recognition
- Sentiment analysis
- Speech recognition
- Fraud detection
- Customer segmentation
- Recommendation systems



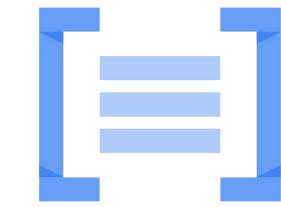
# Google has been a pioneer in machine learning for many years



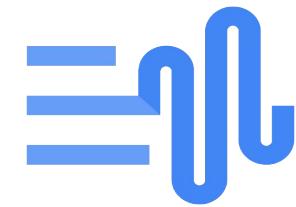
Cloud  
Vision API



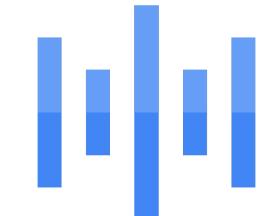
Cloud  
Translation API



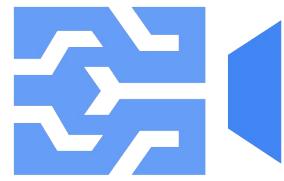
Natural  
Language API



Text to  
Speech



Speech  
to Text



Video  
Intelligence API

# Google Vision API tells you what's in a picture

- Detects objects and landmarks
- Labels photos
- Recognizes text
- Detects inappropriate content

Landmarks      Labels      Text      Properties      Safe Search

Stonehenge 85%

Stonehenge Cursus

A303

Google Map data ©2023

Location: E1 170000, N 926215

IMG\_20180218\_131834.jpg

<https://cloud.google.com/vision/docs/drag-and-drop>

Google Cloud

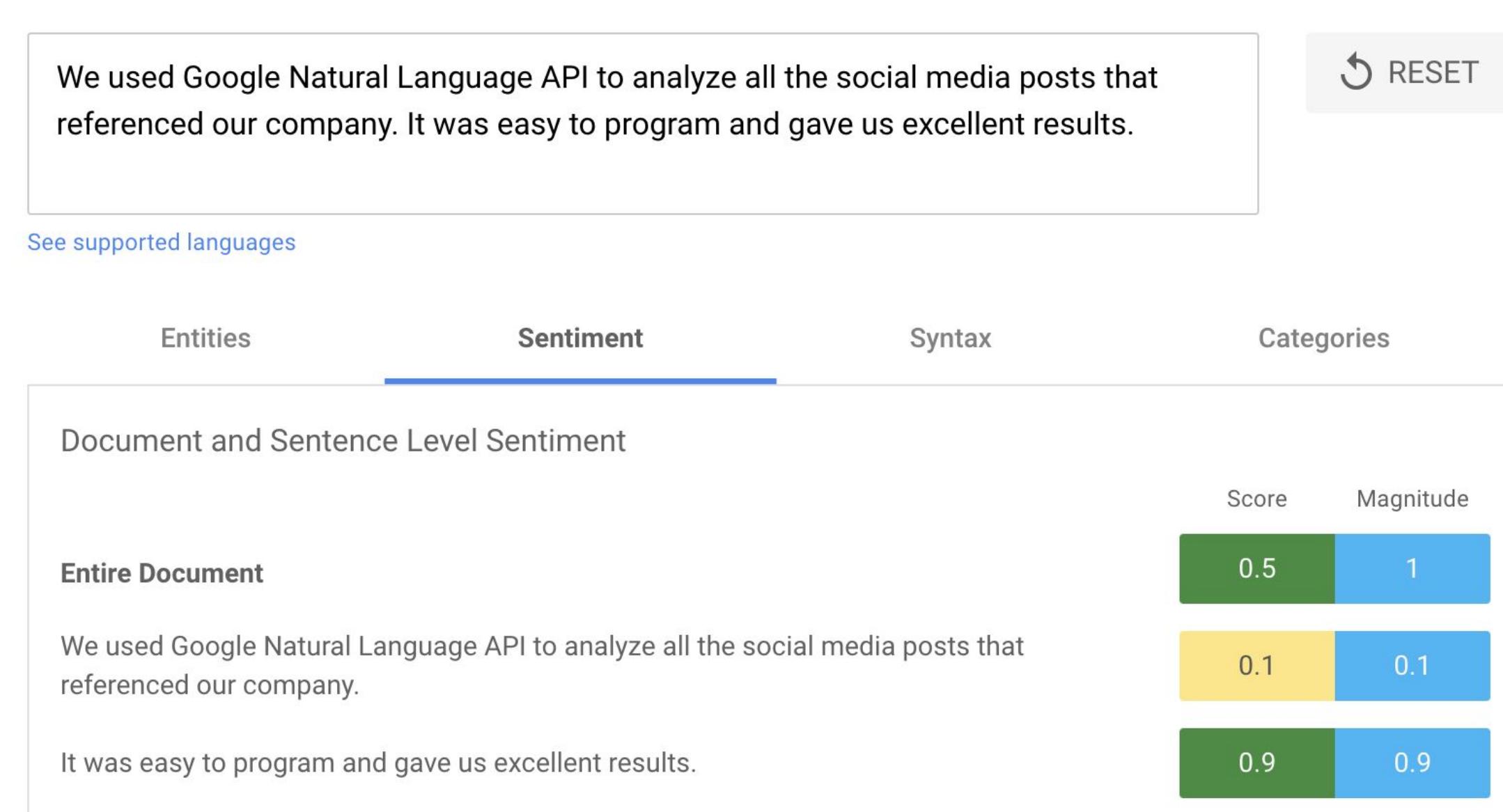
# You can use Google ML to translate between different languages

The screenshot shows the Google Translate web interface. At the top, there's a navigation bar with the Google Translate logo, a menu icon, and a user profile icon. Below the navigation bar are four tabs: Text (selected), Images, Documents, and Websites. Underneath these tabs, there are language selection dropdowns. The left side shows the source text in English: "Google Translate is a multilingual neural machine translation service developed by Google to translate text, documents and websites from one language into another." The right side shows the translated text in Spanish: "Google Translate es un servicio de traducción automática neuronal multilingüe desarrollado por Google para traducir texto, documentos y sitios web de un idioma a otro." Below the text boxes are various interaction icons like microphone, speaker, and share.

<https://translate.google.com/>

Google Cloud

# Natural Language API can analyze text, extract entities, and perform sentiment analysis



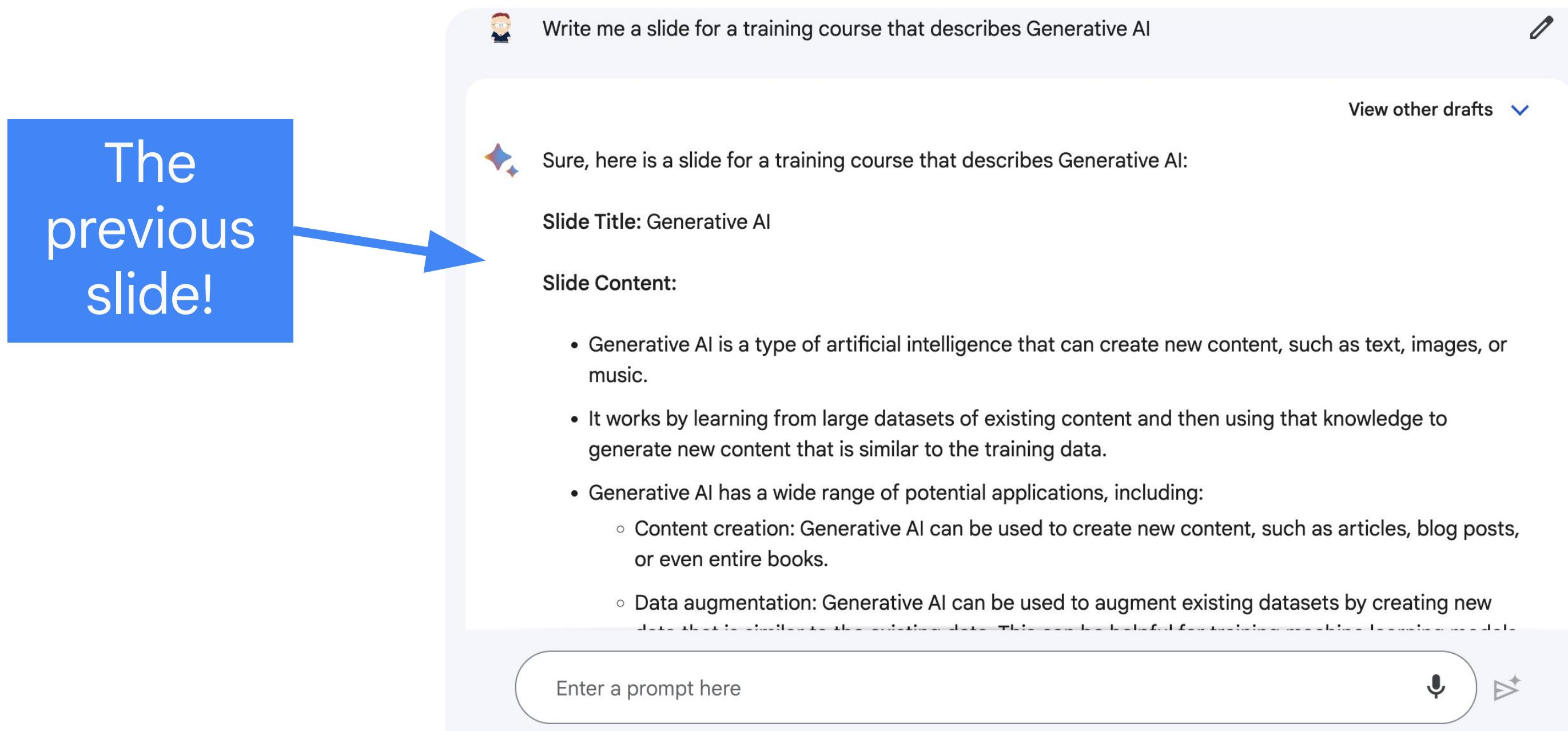
<https://cloud.google.com/natural-language/>

Google Cloud

# Generative AI

- Generative AI is a type of artificial intelligence that can create new content, such as text, images, or music
- It works by learning from large datasets of existing content and then using that knowledge to generate new content that is similar to the training data
- Generative AI has a wide range of potential applications, including:
  - Content creation: Generative AI can be used to create new content, such as articles, blog posts, or even entire books.
  - Data augmentation: Generative AI can be used to augment existing datasets by creating new data that is similar to the existing data. This can be helpful for training machine learning models.
  - Creative applications: Generative AI can be used to create new and innovative art, music, and other creative content.

# Bard is a Google consumer-oriented generative AI tool for creating custom text-based content



The previous slide!

Write me a slide for a training course that describes Generative AI

Sure, here is a slide for a training course that describes Generative AI:

Slide Title: Generative AI

Slide Content:

- Generative AI is a type of artificial intelligence that can create new content, such as text, images, or music.
- It works by learning from large datasets of existing content and then using that knowledge to generate new content that is similar to the training data.
- Generative AI has a wide range of potential applications, including:
  - Content creation: Generative AI can be used to create new content, such as articles, blog posts, or even entire books.
  - Data augmentation: Generative AI can be used to augment existing datasets by creating new

Enter a prompt here

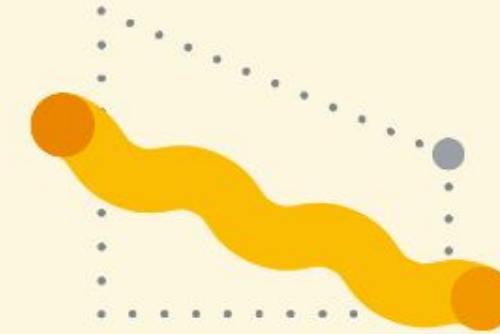
<https://bard.google.com/>

Google Cloud

# There are huge benefits of generative AI



Increase in efficiency & productivity



Reduce costs for your organization



Automate monotonous tasks

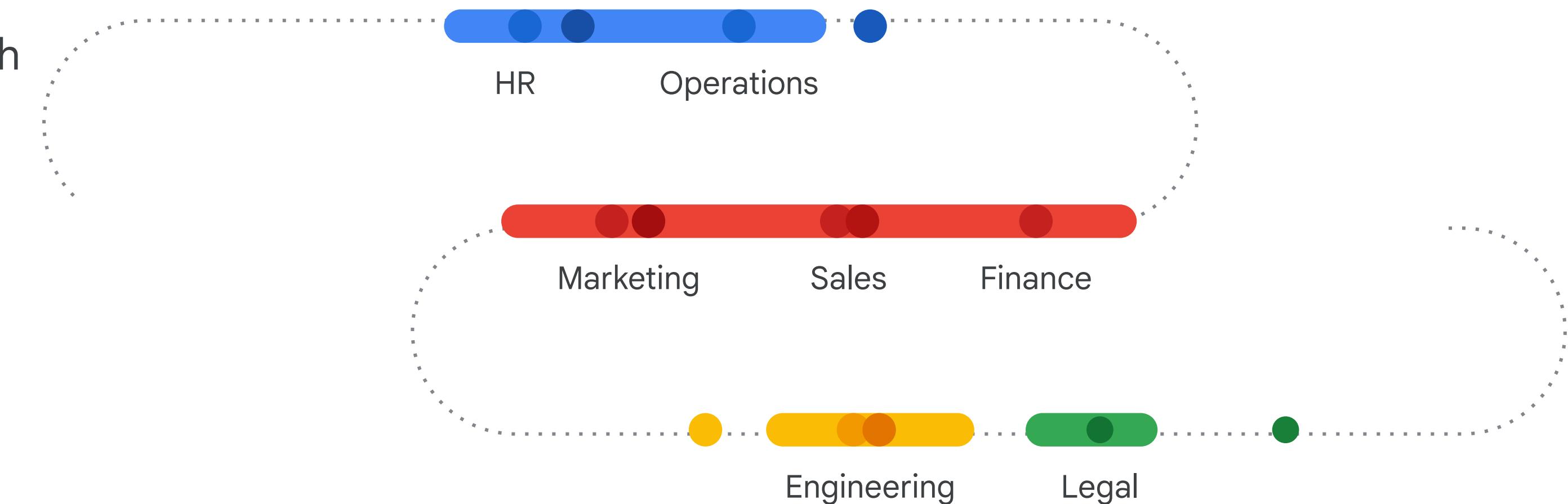
# There are also challenges with generative AI

- Can be difficult to control the quality of generated content
- Can be difficult to ensure that generated content is accurate
  - Untrue statements can be presented in a confident manner
  - These are known as hallucinations in generative AI terms
- Can be difficult to ensure that generated content is not offensive or harmful



# There are many potential use-cases for generative AI across all aspects of an organization

- Content creation
- Marketing and advertising
- Customer service
- Education and research
- Interactive chat
- Code completion
- Many more...



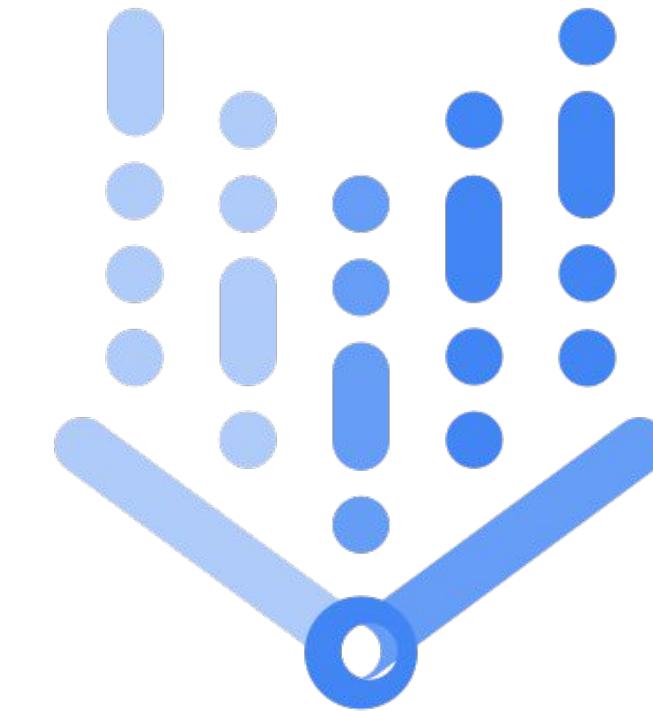
# Topics

- 01 What is Generative AI
- 02 Vertex AI on Google Cloud
- 03 Generative AI Options on Google Cloud
- 04 Introduction to the Course Use Case



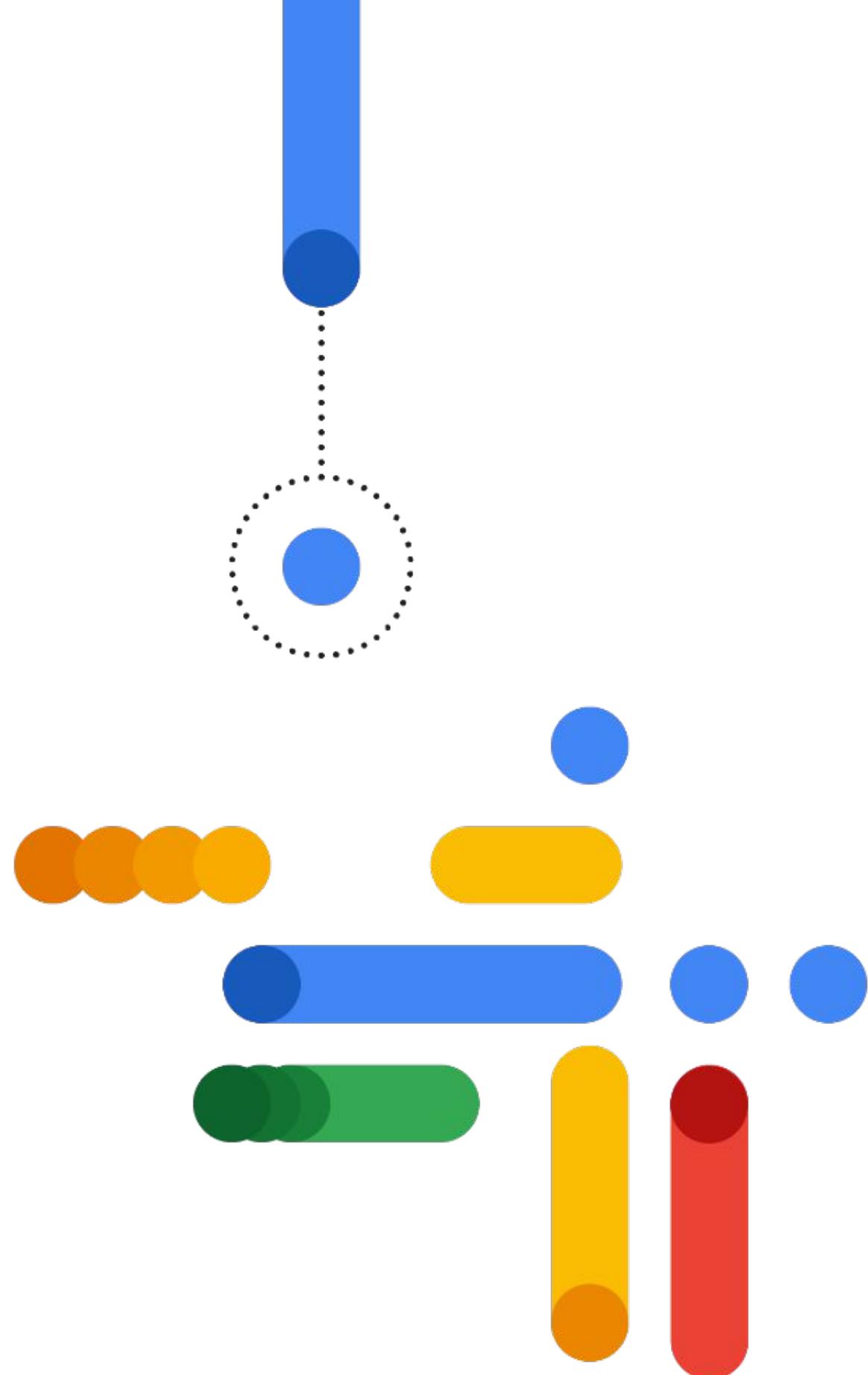
# Vertex AI is a machine learning platform that helps you build, deploy, and manage ML models

- Combines data engineering, data science, and ML engineering workflows
- Provides several options for model training and deployment
  - AutoML
  - Custom Training
  - Model Garden
  - Generative AI
- Uses fully-managed infrastructure that you can customize based on your performance and budget needs
- Supports Python, the Google Cloud Console, the gcloud command line tool, client libraries, and Terraform



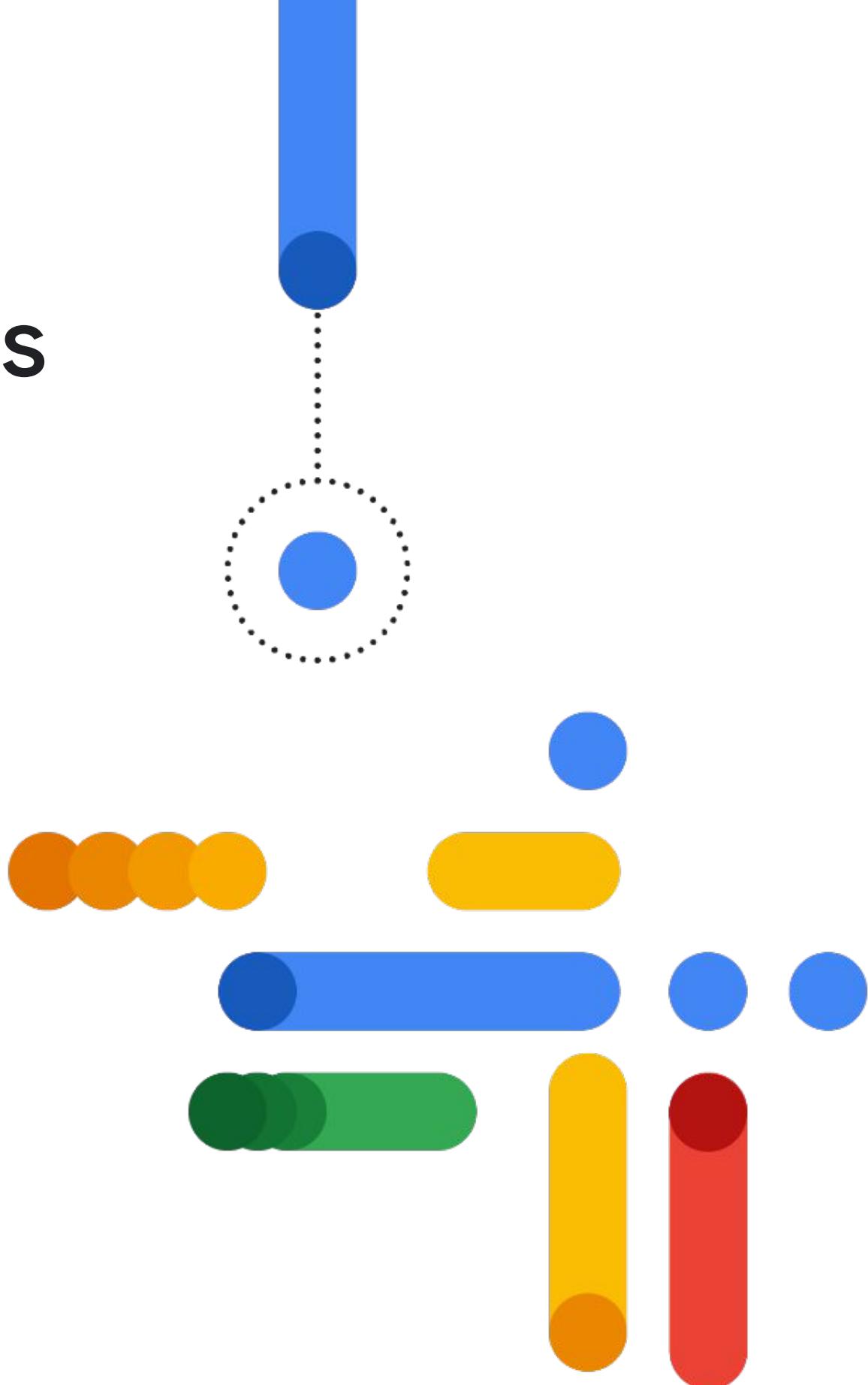
# Vertex AI simplifies model training with AutoML

- Supports datasets created with image, text, tabular, and video data
- Zero-code, custom-model training for various use cases
  - Image detection and classification
  - Text classification, entity extraction, and sentiment analysis
  - Linear regression, classification, and forecasting from tabular datasets
  - Video action recognition and object tracking
- Automated deployment of models to service endpoints managed by Google



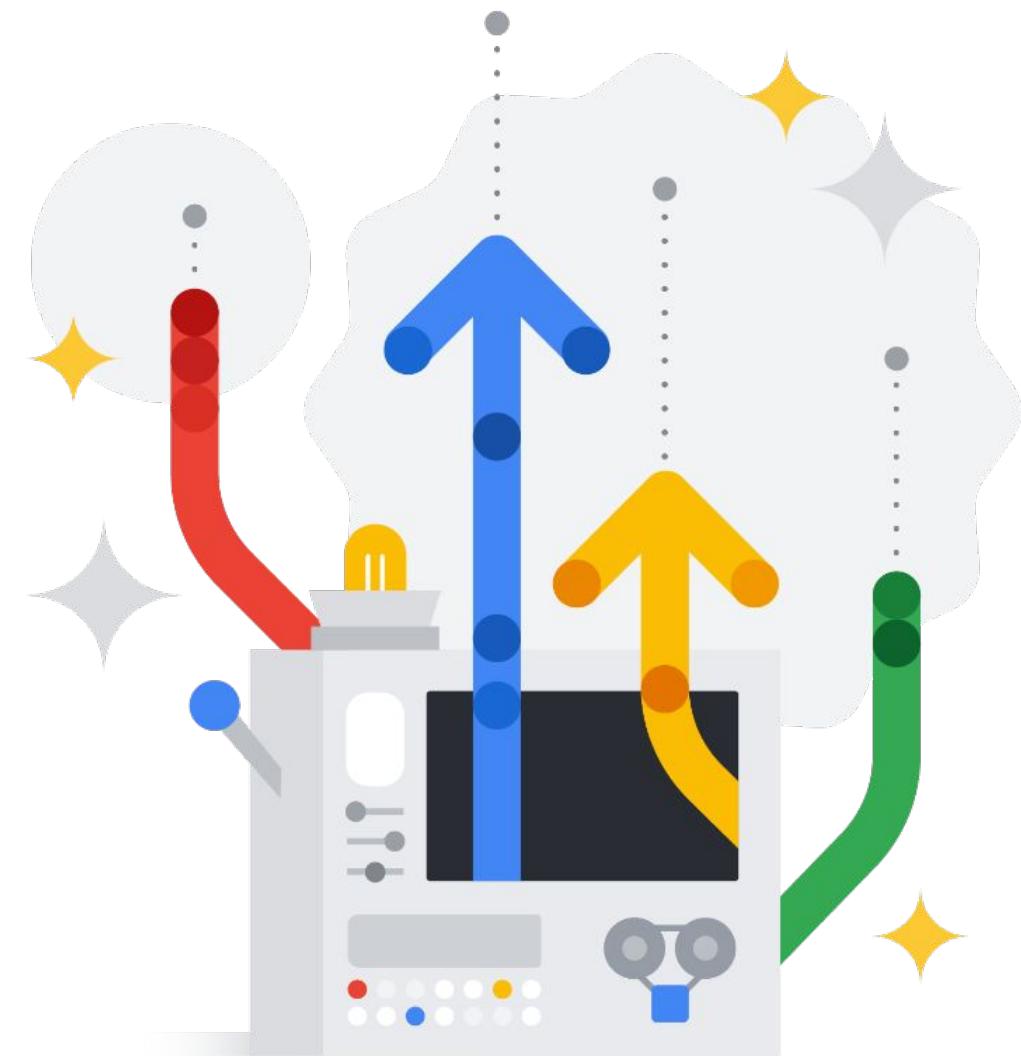
# Model Garden provides a catalog of pre-trained models to build AI applications

- Foundation models are pre-trained multitask large models that can be tuned or customized for specific tasks
- Fine-tunable models are models that you can fine-tune using a custom notebook or pipeline
- Task-specific solutions are pre-built models that are ready to use and can be customized using your own data
- Model garden provides and easy to use interface for searching for models
  - Documentation and code samples are provided



# Generative AI Studio simplifies prompting, tuning, and deploying Google foundational models

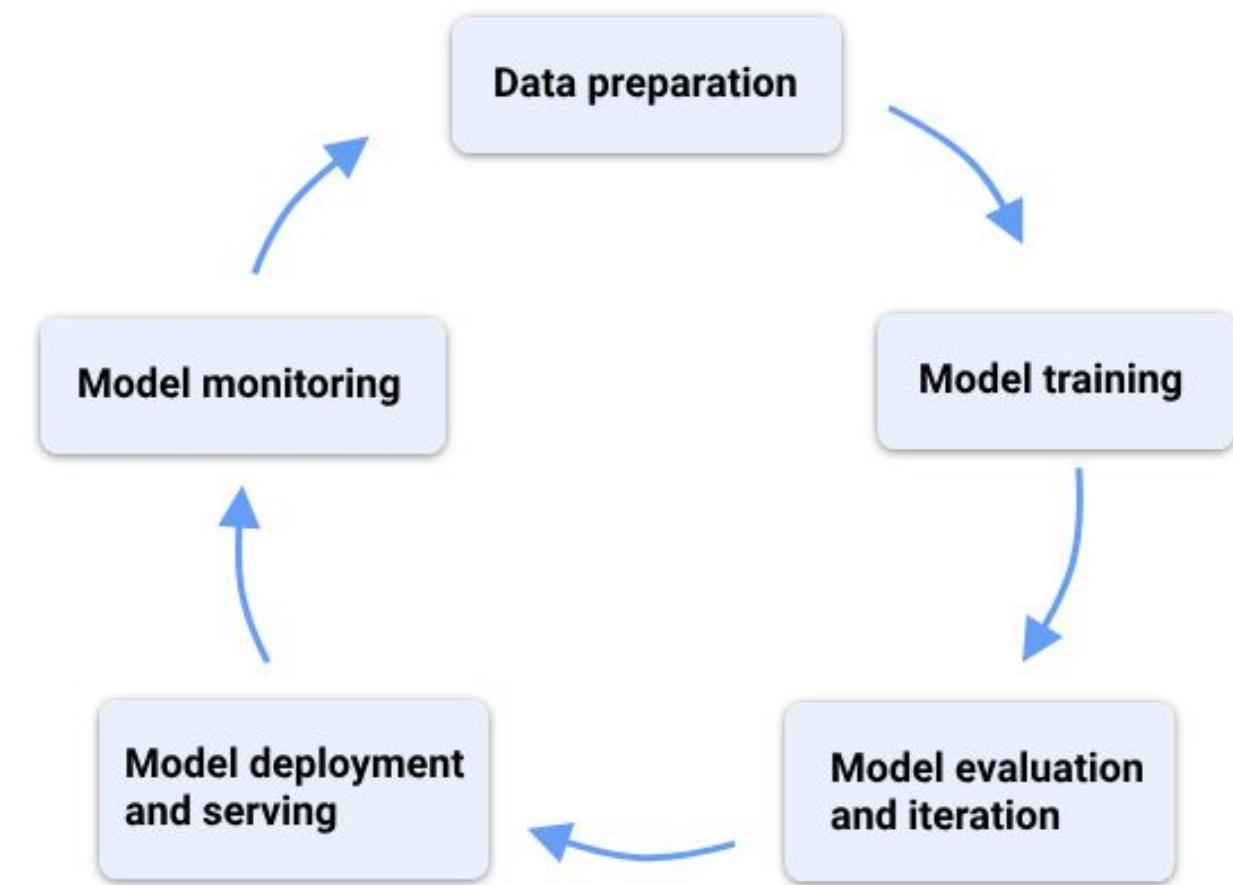
- Low-code, customizable solution for building generative AI solutions
- Easy access to Google foundational models
  - PaLM 2 (Pathways Language Model) for text and chat
  - Codey for code generation, completion, and chat
  - Chirp for speech
  - Imagen for text to image generation
  - Options for building question-answer and conversational chat applications



# Vertex AI helps with all steps in an AI workflow

- Data preparation: clean and explore data, apply transformations and feature engineering
- Model training: choose a training method, train and tune the model
- Model evaluation and iteration: evaluate the model, make adjustments and iterate
- Model serving: deploy the model to production and get predictions
- Model monitoring: monitor the performance of the deployed model and retrain as needed

## Machine learning workflow



# Vertex AI features by task

01

## Data Preparation

Datasets  
Labeling tasks  
Feature store  
AI Pipelines

02

## Model Training

Auto ML  
Model Garden  
Gen AI Studio  
Online Training  
Workbench

03

## Model Evaluation

Built-in metrics  
Experiments

04

## Model Deployment

AI Pipelines  
Endpoints  
Batch predictions

05

## Model Monitoring

Operations Suite  
Monitoring  
Logging

# Topics

- 01 What is Generative AI
- 02 Vertex AI on Google Cloud
- 03 Generative AI Options on Google Cloud
- 04 Introduction to the Course Use Case



# Generative AI use cases with Vertex AI



## Language

Writing  
Summarization  
Ideation  
Classification  
Sentiment analysis  
Extraction  
Customer chat

## Code

Code generation  
Code completion  
Code chat  
Code conversion

## Speech

Speech to text  
Text to speech

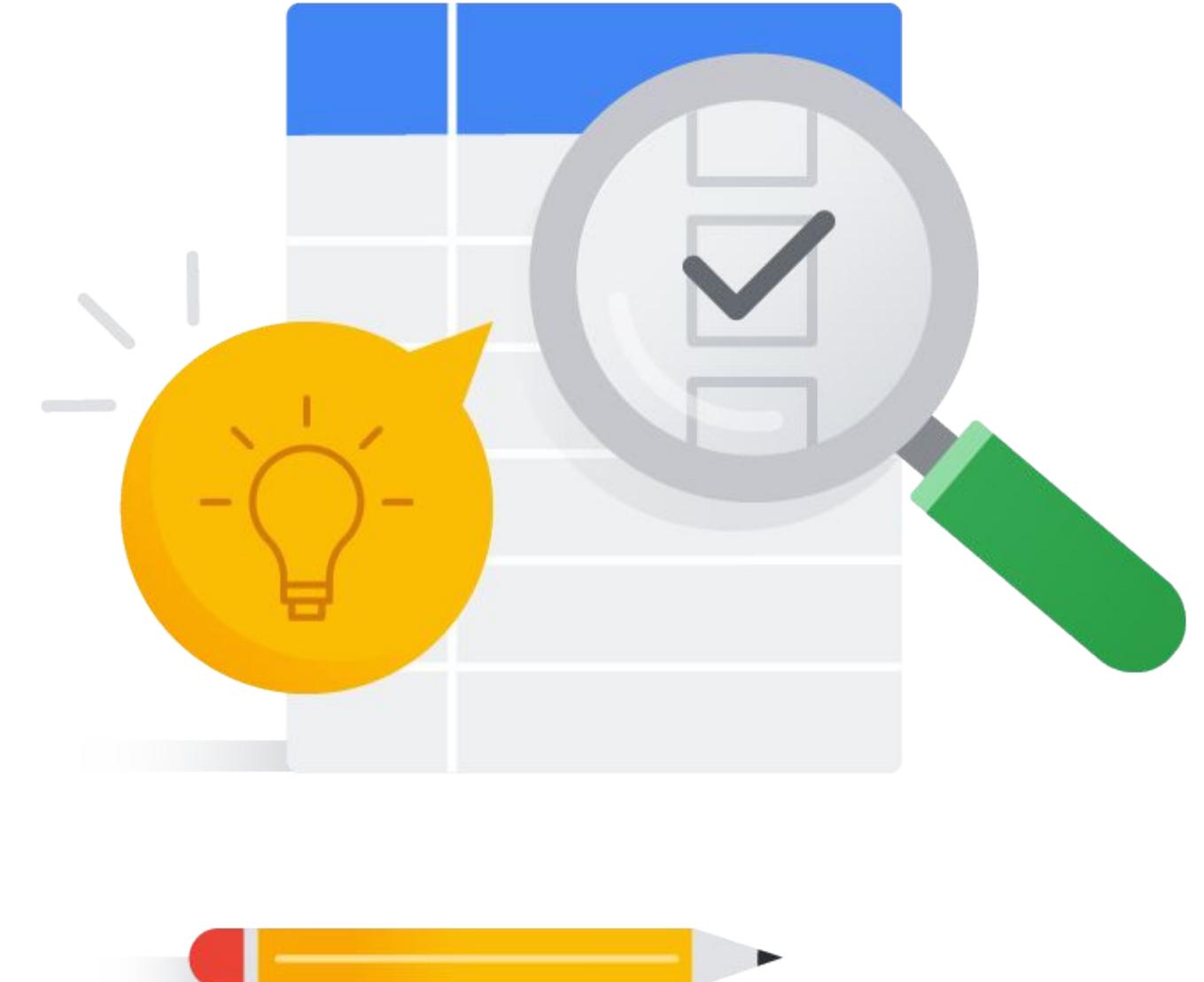
# Group Discussion

10 min



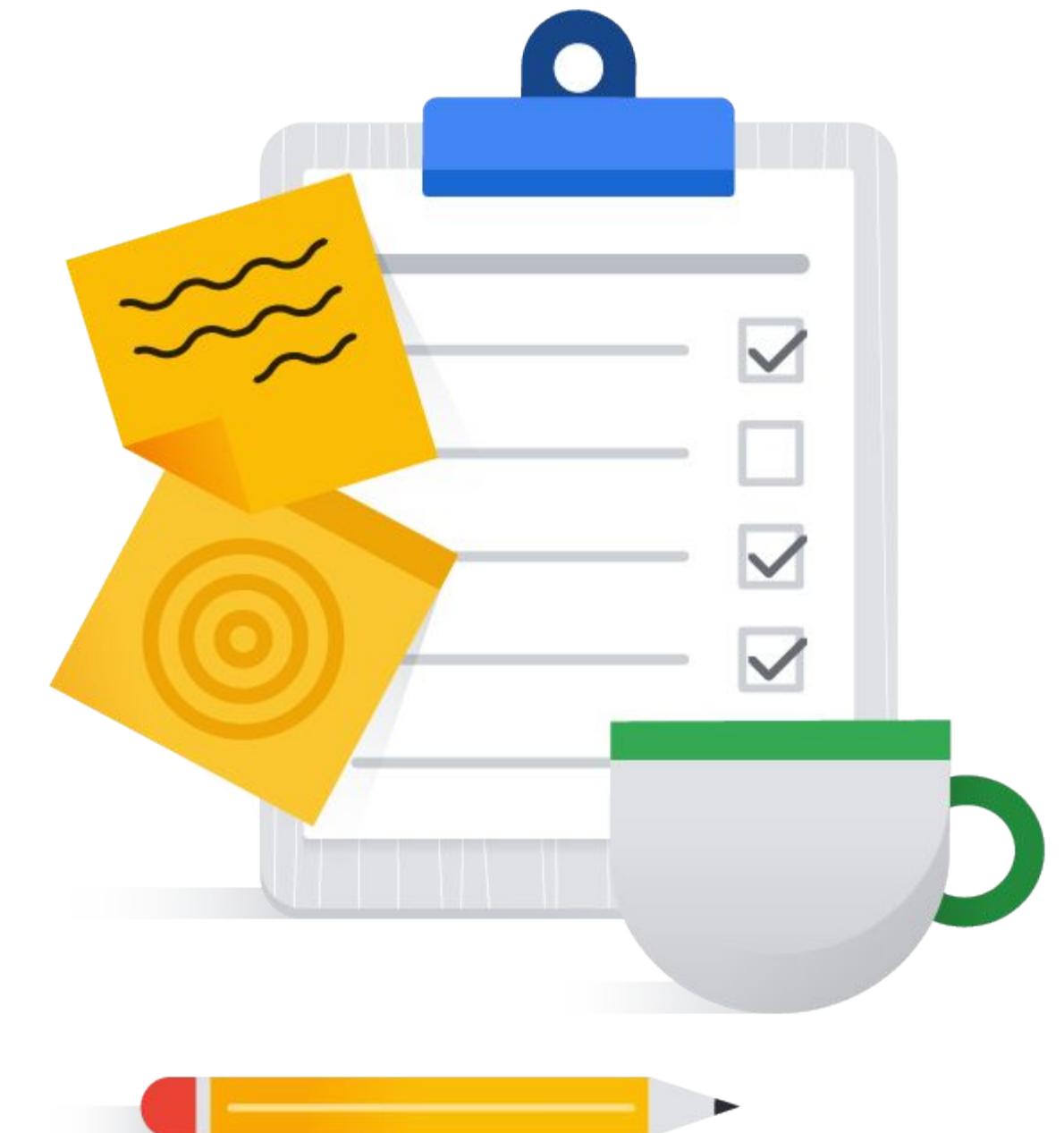
Take a few minutes to come up with some use cases for generative AI that would benefit you or your organization

You will share your ideas with the class



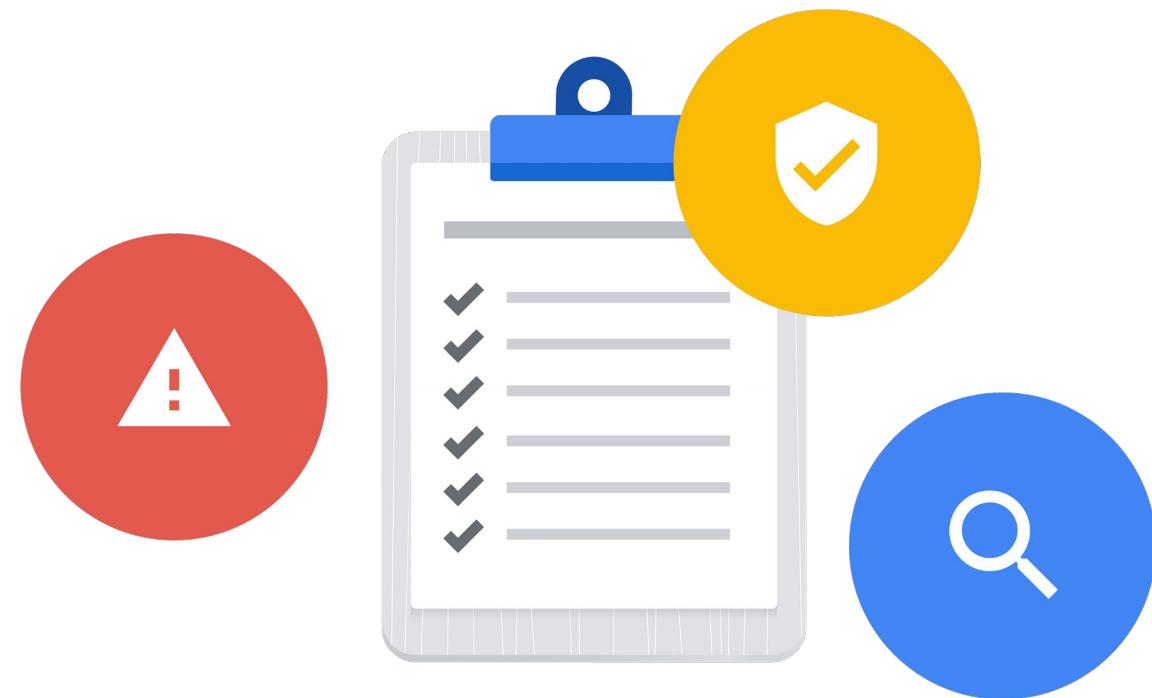
# Topics

- 01 What is Generative AI
- 02 Vertex AI on Google Cloud
- 03 Generative AI Options on Google Cloud
- 04 Introduction to the Course Use Case

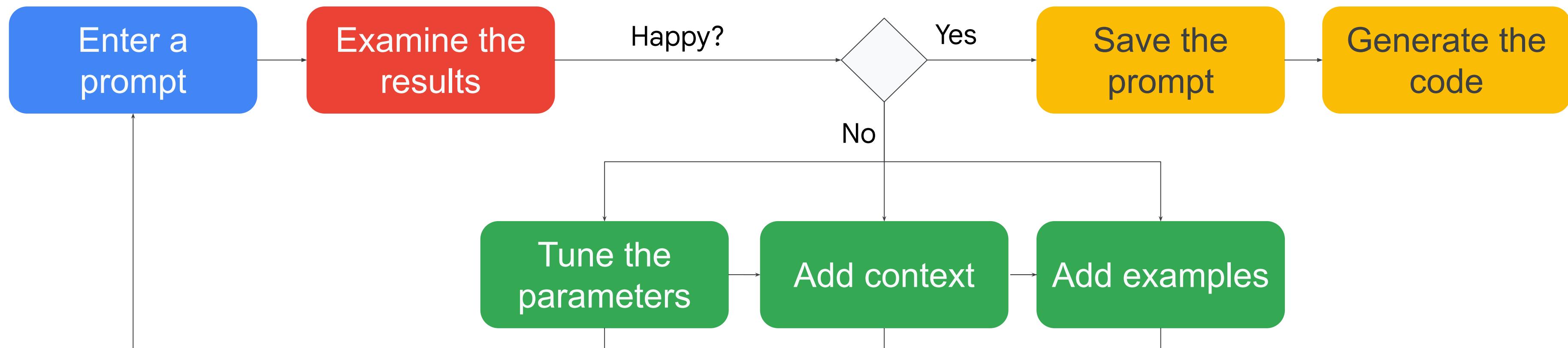


# This course focuses on using generative AI for chat-based text and code use cases

- Q&A
- Code generation
- Content creation
- Ideation
- Summarization
- Classification
- etc.



# You will learn the process of designing, tuning, and deploying prompts to generate content for each use case

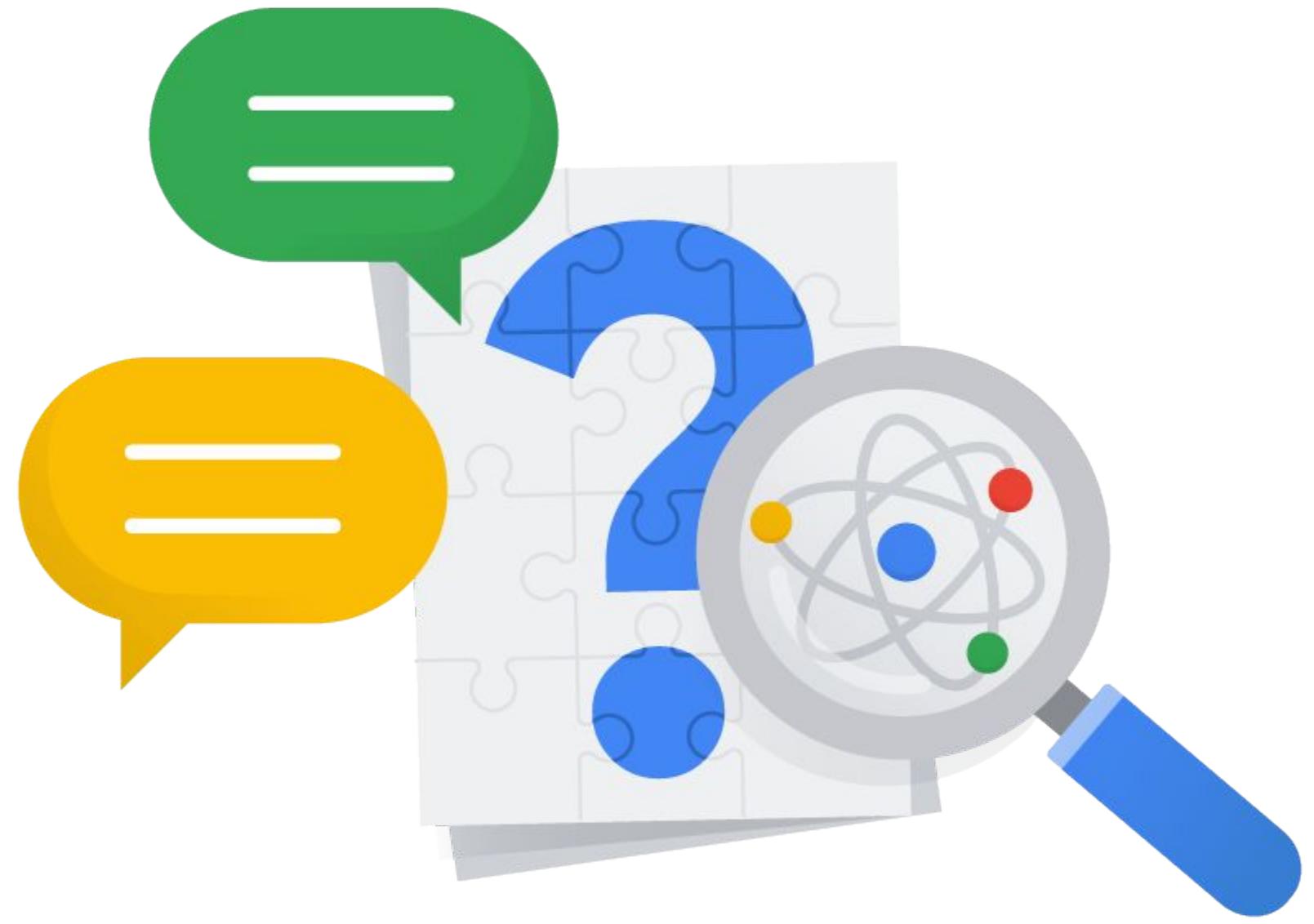


# In this module, you learned to ...

- 01 Differentiate between machine learning in general and generative AI
- 02 Automate ML tasks using Vertex AI on Google Cloud
- 03 Choose from the available Generative AI options on Google Cloud
- 04 Explore text and code Chat AI use cases



# Questions and answers



# Quiz question

What is a type of artificial intelligence that allows computers to learn based on data without having to be explicitly programmed?

- A: Generative AI
- B: Machine learning
- C: Linear regression
- D: Linear classification

# Quiz question

What is a type of artificial intelligence that allows computers to learn based on data without having to be explicitly programmed?

- A: Generative AI
- B: Machine learning
- C: Linear regression
- D: Linear classification

# Quiz question

What is a type of artificial intelligence that can create new content, such as text, images, or music?

- A: Generative AI
- B: Machine learning
- C: Linear regression
- D: Linear classification

# Quiz question

What is a type of artificial intelligence that can create new content, such as text, images, or music?

- A: Generative AI
- B: Machine learning
- C: Linear regression
- D: Linear classification

# Quiz question

- When generative AI systems just make something up that isn't true, it is called what?
- A: A lie
  - B: A hallucination
  - C: A bug
  - D: It is not possible for that to happen

# Quiz question

- When generative AI systems just make something up that isn't true, it is called what?
- A: A lie
  - B: A hallucination
  - C: A bug
  - D: It is not possible for that to happen

# Quiz question

List some use cases for generative AI:

---

---

---

---

# Quiz question

List some use cases for generative AI:

Customer Service Chat

Writing content

Ideation

Summarization

Classification

Sentiment analysis

....





# Gen AI Studio

# In this module, you learn to ...

- 01 Use Gen AI Studio to implement generative AI use cases
- 02 Explore the available models and sample use cases in Gen AI Studio
- 03 Write and test prompts using the Google Cloud Console
- 04 Protect your data while using Gen AI Studio



# Topics

01

Gen AI Studio

02

Designing and testing prompts

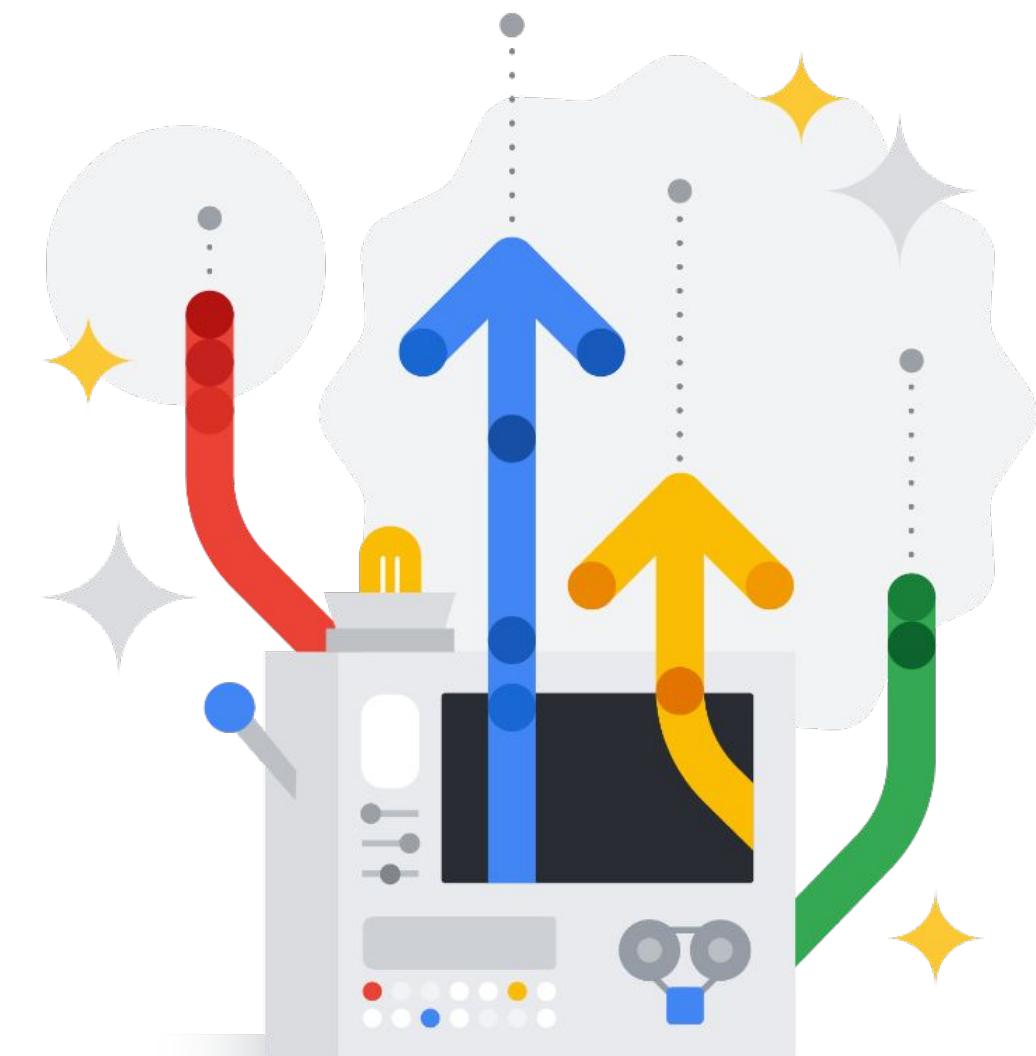
03

Data governance in GenAI Studio



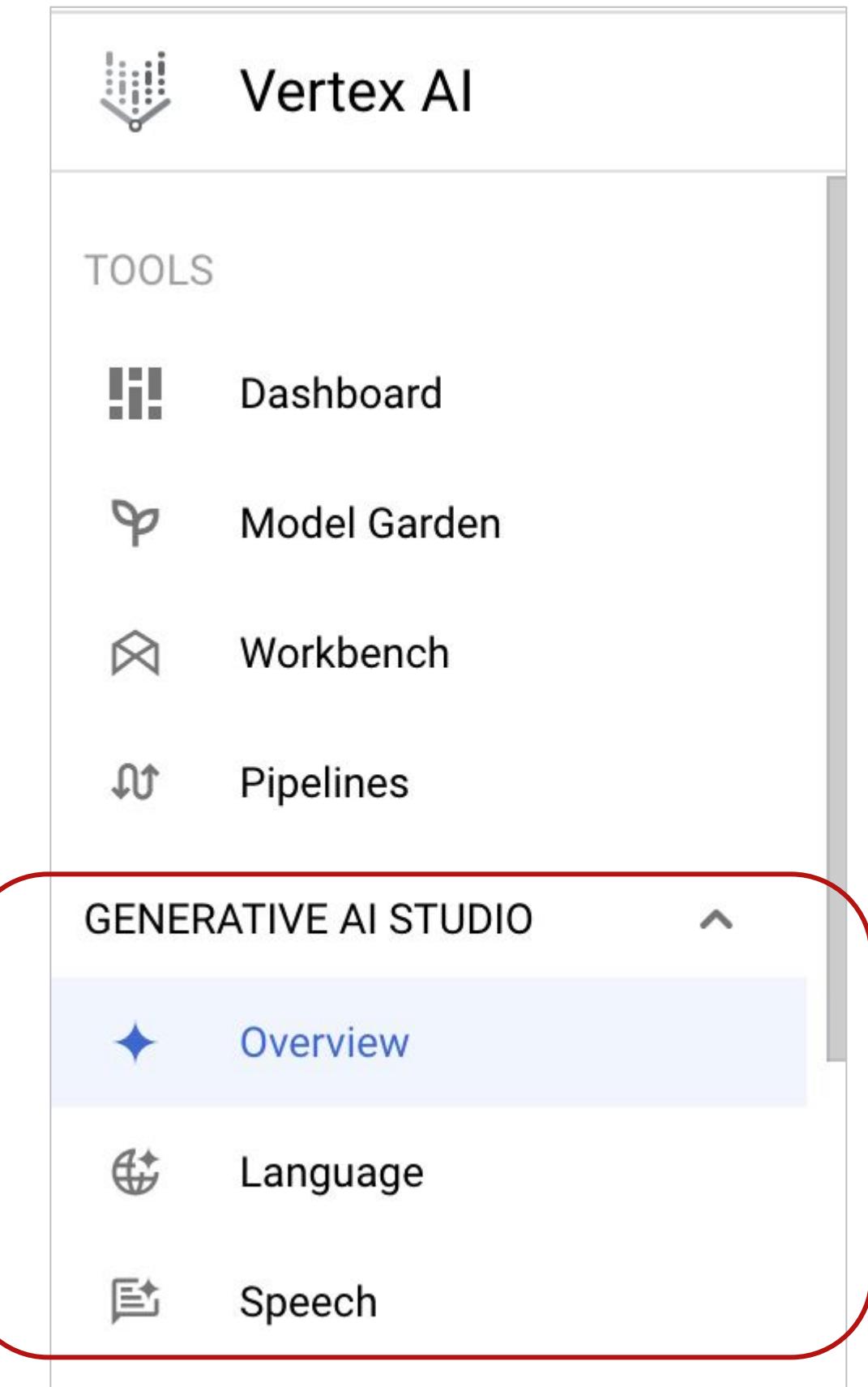
# Generative AI Studio provides an intuitive interface to prompt, tune, and deploy Google foundation models

- Supports Google foundational models
  - PaLM 2 for text and chat
  - Codey for code generation and chat
  - Chirp for speech to text and text to speech
  - Imagen for text to image generation
- Allows users to easily experiment with prompts
  - Simple, intuitive design
  - Easily experiment with parameters
  - Add context and examples

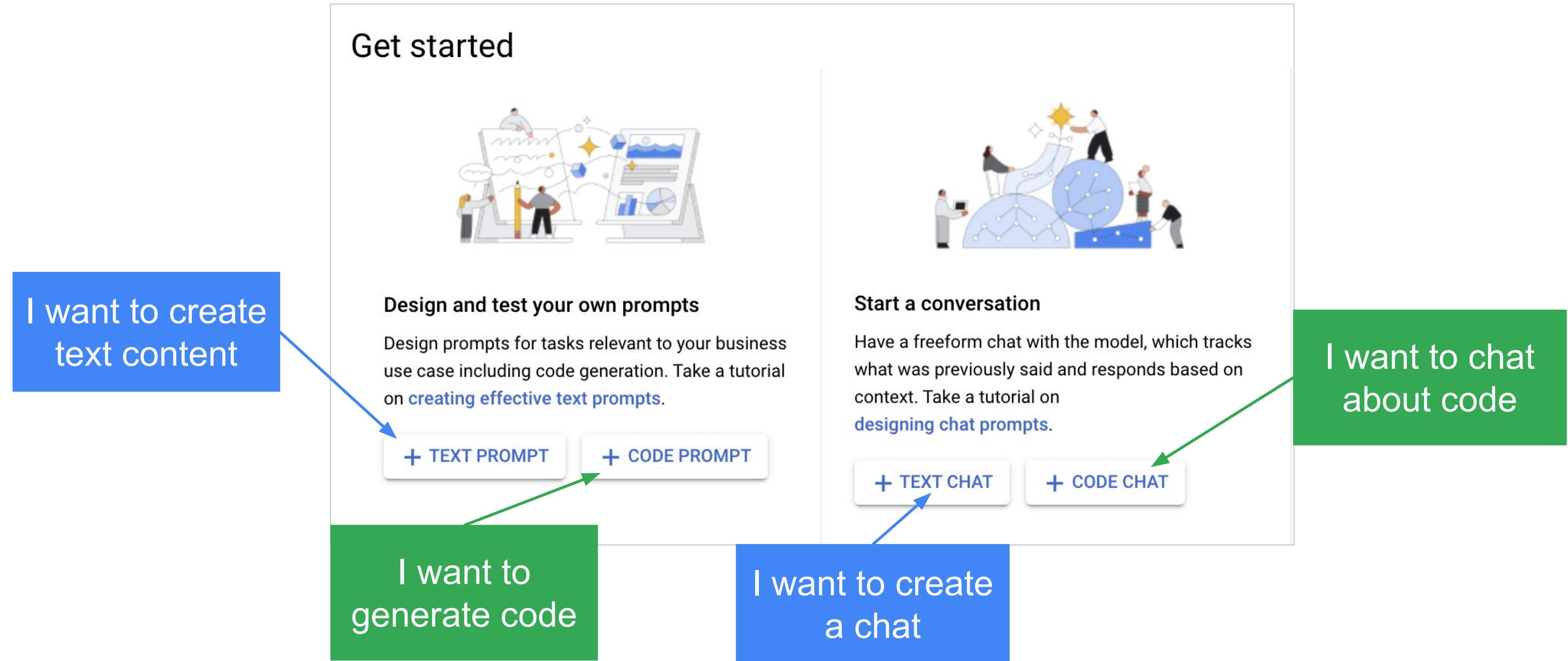


# Gen AI Studio is available as a feature of Google Cloud Vertex AI

- Choose from Language or Speech models



# To get started, choose a task based on your goal



# Gen AI Studio UI for Prompt Design

The screenshot illustrates the Gen AI Studio UI for Prompt Design, showing the workflow for creating and managing prompts.

**Enter a Prompt**: A blue callout points to the "Prompt" section where the user can enter their query. In this example, the question "Who was Grace Hopper?" is entered.

**Select the foundational model**: A blue callout points to the "Model" dropdown menu, which is set to "text-bison@001".

**Adjust the parameters**: A blue callout points to the "Temperature" slider (set to 0.2), "Token limit" slider (set to 256), "Top-K" slider (set to 40), "Top-P" slider (set to 0.8), and "Safety filter threshold" dropdown (set to "Block few").

**The response**: A blue callout points to the "Response" section, which displays the generated text: "Grace Hopper was an American computer scientist and United States Navy rear admiral. She was one of the pioneers of computer programming and is credited with coining the term "debugging". Hopper was born in New York City in 1906. She attended Vassar College and earned a master's degree in mathematics from Yale University. In 1943, Hopper joined the United States Navy Reserve. She was assigned to the Bureau of Ordnance Computation Project where she worked on the Mark I computer. In 1945, Hopper was one of the first programmers for the Harvard Mark II computer."

**Submit the prompt**: A blue callout points to the "SUBMIT" button at the bottom of the interface.

**Save the model**: A blue callout points to the "SAVE" button in the top right corner.

# Gen AI Studio UI for Chat

The diagram illustrates the Gen AI Studio UI for Chat interface, showing the flow from entering a prompt to saving the model.

**Add Context**: Points to the "Context (Optional)" input field where "Write your context here..." is entered.

**Add Examples**: Points to the "Examples (Optional)" section, which includes a "USER" row with "Write an input to add a new row" and an "AI" row with "Write an output to add a new row".

**The response**: Points to the AI-generated response: "Hawaii is a state of the United States located in the Pacific Ocean. It is the only state in the United States that is not located in North America or South America. Hawaii is an archipelago of eight main islands, as well as several smaller islands and atolls. The capital of Hawaii is Honolulu, which is located on the island of Oahu. Hawaii is a popular tourist destination, and is known for its beautiful".

**Enter a Prompt**: Points to the "Continue the conversation" button.

**Select the foundational model**: Points to the "Model" dropdown set to "chat-bison@001".

**Adjust the parameters**: Points to the parameter settings: Temperature (0 to 1), Token limit (1 to 1024, set to 256), Top-K (1 to 40, set to 40), Top-P (0 to 1, set to 0.8), and Safety filter threshold (Block few).

**Save the model**: Points to the "SAVE" button.

**Submit the prompt**: Points to the "CONTINUE" button.

**Google Cloud**: Points to the "REPORT INAPPROPRIATE RESPONSES" link.

# You can use Gen AI Studio to fine-tune foundational models

- You supply your own training data in JSON format
- The foundational model is customized for your particular use case
  - A new model is created based on PaLM 2
  - Uses a process called “Transfer Learning”
- The training job runs on Google infrastructure
- Your model and data are stored separately in your own project



## Tune a model

Tune a model so it's better equipped for your use case, then deploy to an endpoint to get predictions or test it in prompt design. Take a tutorial on [creating a tuned model](#).

[+ CREATE TUNED MODEL](#)

# Topics

01

Gen AI Studio

02

Designing and testing prompts

03

Data governance in GenAI Studio



# Start with a simple text prompt and click the Submit button to see the results



Responses :

USER	AI
Tell me about Hawaii	Hawaii is a state of the United States located in the Pacific Ocean. It is the only state in the United States that is an archipelago, meaning that it is made up of a group of islands. Hawaii is the 50th state and the only one located outside of North America. The state is made up of eight main islands: Hawaii, Maui, Oahu, Kauai, Molokai, Lanai, Niihau, and Kahoolawe. Hawaii is also

# How about a code prompt?

What is the Terraform code to create a Google Cloud Virtual Machine

e

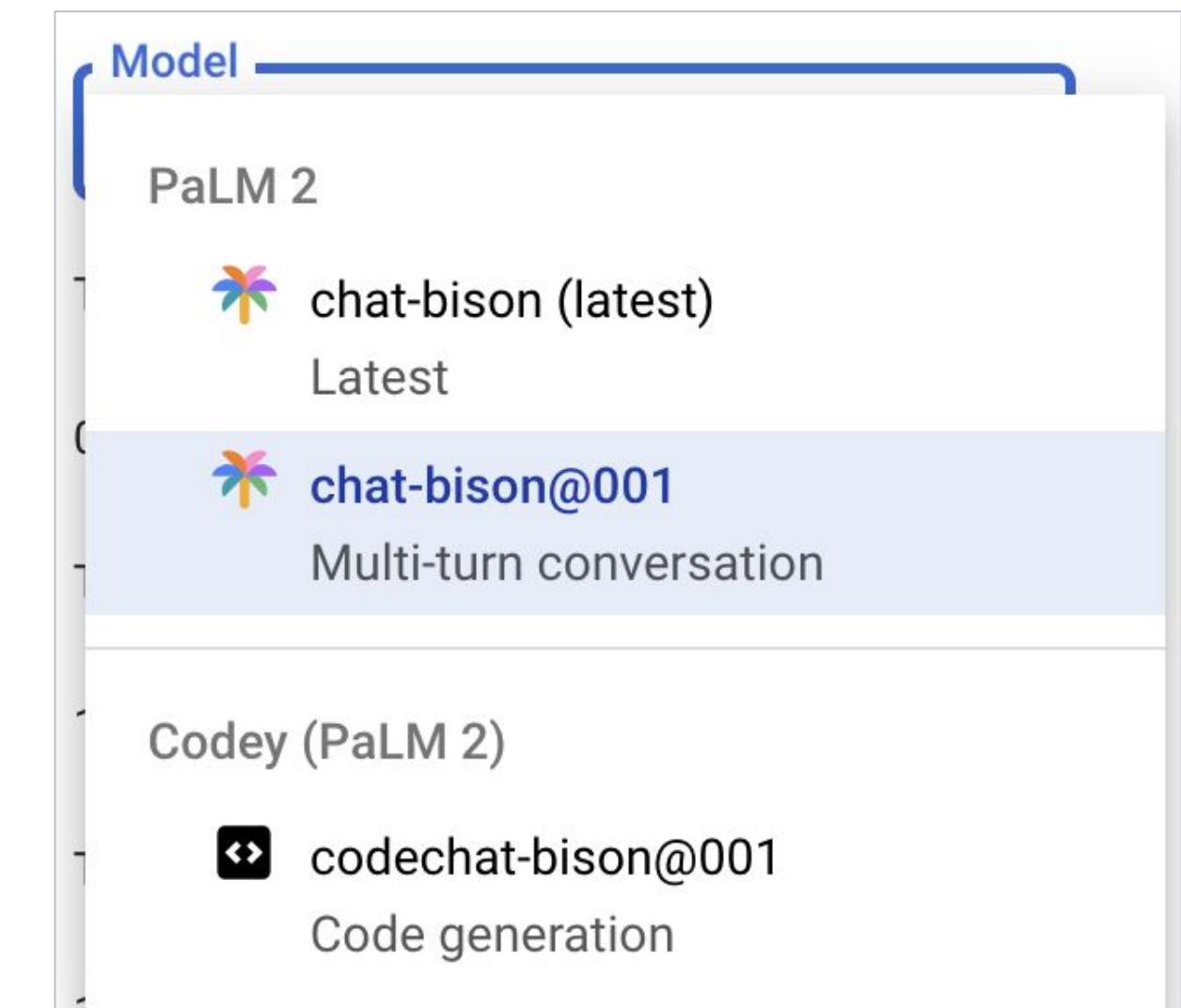
AI

Here is the Terraform code to create a Google Cloud Virtual Machine:

```
resource "google_compute_instance" "default" {  
    name = "my-vm"  
    machine_type = "n1-standard-1"  
    zone = "us-central1-a"  
  
    boot_disk {  
        initialize_params {  
            image = "debian-cloud/debian-9"  
        }  
    }  
  
    network_interface {  
        network = "default"  
        subnetwork = "default"  
    }  
}
```

# In Gen AI Studio, choose the right model for text or code

- Use different models depending on the use case
- PaLM 2 is for text
  - chat-bison@### is for multi-turn text conversation
  - text-bison@### is single request-response text generation
- Codey is for code
  - codechat-bison@### is for multi-turn code conversation
  - code-bison@### is single request-response code generation



# Select a chat to have a conversation, use Codey to talk about code

- Chats (multi-turn conversations) remember the history of what was said in the session

USER Write me the code to create a Google Cloud Storage bucket

AI To create a Google Cloud Storage bucket, you can use the following code:

```
import google.cloud.storage as gcs

# Create a client
client = gcs.Client()

# Create a bucket
bucket = client.create_bucket('my_bucket')
```

This code will create a bucket named `my_bucket` in your Google Cloud Storage account. You can then use this bucket to store data.

USER How about using a shell script

AI Sure, here is the code for creating a Google Cloud Storage bucket using a shell script:

```
#!/bin/bash

# Create the bucket
gsutil mb gs://my_bucket
```

This code will create a bucket named `my_bucket` in your Google Cloud Storage account. You can then use this bucket to store data.

# PaLM 2 is good at creating summaries

List some fun things to do in Hawaii



USER

List some fun things to do in Hawaii

AI

There are many fun things to do in Hawaii, including:

- **Visiting the beaches.** Hawaii is home to some of the most beautiful beaches in the world, including Waikiki Beach, Hanauma Bay, and Waimea Bay.
- **Snorkeling and scuba diving.** Hawaii's clear waters and abundant marine life make it a great place for snorkeling and scuba diving.
- **Hiking.** Hawaii has many hiking trails, ranging from

# How about sentiment analysis...

## Responses

:

USER

Is the following comment about Hawaii positive or negative: We had a great time in Hawaii. The food was great. Though the flight was long.

AI

The comment is positive. The person said they had a great time in Hawaii, and that the food was great. The only negative thing they mentioned was that the flight was long. However, they still had a great time overall.

# Ideation...

USER

I want to write a blog post about vacationing in Hawaii. What are some good titles?

AI

Here are some good titles for a blog post about vacationing in Hawaii:

- The Best Things to Do in Hawaii
- The Best Beaches in Hawaii
- The Best Hikes in Hawaii
- The Best Restaurants in Hawaii
- The Best Things to See in Hawaii
- The Best Places to Stay in Hawaii
- How to Plan a Trip to Hawaii

# Classification...

USER

Is the following message  
Spam or Not Spam: 2 Days  
and Nights FREE in Hawaii.  
Just listen to our 2-hour  
exclusive member sharing  
presentation.

AI

This message is spam. It is  
trying to get you to listen to  
a presentation in order to  
get a free trip to Hawaii.  
This is a common tactic  
used by scammers to get  
your personal information.

# Content creation...

USER

Can you write me a 1 page blog post on the history of Hawaii

AI

Sure, here is a 1 page blog post on the history of Hawaii:

The history of Hawaii begins with the arrival of the first Polynesians around 1,000 years ago. These early settlers were from the Marquesas Islands, and they brought with them their language, culture, and religion. Over time, the Hawaiian people developed their own unique

# Extraction...

USER

Can you extract and list the key dates from that history

AI

Sure, here are some of the key dates in the history of Hawaii:

- 1,000 years ago: The first Polynesians arrive in Hawaii.
- 1778: British explorer James Cook becomes the first European to visit Hawaii.
- 1893: A group of American businessmen and sugar planters overthrow the Hawaiian monarchy.
- 1898: The United States annexes Hawaii.
- 1959: Hawaii becomes a state of the United States.

# Ok, but how do I use this in my app?

- In Gen AI Studio, click the **View Code** button
  - Returns the code in Python, iPython, and cURL



# Python code example

[View code](#)

PYTHON

PYTHON COLAB

CURL

Use this script to request a model response in your application.

1. Set up the [Vertex AI SDK for Python](#) ↗
2. Use the following code in your application to request a model response

```
import vertexai
from vertexai.preview.language_models import ChatModel, InputOutputTextPair

vertexai.init(project="vertext-ai-dar", location="us-central1")
chat_model = ChatModel.from_pretrained("chat-bison@001")
parameters = {
    "temperature": 0.2,
    "max_output_tokens": 1024,
    "top_p": 0.8,
    "top_k": 40
}
chat = chat_model.start_chat()
response = chat.send_message("""Tell Me about Hawaii""", **parameters)
print(f"Response from Model: {response.text}")
```



# cURL code example

View code

PYTHON   PYTHON COLAB   CURL

1. Install Google Cloud SDK [Google Cloud SDK](#) if you haven't already or open Cloud Shell and skip to step three.
2. Run the following command to authenticate using your Google account.

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



3. Enter the following to request a model response

```
API_ENDPOINT="us-central1-aiplatform.googleapis.com"
PROJECT_ID="vertext-ai-dar"
MODEL_ID="chat-bison@001"

curl \
-X POST \
-H "Authorization: Bearer $(gcloud auth print-access-token)" \
-H "Content-Type: application/json" \
"https://${API_ENDPOINT}/v1/projects/${PROJECT_ID}/locations/us-central1/publishers/google/models/${MODEL_ID}:generateText"
$'{
  "instances": [
    {
      "context": "",
      "examples": [],
      "messages": [
        {
          "author": "user",
          "content": "Tell Me about Hawaii"
        }
      ]
    },
  ]
}'
```



# But I want a different language... Ask Codey!

▼ **Prompt**

Can you convert the following Python code into Go?

```
import vertexai
from vertexai.language_models import TextGenerationModel

vertexai.init(project="vertext-ai-dar", location="us-central1")
parameters = {
    "temperature": 0.2,
    "max_output_tokens": 256,
    "top_p": 0.8
```



▼ **Response**

Markdown

```
package main

import (
    "fmt"
    "google.golang.org/api/vertexai/v1alpha1"
)

func main() {
    // Initialize the client.
    client, err := vertexai.NewClient(vertexai.ClientConfig{
        ProjectID: "vertext-ai-dar",
        Location:  "us-central1",
    })
    if err != nil {
        panic(err)
    }
```

# You can explore sample use cases and prompts from Vertex AI Gen AI Studio

The screenshot displays the Vertex AI Gen AI Studio interface, specifically the 'Prompt examples' section. At the top, there are navigation tabs: 'GET STARTED' (blue), 'MY PROMPTS' (grey), 'TUNING' (grey), and 'PREVIEW' (black). Below the tabs, the title 'Prompt examples' is followed by the sub-instruction 'Try a sample prompt to test a foundation model with a practical use case'.

**Chat**

This section contains three examples:

- Summarization:** Kindergarten Science Teacher. Your name is Miss Astronomer who wants to teach children about the solar system. You can only use short sentences. **OPEN**
- Summarization:** Online Return Customer Support. You are a customer support rep and want to summarize a long customer message. **OPEN**
- Summarization:** Gluten Free Advisor. You are a nutritionist who wants to summarize a news article about gluten-free diets. **OPEN**

**Classification**

This section contains two examples:

- Freeform:** Sentiment analysis about a person. You would like to see how reporters write about certain people. You have articles and would like to see if a certain person is written about... **OPEN**
- Structured:** Customer request classification. Based on customer's answer, you want to automate routing of your customer to the proper service queue. Use few-shot... **OPEN**

**Writing**

This section contains three examples:

- Freeform:** Marketing generation Pixel. You work in Google's device marketing team and you need to create marketing pitch for the new Pixel 7 Pro. You have writer's block... **OPEN**
- Freeform:** Ad copy generation. You are a marketer and want to create different versions of the same ad to target different audiences. You would like some suggestions. **OPEN**
- Freeform:** Essay outline. Generate an outline for an essay on a particular topic. **OPEN**

# Model Garden allows you to explore both Google and open-source models for various use cases

The screenshot shows the Model Garden interface. On the left, there's a sidebar with two sections: 'Modalities' and 'Tasks'. The 'Modalities' section lists Language (23), Vision (46), Tabular (2), Documents (2), and Speech (1). The 'Tasks' section lists Generation (22), Classification (26), Detection (17), Extraction (7), Recognition (6), Translation (2), and Embedding (2). At the top right, there are buttons for 'EXPLORE GENERATIVE AI' and 'VIEW MY MODELS'. Below these are a search bar ('Search models') and a 'Suggestions' section with links to 'text embedding', 'essay outline', and 'BERT'. The main content area is titled 'Foundation models' and describes pre-trained multi-task models. It features three cards for PaLM 2 for Text, PaLM 2 for Chat, and Embeddings for text, each with a 'Foundation' and 'Language' tag, a brief description, and a 'VIEW DETAILS' button. At the bottom, there's a link to 'SHOW ALL (25)'.

Model Garden

EXPLORE GENERATIVE AI

VIEW MY MODELS

Modalities

Modality	Count
Language	23
Vision	46
Tabular	2
Documents	2
Speech	1

Tasks

Task	Count
Generation	22
Classification	26
Detection	17
Extraction	7
Recognition	6
Translation	2
Embedding	2

Search models

Suggestions [text embedding](#) [essay outline](#) [BERT](#)

## Foundation models

Pre-trained multi-task models that can be further tuned or customized for specific tasks.

Foundation Language

**PaLM 2 for Text**

Fine-tuned to follow natural language instructions and is suitable for a variety of language tasks, such as: classification, extraction, summarization and content...  
text-bison@001

[VIEW DETAILS](#)

Foundation Language

**PaLM 2 for Chat**

Fine-tuned to conduct natural conversation. Use this model to build and customize your own chatbot application.  
chat-bison@001

[VIEW DETAILS](#)

Foundation Language

**Embeddings for text**

Text embedding is an important NLP technique that converts textual data into numerical vectors that can be processed by machine learning algorithms, especially large models...  
textembedding-gecko@001

[VIEW DETAILS](#)

▼ SHOW ALL (25)

# Vertex Model Garden

Choose a model that fits your needs

Model Type	Model Description	Model Details
First Party	<p><b>Foundation models</b> Leverage Google's multimodal models across vision, dialog, code generation/completion</p> <p><b>Pre-trained APIs</b> Build and deploy AI applications faster with our pre-trained APIs powered by the best Google AI research and technology</p>	<ul style="list-style-type: none"><li>• PaLM for Text   PaLM for Chat</li><li>• Imagen for text-to-image</li><li>• Codey for code completion</li><li>• Chirp for speech-to-text</li></ul> <ul style="list-style-type: none"><li>• <a href="#">Speech-to-Text</a></li><li>• <a href="#">Natural Language Processing</a></li><li>• <a href="#">Translation</a></li><li>• <a href="#">Vision</a></li></ul>
Open Source	<p><b>Open Source</b> Access a wide variety of enterprise-ready open source models</p>	<ul style="list-style-type: none"><li>• StableDiffusion</li><li>• ViT</li><li>• EfficientNet</li></ul>
3rd Party	<p><b>3rd Party</b> Over time Model Garden will support 3rd-party models from partners with foundation models</p>	Coming soon

# Topics

01

Gen AI Studio

02

Designing and testing prompts

03

Data governance in GenAI Studio



# AI Data Governance

Google Cloud's approach to governance of customer data for Cloud AI Large Models and Generative AI



We have no access to customer data, in accordance with [GCP Terms](#) and [Cloud Data Processing Addendum](#).



We're committed to transparency, compliance with regulations like the EU General Data Protection Regulation (GDPR), and privacy best practices.



We will not use customer data to train our models



# Security Gen AI Risk Pillars

01

## Privacy & Safety

Transparency on how Google LLMs operate is core to the Google's mission for Responsible AI

02

## Prompt Hacking

Large Language Models can be manipulated to output responses that are not aligned with its objectives

03

## Sec Operations

Gen AI can be leveraged to monitor for malicious activities

04

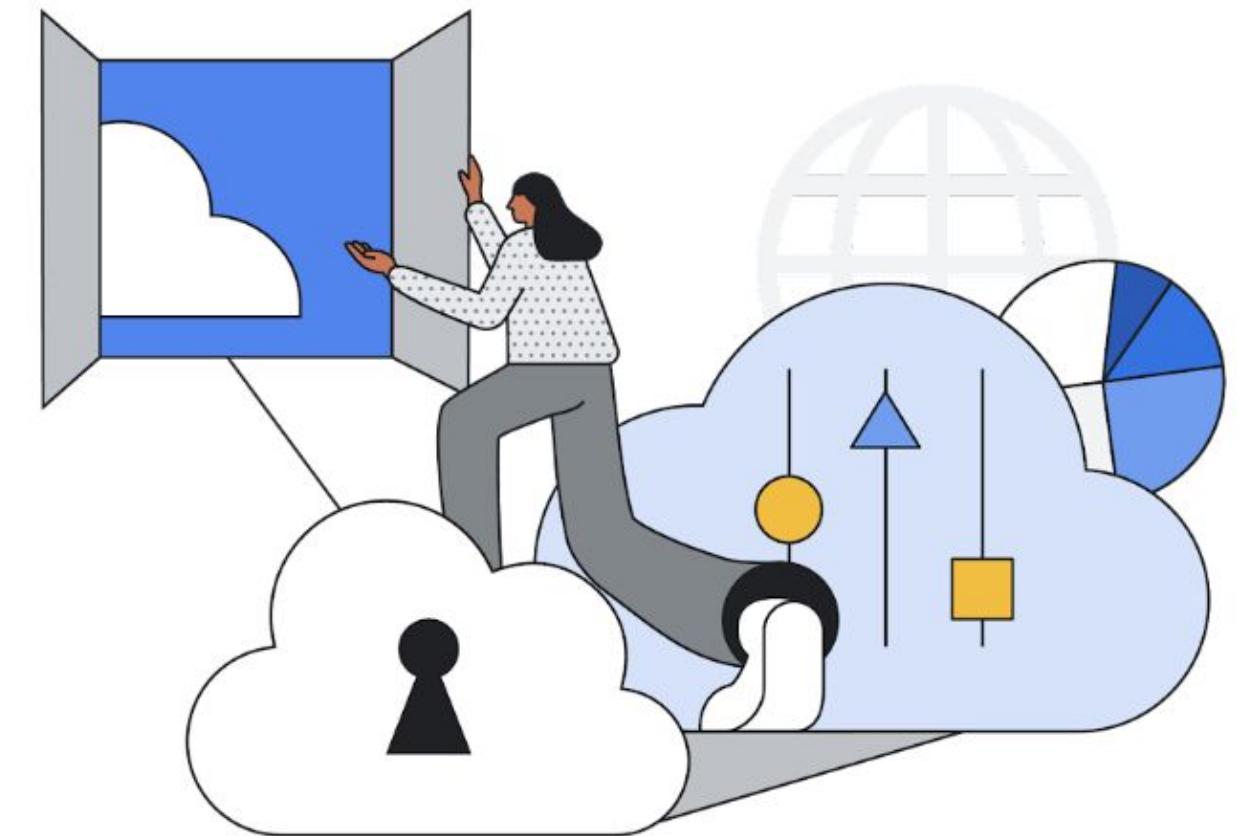
## Cloud Resources

Gen AI workloads require the same security controls as traditional workloads



# Customer data is kept private during prompt design

- Customer data is encrypted in-transit when a request is submitted to a foundational model
  - Uses industry-standard cryptographic protocols to protect the confidentiality of customer data
- The foundation model processes customer data to generate a response
  - Does not retain the data for any other purpose
- Google does not use customer data to train its models without the express consent of its customers



# Model tuning data is not retained by Google

Fine tuning does not rebuild foundational models

- Creates additional parameters, called “adapter weights”
- Passed to foundational model during inference

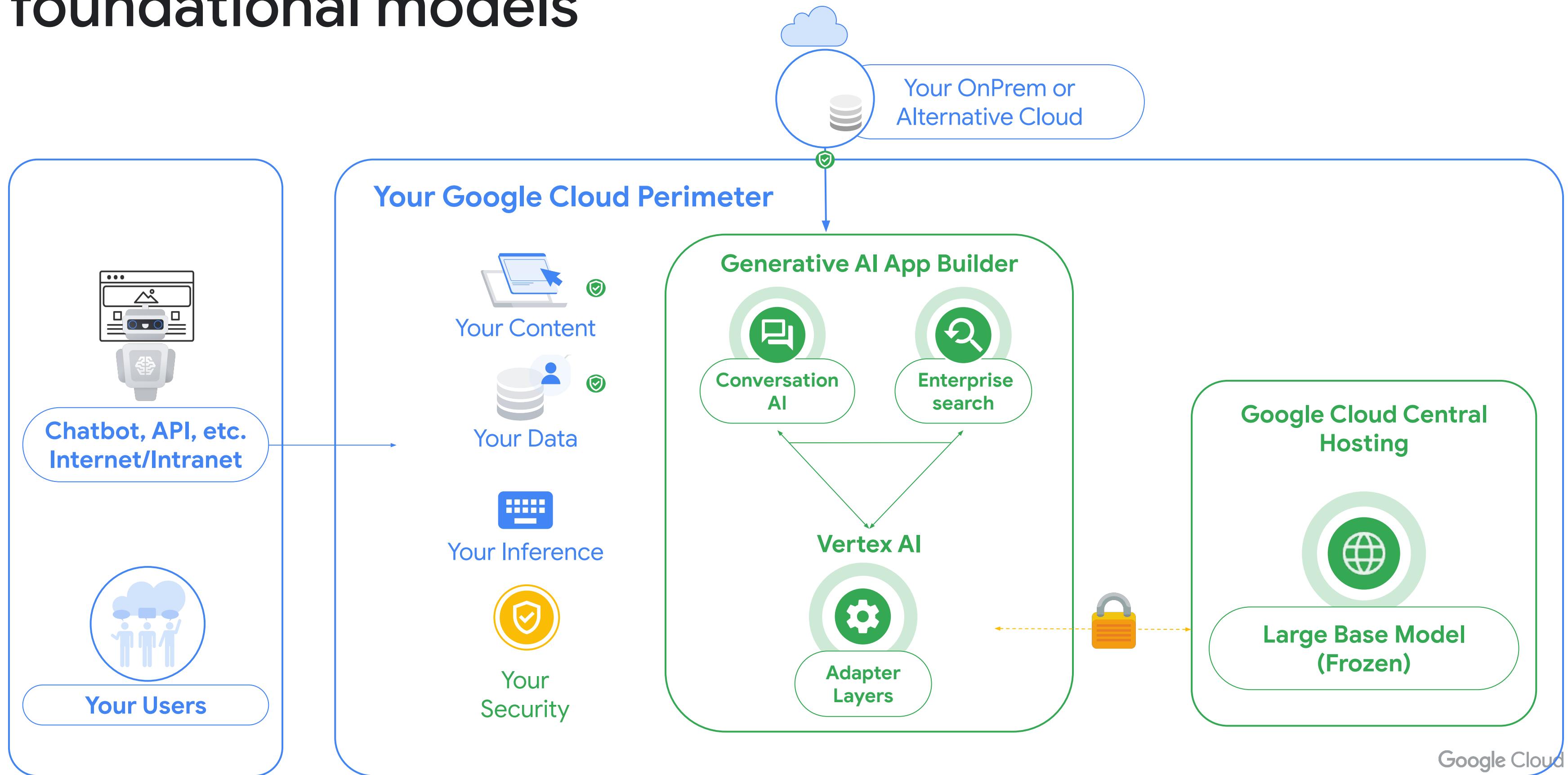
Customers have sole access to tuned models

- Can control encryption keys
- Can delete the model

Customer data is secure

- All data is encrypted at rest and in-transit
- Prompts and training data aren't logged

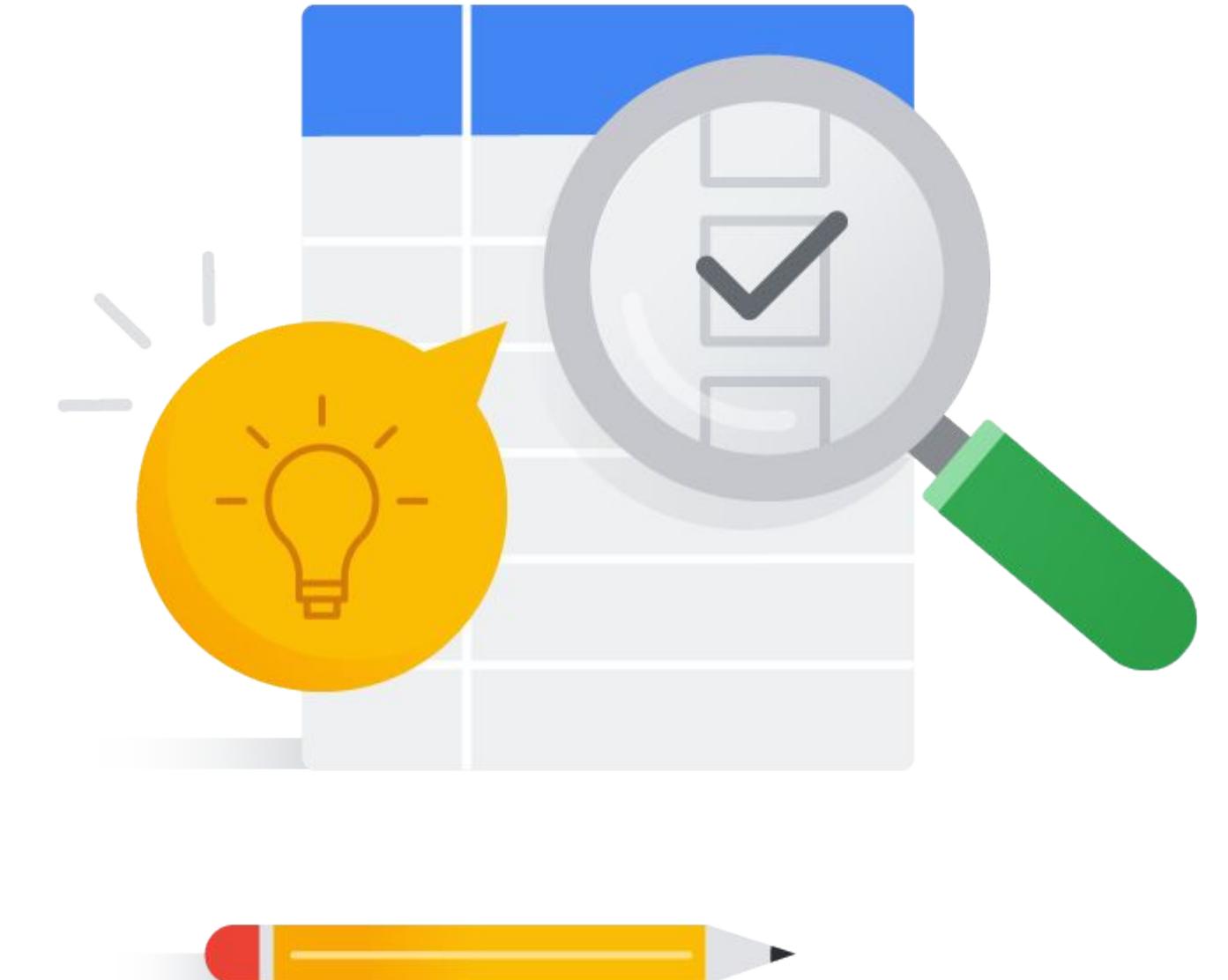
# Your fine-tuned models are separated from the foundational models



# Lab

⌚ 30 min ⚙️

Getting started with Vertex AI Gen AI Studio's User Interface

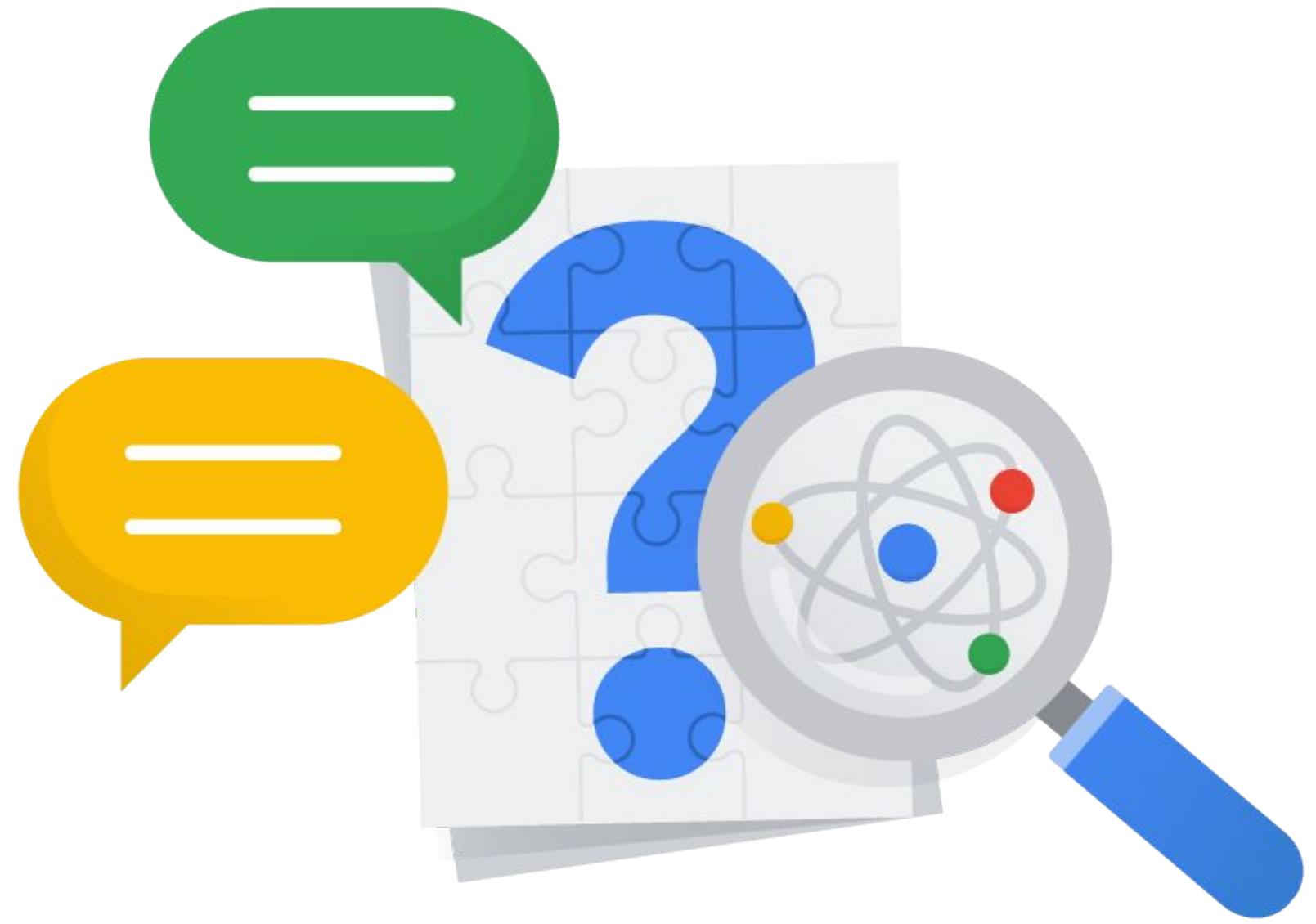


# In this module, you learned to ...

- 01 Use Gen AI Studio to implement generative AI use cases
- 02 Explore the available models and sample use cases in Gen AI Studio
- 03 Write and test prompts using the Google Cloud Console
- 04 Protect your data while using Gen AI Studio



# Questions and answers



# Quiz question

Match the model to the task

PaLM 2

Generate Speech from text

Codey

Generate code

Chirp

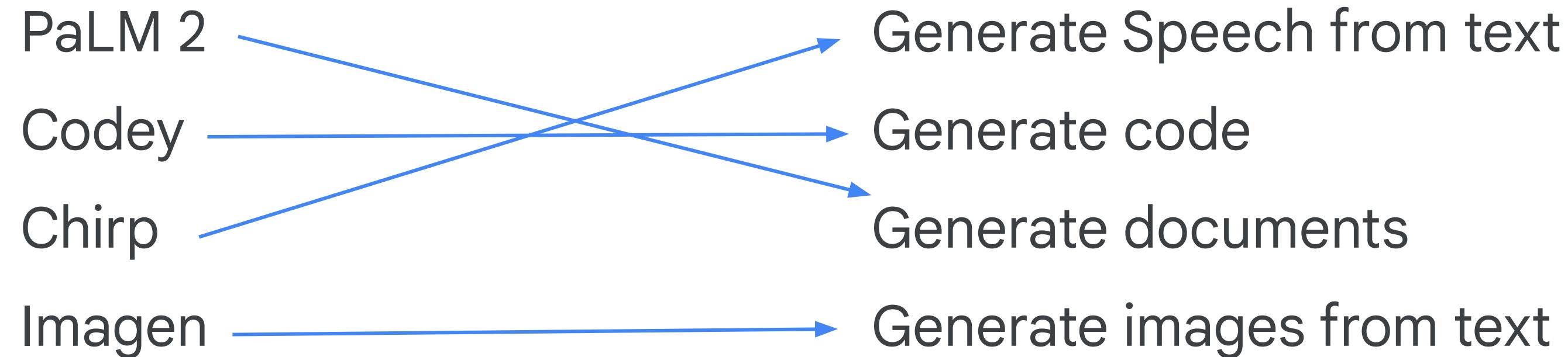
Generate documents

Imagen

Generate images from text

# Quiz question

Match the model to the task



# Quiz question

Gen AI Studio will generate code in which languages? (choose all that apply)

- A: Python
- B: IPython (Jupyter)
- C: Go
- D: C#
- E: cURL
- F: Java

# Quiz question

Gen AI Studio will generate code in which languages? (choose all that apply)

- A: Python
- B: IPython (Jupyter)
- C: Go
- D: C#
- E: cURL
- F: Java

# Quiz question

List some use cases for PaLM 2 language models:

---

---

---

---

---

---

---

# Quiz question

List some use cases for PaLM 2 language models:

Writing

Sentiment analysis

Classification

Extraction

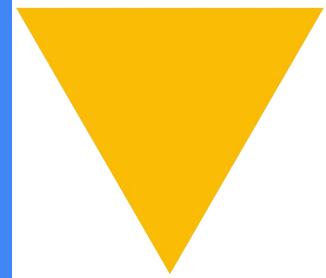
Summarization

Ideation

Chat



03



# Prompt Design

# In this module, you learn to ...

01

Design prompts for various Gen AI use cases

02

Add examples and context to customize generative AI output

03

Experiment with parameters for model fine-tuning



# Topics

01

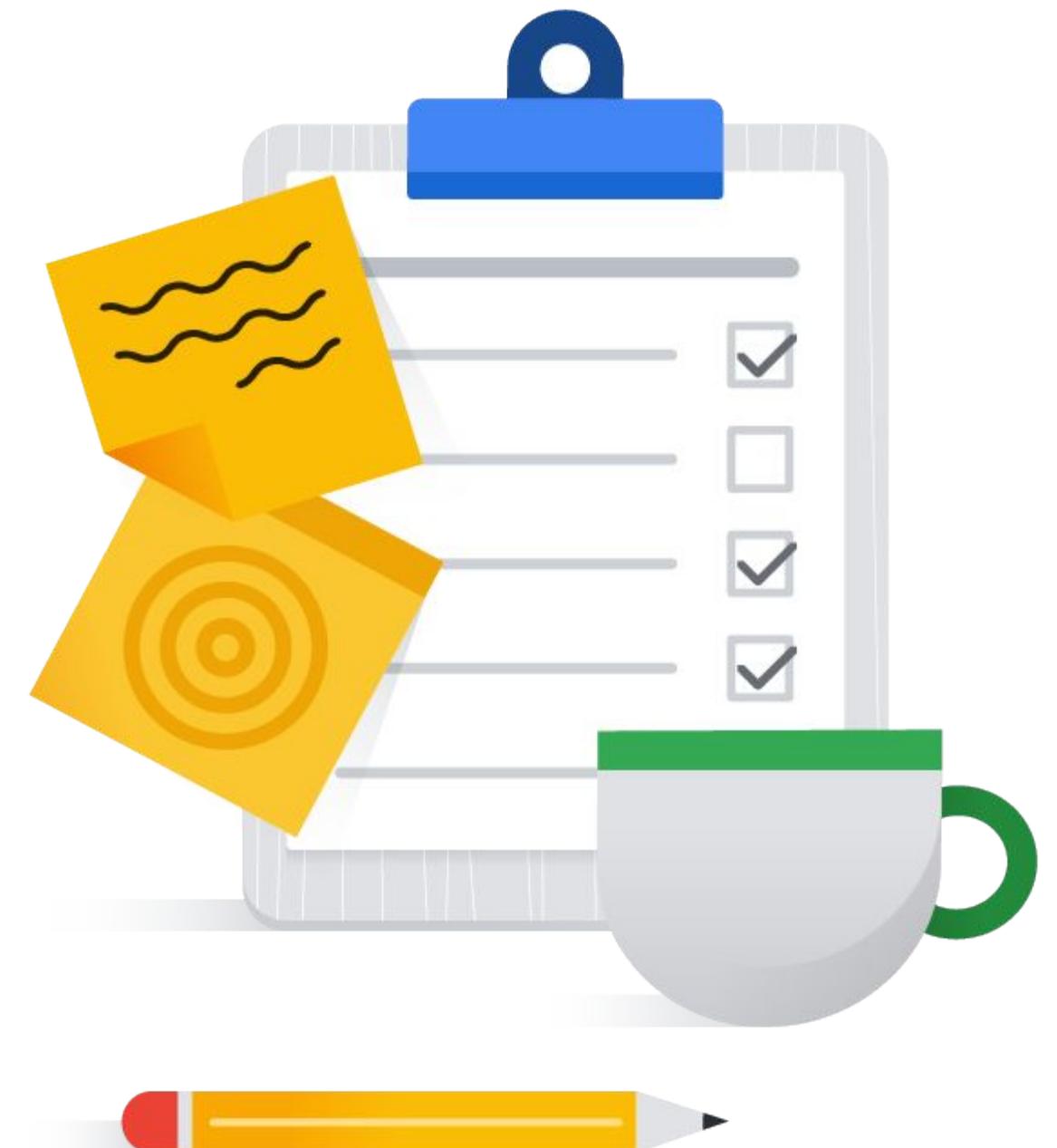
Understanding Prompt Design

02

Context, Examples, and Fine-Tuning Parameters

03

Prompt Design Best Practices



# Prompt design is the process of writing instructions to generative AI models

- Well-designed prompts can help to improve the accuracy, creativity, and safety of generative AI output
- Factors to consider when designing prompts
  - Desired output
  - Target audience
  - Limitations of the generative AI model

# Some tips for writing effective prompts include...

- Be clear and concise
- Use specific language
- Provide examples
- Use negative prompts to avoid unwanted output

With careful attention to prompt design, generative AI models can be used to create powerful and useful applications

# Those last couple of slides were written by Bard again

A screenshot of a Google Docs draft interface. At the top, there's a header bar with a small profile icon of a person with glasses, followed by the text "Write for me a slide that summarizes the importance of prompt design when using generative AI" and a pencil icon for editing. To the right of the text is a "View other drafts" button with a dropdown arrow. The main content area starts with a yellow starburst icon and the text "Sure, here is a slide that summarizes the importance of prompt design when using generative AI:". Below this, the slide structure is outlined: "Title: The Importance of Prompt Design in Generative AI", "Subtitle: How to write effective prompts that guide generative AI models to produce the desired output.", and "Body:" followed by a bulleted list of five points about prompt design. At the bottom of the slide, there's a footer bar with a "Enter a prompt here" input field, a microphone icon, and a large blue "Send" button with a white starburst icon.

Write for me a slide that summarizes the importance of prompt design when using generative AI

View other drafts ▾

Sure, here is a slide that summarizes the importance of prompt design when using generative AI:

**Title:** The Importance of Prompt Design in Generative AI

**Subtitle:** How to write effective prompts that guide generative AI models to produce the desired output.

**Body:**

- Prompt design is the process of writing instructions to generative AI models.
- Well-designed prompts can help to improve the accuracy, creativity, and safety of generative AI output.
- There are a number of factors to consider when designing prompts, including the desired output, the target audience, and the limitations of the generative AI model.
- Some tips for writing effective prompts include:

Enter a prompt here

# How prompts effect output

## Prompt

Who is Linus Torvalds?

Summarize some important dates in the life of Linus Torvalds

Write me a tweet to celebrate Linus Torvald's birthday

## Results

Linus Torvalds is a Finnish software engineer who is best known as the creator and lead developer of the Linux kernel, the operating system kernel at the core of most Linux distributions. He is also the creator of the Git distributed version control system.

1969: Linus Torvalds is born in Helsinki, Finland.  
1988: Torvalds begins studying computer science at the University of Helsinki.  
1991: Torvalds creates the Linux kernel.  
1992: The first version of Linux is released to the public.  
1996: Torvalds becomes the lead developer of Linux.  
2000: Linux is adopted by IBM as the operating system for its servers.  
2007: Torvalds is named one of Time magazine's "100 Most Influential People in the World."  
2011: Torvalds receives the ACM Turing Award for his contributions to the development of Linux.  
2015: Torvalds steps down as the lead developer of Linux, but remains an active contributor to the project.

Happy birthday to Linus Torvalds, the creator of Linux! Thank you for your contributions to the open source community. #Linux #OpenSource

# Be specific when writing your prompts

Prompt	Response
Write for me a short 2-level outline in bullets for a 1-page paper about Linus Torvalds	<p>Linus Torvalds</p> <ul style="list-style-type: none"><li>● Biography<ul style="list-style-type: none"><li>○ Born in Helsinki, Finland in 1969</li><li>○ Studied computer science at the University of Helsinki</li><li>○ Started working on Linux in 1991</li><li>○ Linus Torvalds is the creator of the Linux kernel, the core of the Linux operating system.</li></ul></li><li>● Accomplishments<ul style="list-style-type: none"><li>○ Linux is the most popular operating system for servers and supercomputers</li><li>○ Linux is also used on a wide variety of devices, including smartphones, tablets, and personal computers</li><li>○ Linus Torvalds has received numerous awards for his work on Linux, including the ACM Turing Award in 2015</li></ul></li><li>● Influence<ul style="list-style-type: none"><li>○ Linus Torvalds has had a profound influence on the open source software movement</li><li>○ Linux has inspired the development of many other open source operating systems, such as FreeBSD and Solaris</li><li>○ Linus Torvalds is a role model for many programmers and engineers</li></ul></li><li>● ...</li></ul>

# Topics

01

Understanding Prompt Design

02

Context, Examples, and Fine-Tuning Parameters

03

Prompt Design Best Practices



# Quality of the output can be improved by providing context to the prompt

- Context can be a few words or a paragraph
- You can specify information about you or your organization
- Specify a genre or style for the results

# Here are the results of a prompt with no context

The screenshot shows a conversational AI interface with two main sections: 'Context (Optional)' and 'Responses'.

**Context (Optional):** A text input field with placeholder text "Write your context here..." and a question mark icon.

**Examples (Optional):** A section containing two rows:

- USER:** Write an input to add a new row
- AI:** Write an output to add a new row

**Responses:** A list of messages:

- USER:** Tell me about New Orleans
- AI:** New Orleans is a city in the U.S. state of Louisiana. It is the most populous city in Louisiana, the second-most populous city in the Southern United States, and the 41st-most populous city in the United States. New Orleans is the parish seat of Orleans Parish, the most populous parish in Louisiana. The city is located on the Mississippi River, at the mouth of the Gulf

# Let's see how adding context changes the output

The screenshot shows a user interface for generating AI responses. On the left, there are two sections: 'Context (Optional)' and 'Examples (Optional)'. The 'Context' section contains a single paragraph: 'You work for an online travel agent selling exclusive tourist packages to popular destinations'. The 'Examples' section contains two rows: one from 'USER' with the prompt '+ Write an input to add a new row' and one from 'AI' with the prompt '+ Write an output to add a new row'. On the right, under the heading 'Responses', there are three generated paragraphs. The first paragraph continues the context about travel destinations. The second paragraph provides information about New Orleans, mentioning its French Quarter, jazz music, and Mardi Gras celebrations, as well as its museums like the National World War II Museum and the New Orleans Museum of Art. The third paragraph concludes by stating that New Orleans is a perfect destination for everyone, from history buffs to music lovers.

Context (Optional)

You work for an online travel agent selling exclusive tourist packages to popular destinations

Responses

by the United States as part of the Louisiana Purchase.

New Orleans is a major tourist destination, known for its French Quarter, jazz music, and Mardi Gras celebrations. The city is also home to a number of museums, including the National World War II Museum and the New Orleans Museum of Art.

If you're looking for a unique travel experience, New Orleans is the perfect destination. The city has something to offer everyone, from history buffs to music lovers to

# Let's try a different context

The screenshot shows a user interface for a conversational AI application. On the left, there are two sections: 'Context (Optional)' and 'Examples (Optional)'. The 'Context' section contains a descriptive paragraph about a user working for the National Weather Service. The 'Examples' section shows a user input to add a new row and an AI-generated output to add a new row. On the right, there is a 'Responses' section displaying several paragraphs of text about New Orleans, its location, architecture, culture, history, and vulnerability to hurricanes and flooding.

**Context (Optional)**

You work for the National Weather service writing blog posts about climate events in various locations around the world.

**Examples (Optional)**

USER (+) Write an input to add a new row

AI (+) Write an output to add a new row

**Responses**

River, at the mouth of the Gulf Intracoastal Waterway, 125 miles (201 km) upriver from the Gulf of Mexico. The city is known for its distinctive architecture, vibrant culture, and rich history.

New Orleans is also known for its vulnerability to hurricanes and flooding. The city has been hit by several major hurricanes in recent years, including Hurricane Katrina in 2005 and Hurricane Ida in 2021. These hurricanes caused widespread damage and displacement, and

# Let's try requesting a style of output

▼ **Context (Optional)** ?

You work for an online travel agent that sells vacation packages to popular destinations. You write ads in the style of Dr. Seuss.

Write me an add for a vacation rental in Hawaii on the breach

▶ 

AI

Oh, the places you'll go  
When you stay in Hawaii,  
On the beach, by the sea,  
With a view of the palm trees.

You can swim in the ocean,  
And surf on the waves, And build sandcastles on the beach, And watch the sunset over the water.

You can go for hikes in the mountains, And explore the rainforest, And visit the volcanoes, And see the native wildlife.

# Examples demonstrate what the output should look like

- Use a history of what you have written in the past to get the model to output content in your style
  - Zero-shot prompting means there are no examples
  - One-shot prompting means there is one example
  - Few-shot prompting means there are multiple examples

# Let's add some context and some examples and see how it affects output

The screenshot illustrates a conversational AI interface with three main sections: Examples (Optional), Context (Optional), and a main output area.

**Examples (Optional):**

- USER:** Write ad ad for a beech vacation
- AI:** Great deals on 1-week rentals in Tahiti. Going fast, visit us at [www.funtimes.co](http://www.funtimes.co)

**Context (Optional):**

You work for Fun Times an online travel agent selling vacation packages to popular destinations. Your site is: [www.funtimes.com](http://www.funtimes.com)

**Output Area:**

**USER:** Write an ad for a 1 week vacation on in Waikiki Beach.

**AI:** Looking for a tropical vacation? Look no further than Waikiki Beach. With its white sand beaches, crystal clear water, and endless activities, Waikiki Beach is the perfect place to relax and have fun. Book your vacation today at [www.funtimes.com](http://www.funtimes.com).

# Use the other model parameters to further customize the results

- Temperature
- Token limit
- Top-K
- Top-P

The screenshot shows a configuration interface for model parameters. It includes five sliders with their current values displayed in boxes:

- Temperature: 0.2
- Token limit: 256
- Top-K: 40
- Top-P: 0.8
- Safety filter threshold: Block few

Below the sliders are two buttons: SUBMIT and RESET PARAMETERS. At the bottom, there is a link to REPORT INAPPROPRIATE RESPONSES and a note about content processing.

Content processed through Vertex AI is

# Temperature controls the degree of randomness in token selection

- Range from 0 to 1
- Lower temperatures are good for prompts that require correctness
  - More likely to select the most expected next token
- Higher temperatures can lead to more diverse or unexpected results
  - More creative
- A temperature of 0 is deterministic
  - The highest probability token is always selected
  - Will always return the same result for a given prompt
- Default is a temperature of .2
  - Start there and experiment with your results

The screenshot shows the Vertex AI Text Generation interface with several configuration options:

- Temperature**: A slider set to 0.2.
- Token limit**: A slider set to 256.
- Top-K**: A slider set to 40.
- Top-P**: A slider set to 0.8.
- Safety filter threshold**: Set to "Block few".
- Buttons**: SUBMIT, RESET PARAMETERS, and a blue REPORT INAPPROPRIATE RESPONSES button.
- Text at bottom**: Content processed through Vertex AI is

# Token limit determines the maximum amount of output for each prompt

- The range is from 1 to 1024
  - The default value is 256
- A token is a basic unit of text that the LLM understands
  - When using the PaLM API, a token is approximately four characters

The screenshot shows the 'Temperature' slider at 0.2, the 'Token limit' slider at 256 (which is highlighted with a red border), the 'Top-K' slider at 40, the 'Top-P' slider at 0.8, and the 'Safety filter threshold' dropdown set to 'Block few'. At the bottom, there are 'SUBMIT' and 'RESET PARAMETERS' buttons, and a link to 'REPORT INAPPROPRIATE RESPONSES'.

Temperature ? 0.2

Token limit ? 256

Top-K ? 40

Top-P ? 0.8

Safety filter threshold  
Block few

SUBMIT RESET PARAMETERS

REPORT INAPPROPRIATE RESPONSES

Content processed through Vertex AI is

# Top-K changes how the model selects tokens for output

- Range is from 1 to 40
- A top-K of 1 means the selected token is the most probable among all tokens in the model's vocabulary
- A top-K of 3 means that the next token is selected from among the 3 most probable tokens
- The default top-K value is 40
- The higher the value the more tokens are possible when selecting the next token

The screenshot shows the 'Model parameters' section of the Vertex AI interface. It includes the following settings:

- Temperature**: A slider set to 0.2.
- Token limit**: A slider set to 256.
- Top-K**: A slider highlighted with a red border, set to 40.
- Top-P**: A slider set to 0.8.
- Safety filter threshold**: A dropdown set to 'Block few'.

At the bottom are 'SUBMIT' and 'RESET PARAMETERS' buttons, and a link to 'REPORT INAPPROPRIATE RESPONSES'.

Content processed through Vertex AI is

# Top-P also changes how the model selects tokens for output

- Range is from 0 to 1
- Tokens are selected from most probable to least until the sum of their probabilities equals the top-P value
  - For example, if tokens A, B, and C have a probability of .3, .2, and .1 and the top-P value is .5, then the model will select either A or B as the next token
- The default top-P value is .8
- The higher the value the more likely the model would be to select a token that is not the most probable one

The screenshot shows the 'Model Parameters' section of the Vertex AI interface. It includes the following settings:

- Temperature: 0.2
- Token limit: 256
- Top-K: 40
- Top-P: 0.8 (highlighted with a red border)
- Safety filter threshold: Block few

At the bottom are 'SUBMIT' and 'RESET PARAMETERS' buttons, and a link to 'REPORT INAPPROPRIATE RESPONSES'. A footer note states: 'Content processed through Vertex AI is'.

# Topics

01

Understanding Prompt Design

02

Context, Examples, and Fine-Tuning Parameters

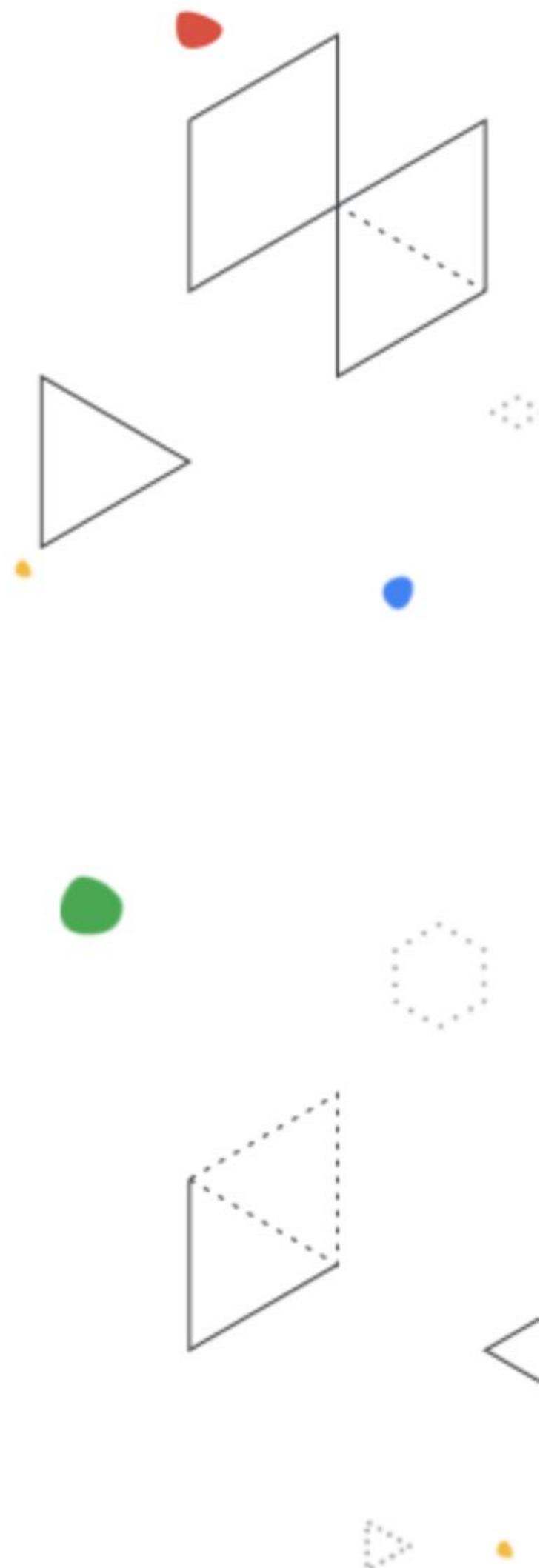
03

Prompt Design Best Practices



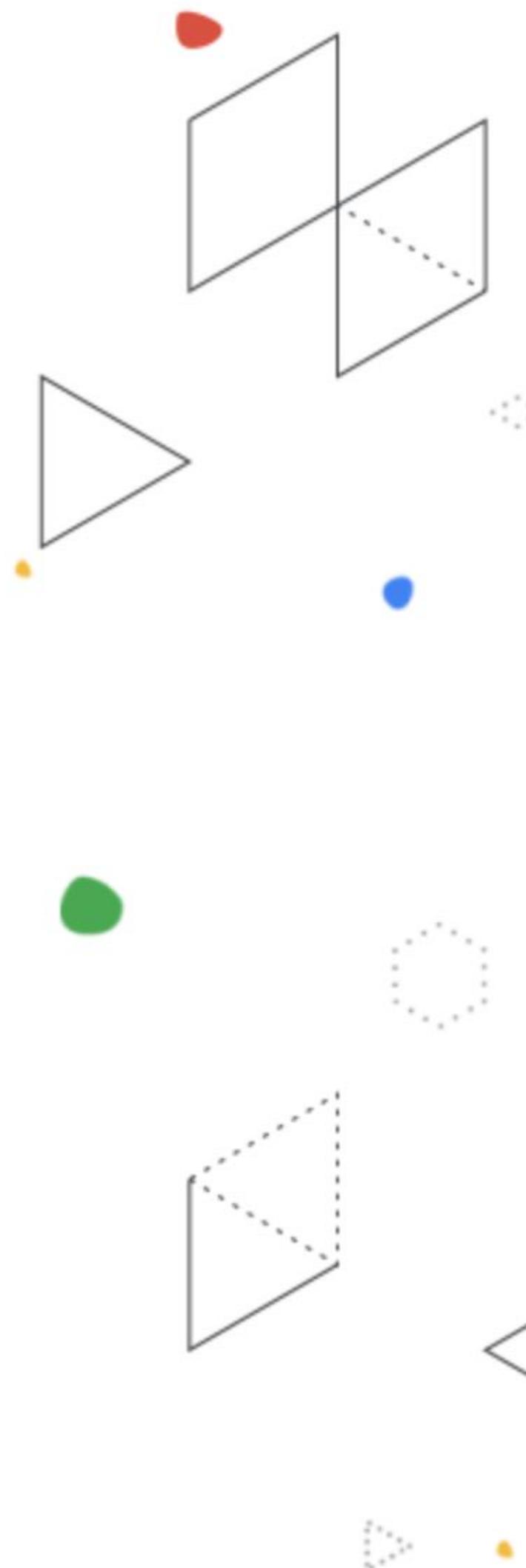
# Prompt design best practices

- Be specific
  - Don't ask: "Write me an a poem"
  - Ask: "Write me a poem about COBOL programming in the style of Edgar Allan Poe"
- Add context
  - Add information about "who" you want the model to emulate
  - What is its job, where does it work, and any other information you would want it to convey
- Add examples
  - Use examples that you have written in the past to teach it to write in your style
  - Give examples of how it should answer questions



# Be clear and concise

- Use positive language
  - Don't say: "Don't use long sentences"
  - Say: "Write short sentences as bullets"
- Use simple language
  - Avoid jargon
  - Avoid complex sentences
- Use correct grammar and spelling



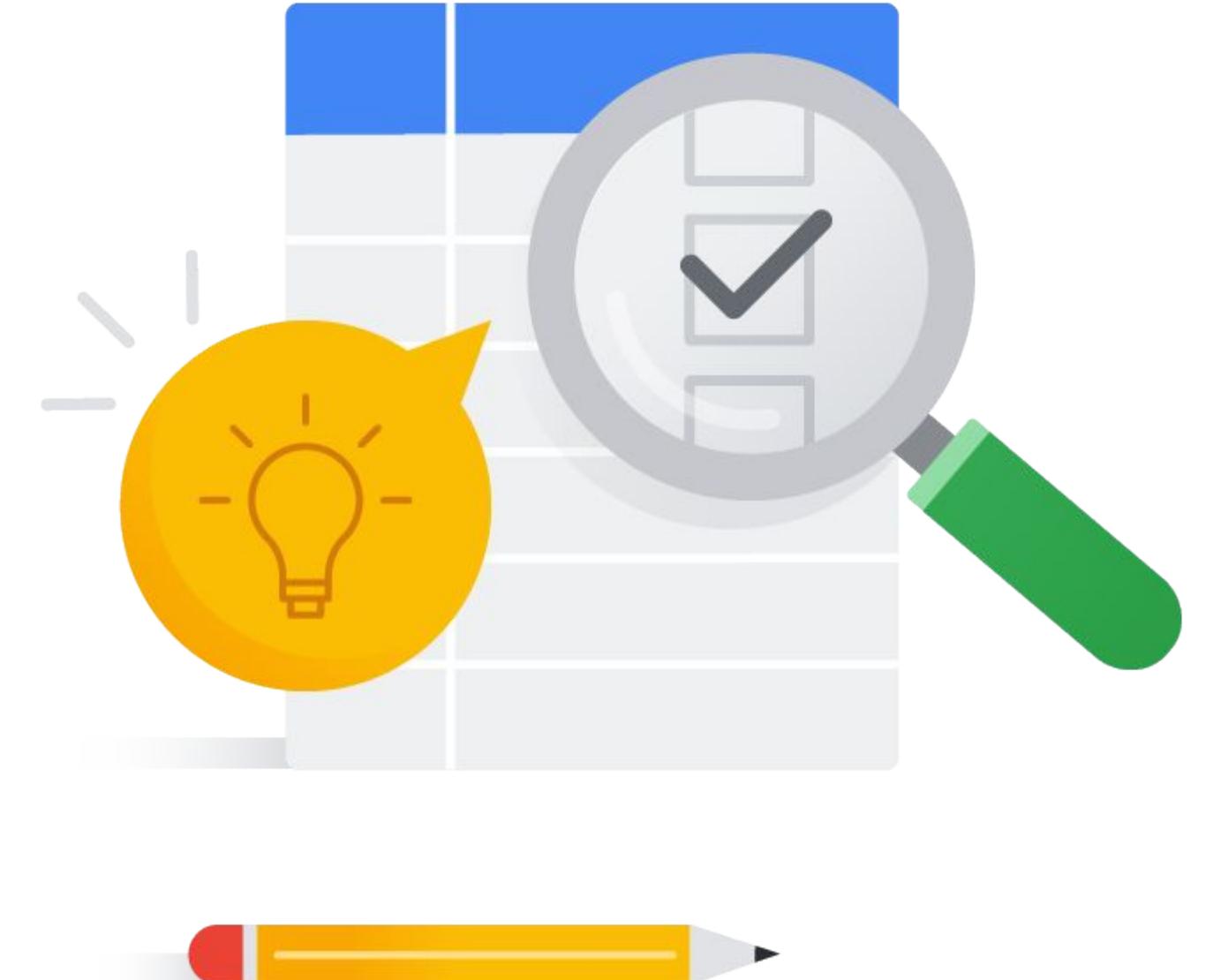
# Experiment with parameter values

- To make the model more creative, set higher values for Temperature, Top-K, and Top-P
- Set lower values when you want the model to be consistent
- A Temperature of 0, Top-K to 1 , and Top-P to 0 would cause the model to always return the same result for a given prompt based on the most likely next token

# Lab

⌚ 30 min ⚙️

Prompt Design for Text Chats in Gen AI Studio

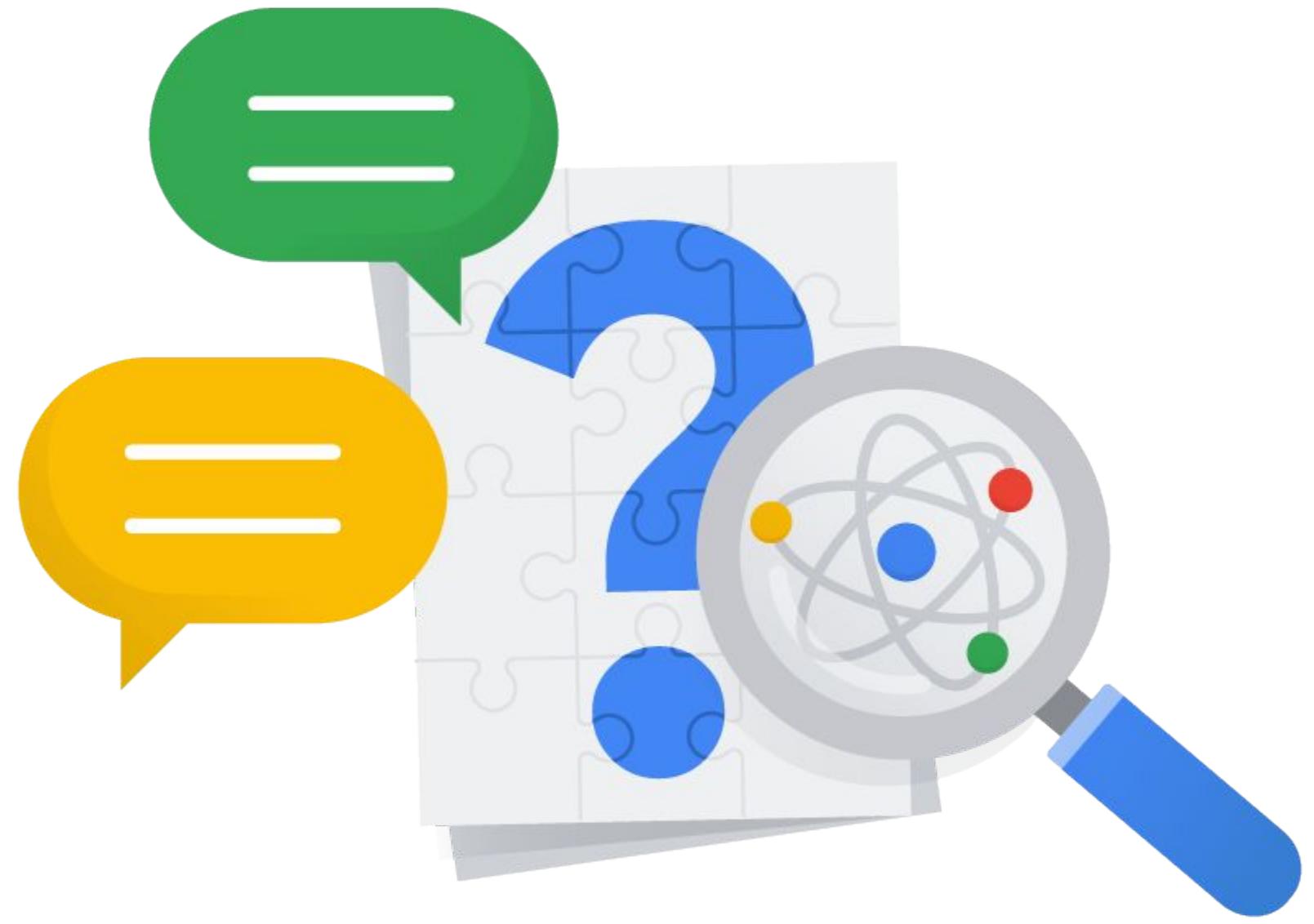


# In this module, you learned to ...

- 01 Design prompts for various Gen AI use cases
- 02 Add examples and context to customize generative AI output
- 03 Experiment with parameters for model fine-tuning



# Questions and answers



# Quiz question

- Which setting for temperature would make the model more creative when answering prompts?
- A: -1
  - B: 0
  - C: 0.1
  - D: 0.9

# Quiz question

- Which setting for temperature would make the model more creative when answering prompts?
- A: -1
  - B: 0
  - C: 0.1
  - D: 0.9

# Quiz question

If you want the model to write in your style what should you do?

- A: Set the temperature to about 0.5
- B: Raise the Top-K and Top-P values
- C: Add context
- D: Add examples

# Quiz question

If you want the model to write in your style what should you do?

- A: Set the temperature to about 0.5
- B: Raise the Top-K and Top-P values
- C: Add context
- D: Add examples

# Quiz question

If you want the model to return content more relevant to your organization or use case, what should you do?

- A: Set the temperature to about 0.5
- B: Raise the Top-K and Top-P values
- C: Add context
- D: Add examples

# Quiz question

If you want the model to return content more relevant to your organization or use case, what should you do?

- A: Set the temperature to about 0.5
- B: Raise the Top-K and Top-P values
- C: Add context
- D: Add examples





# Implementing the PaLM API

# In this module, you learn to ...

01

Use the Vertex AI API to generate content using the Pathways Language Model (PaLM)

02

Program Python applications that use PaLM to generate content

03

Integrate PaLM and GenAI into your applications



# Topics

01

Introduction to the PaLM API

02

Generative AI Powered Applications with Python

03

Using the PaLM API in Applications



# Pathways Language Model (PaLM) is a large language model

- LLMs are very sophisticated autocomplete applications
  - They learn patterns from large amounts of text
  - Use those patterns to generate text
- When generating text they calculate the next most likely tokens (words)
  - They aren't smart; it's math and statistics
- PaLM can generate text with two basic services
  - Text service for single request interactions
  - Chat service is for interactive, multi-turn interactions

# The PaLM API can be used via a REST service call

- cURL is a program that makes web requests
  - -X parameter is the HTTP verb (GET, POST, PUT, DELETE, etc.)
  - -H parameters adds header variables
  - -d parameter is the body of the request, in this case the prompt

```
API_ENDPOINT="us-central1-aiplatform.googleapis.com"
PROJECT_ID="vertext-ai-dar"
MODEL_ID="chat-bison@001"
```

```
curl -X POST -H "Authorization: Bearer $(gcloud auth print-access-token)" -H
"Content-Type: application/json"
"https://${API_ENDPOINT}/v1/projects/${PROJECT_ID}/locations/us-central1/publish
ers/google/models/${MODEL_ID}:predict" -d '${
    "instances": [
        {
            "context": "",
```

**<<code omitted>>**

# You can use the API from various programming languages including Python, Node.js, Java, and Swift

Python:

```
completion = palm.generate_text(model=model, prompt=x)
```

Node.js:

```
client.generateText({
  model: MODEL_NAME,
  prompt: {
    text: prompt,
  },
})
.then((result) => {
  console.log(JSON.stringify(result));
});
)
```

# To use the PaLM API, your application will need to be authenticated

- Ways to authenticate your application
  - Obtain an authorization token
  - Run the application using a service account

# An authorization token identifies the caller of an API

- Created using the Google Cloud CLI
  - The gcloud CLI must be initialized with either a user or service account
- Set the Authorization header variable with the token generated using gcloud

```
curl -H 'Content-Type: application/json' \
      -H "Authorization: Bearer $(gcloud auth print-access-token)" \
      -d '{"prompt": {"text": "Give me five subcategories of jazz"} }' \
      "https://generativelanguage.googleapis.com/v1beta2/models/text-bison-001:generateText"
<<code omitted>>
```

# If running an application in Google Cloud, assign a service account to the runtime

- Create a service account using **IAM**
  - Assign the **Vertex AI Service Agent** role
  - Use the service account to identify the runtime
- If using Cloud Run, App Engine, or Cloud Functions, the runtime will use the Compute Engine Default Service Account by default
  - This will work as it uses the Editor role
  - Violates principle of least privilege
- You can also download Service Account keys to authenticate programs that use the language client libraries

The screenshot shows the 'Service account details' section of the Google Cloud IAM service account creation interface. It includes fields for the service account name (set to 'vertext-ai-sa'), display name, service account ID (also set to 'vertext-ai-sa'), and email address (set to 'vertext-ai-sa@vertext-ai-dar.iam.gserviceaccount.com'). Below this, the 'Grant this service account access to project (optional)' section is shown, featuring a dropdown menu set to 'Vertex AI Service Agent' under the 'Role' heading. A note below the dropdown states: 'Gives Vertex AI the permissions it needs to function.' To the right of the role dropdown, there is an 'IAM condition (optional)' section with a '+ ADD IAM CONDITION' button and a trash bin icon.

# Topics

01

Introduction to the PaLM API

02

Generative AI Powered Applications with Python

03

Using the PaLM API in Applications



# You can use Gen AI Studio to generate Python code for a Python app or Jupyter

- Click the **View Code** button and select Python or Python Colab
  - Colab is a free Google Jupyter service online

View code

PYTHON   PYTHON COLAB

Use this script to request a model response in your application.

1. Set up the [Vertex AI SDK for Python](#)
2. Use the following code in your application to request a model response

```
import vertexai
from vertexai.preview.language_models import ChatModel, InputOutputTextPair

vertexai.init(project="vertext-ai-dar", location="us-central1")
chat_model = ChatModel.from_pretrained("chat-bison@001")
parameters = {
    "temperature": 0.2,
    "max_output_tokens": 256,
    "top_p": 0.8,
    "top_k": 40
}
chat = chat_model.start_chat()
response = chat.send_message("""List 10 fun tourist destinations""", **parameters)
print(f"Response from Model: {response.text}")
```

View code

PYTHON   PYTHON COLAB

Use this script to request a model response with a Colaboratory notebook.

1. Install the [Vertex AI SDK](#) for Python. After installing you'll be prompted to restart your Colab notebook runtime.

```
!pip install "shapely<2.0.0"
!pip install google-cloud-aiplatform >= 1.26.0
```
2. Use the following code in your notebook to request a model response

```
from google.colab import auth as google_auth
google_auth.authenticate_user()

import vertexai
from vertexai.preview.language_models import ChatModel, InputOutputTextPair

vertexai.init(project="vertext-ai-dar", location="us-central1")
chat_model = ChatModel.from_pretrained("chat-bison@001")
parameters = {
    "temperature": 0.2,
    "max_output_tokens": 256,
    "top_p": 0.8,
    "top_k": 40
}
chat = chat_model.start_chat()
response = chat.send_message("""List 10 fun tourist destinations""", **parameters)
print(f"Response from Model: {response.text}")
```

# Vertex AI requirements for Python

- Use pip to install Google Cloud AI Platform
  - Or add to your requirements.txt file

```
pip install google-cloud-aiplatform >= 1.25.0
```

- Import Vertex AI

```
import vertexai
from vertexai.preview.language_models import ChatModel, InputOutputTextPair
```

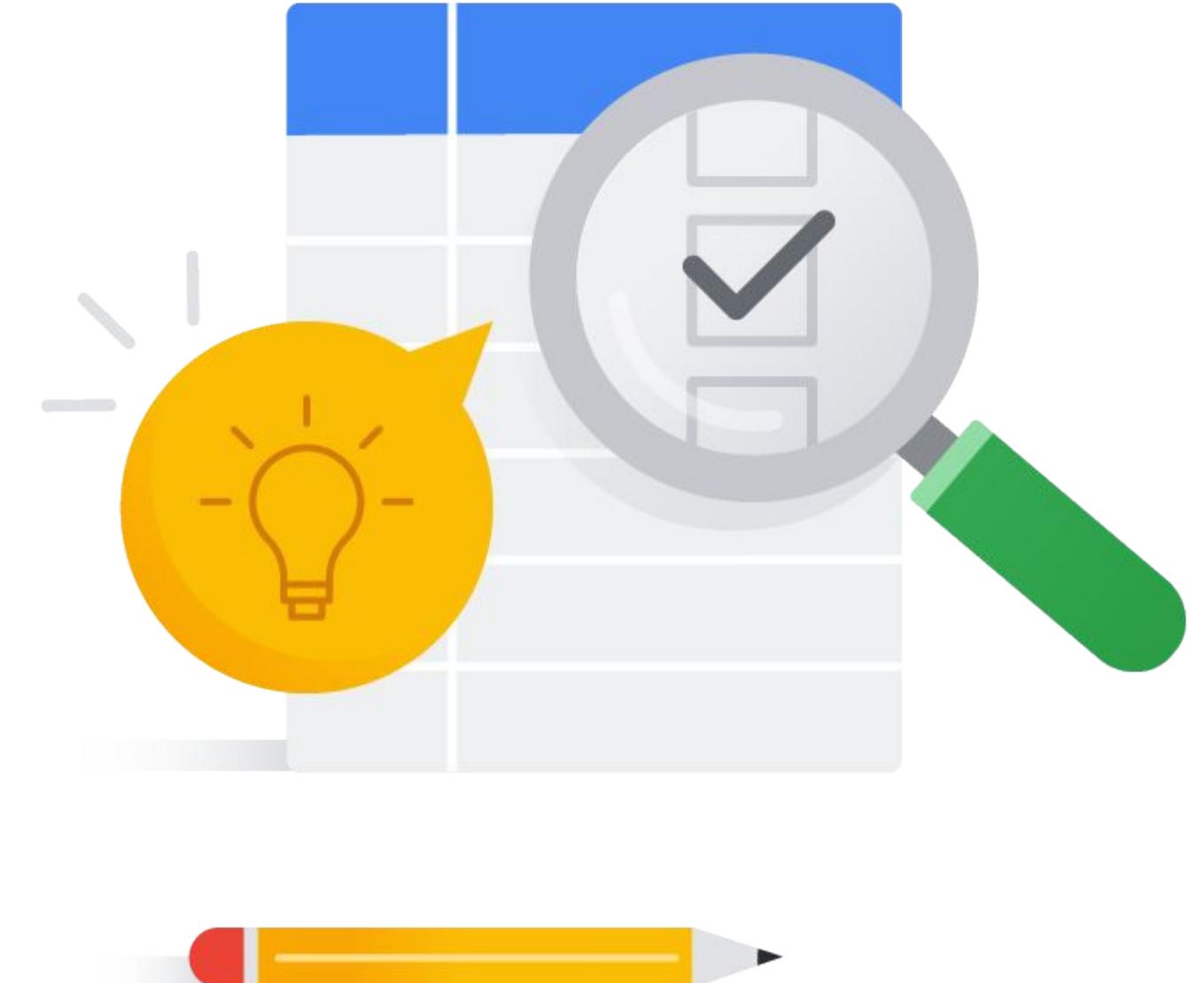
# Using Vertex AI from Python

```
vertexai.init(project="vertext-ai-dar", location="us-central1")
chat_model = ChatModel.from_pretrained("chat-bison@001")
parameters = {
    "temperature": 0.2,
    "max_output_tokens": 256,
    "top_p": 0.8,
    "top_k": 40
}
chat = chat_model.start_chat()
response = chat.send_message("""List 10 fun tourist destinations""",
**parameters)
print(f"Response from Model: {response.text}")
```

# Lab

⌚ 30 min ⚙️

Getting Started with the PaLM API for Chatbots



# Topics

01

Introduction to the PaLM API

02

Generative AI Powered Applications with Python

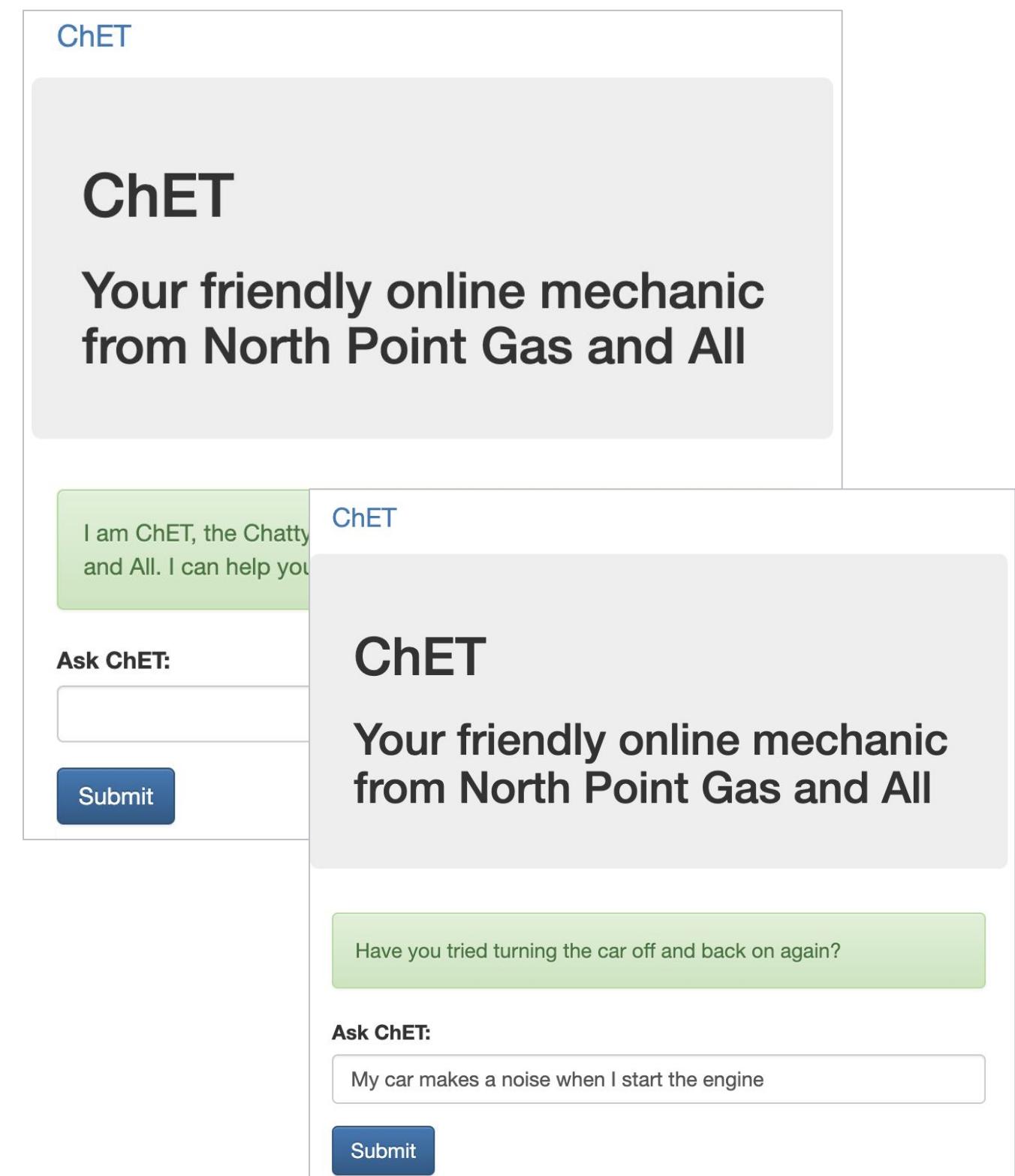
03

Using the PaLM API in Applications



# Python Flask Website example

- This is an example of using the chat service with the PaLM API
  - The chat will remember the history of the conversation
- Context must be added to tell the PaLM API to emulate a customer service agent for the service station
- The coding is simple as you are just submitting an HTML form and making a request to the PaLM API for a response
  - The response is displayed on the screen



# Add the Python requirements

- Add Google Cloud AI Platform to the requirements.txt file
- Add the required imports at the top of the code file

```
requirements.txt ×  
mixabot > requirements.txt  
1 Flask  
2 Jinja2  
3 pytest  
4 google-cloud-aiplatform
```

```
from flask import Flask, render_template, request  
import os  
import vertexai  
from vertexai.preview.language_models import ChatModel, InputOutputTextPair
```

# Handling web requests in Flask

- The default route will handle HTTP posts and gets
  - Post means a question was submitted from the HTML form
  - Get means there is no question (have MixaBot introduce itself)
- The code for using the PaLM API is in the `get_response()` function

```
@app.route("/", methods = ['POST', 'GET'])
def main():
    if request.method == 'POST':
        input = request.form['input']
        response = get_response(input)
    else:
        input = ""
        response = get_response("Who are you and what can you do?")

    model = {"title": "MixaBot", "message": response, "input": input}
    return render_template('index.html', model=model)
```

# Initializing the Chat session

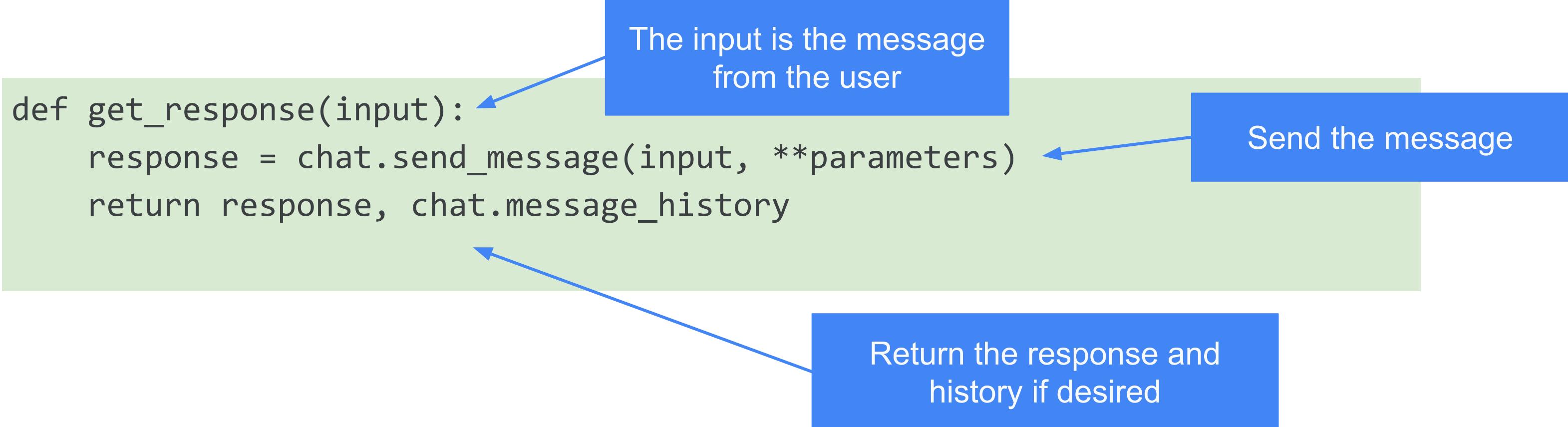
```
vertexai.init(location="us-central1")
chat_model = ChatModel.from_pretrained("chat-bison@001")
parameters = {
    "temperature": TEMPERATURE,
    "max_output_tokens": MAX_OUTPUT_TOKENS,
    "top_p": TOP_P,
    "top_k": TOP_K
}
examples=[  
    InputOutputTextPair(  
        input_text="""When I turn my car on, there is a clicking noise. """,  
        output_text="""Did you try turning the engine off and back on again?"""
    )]
chat = chat_model.start_chat(context=CONTEXT, examples=examples)
```

Initialize the API and set up the parameters

Add examples

Start the chat

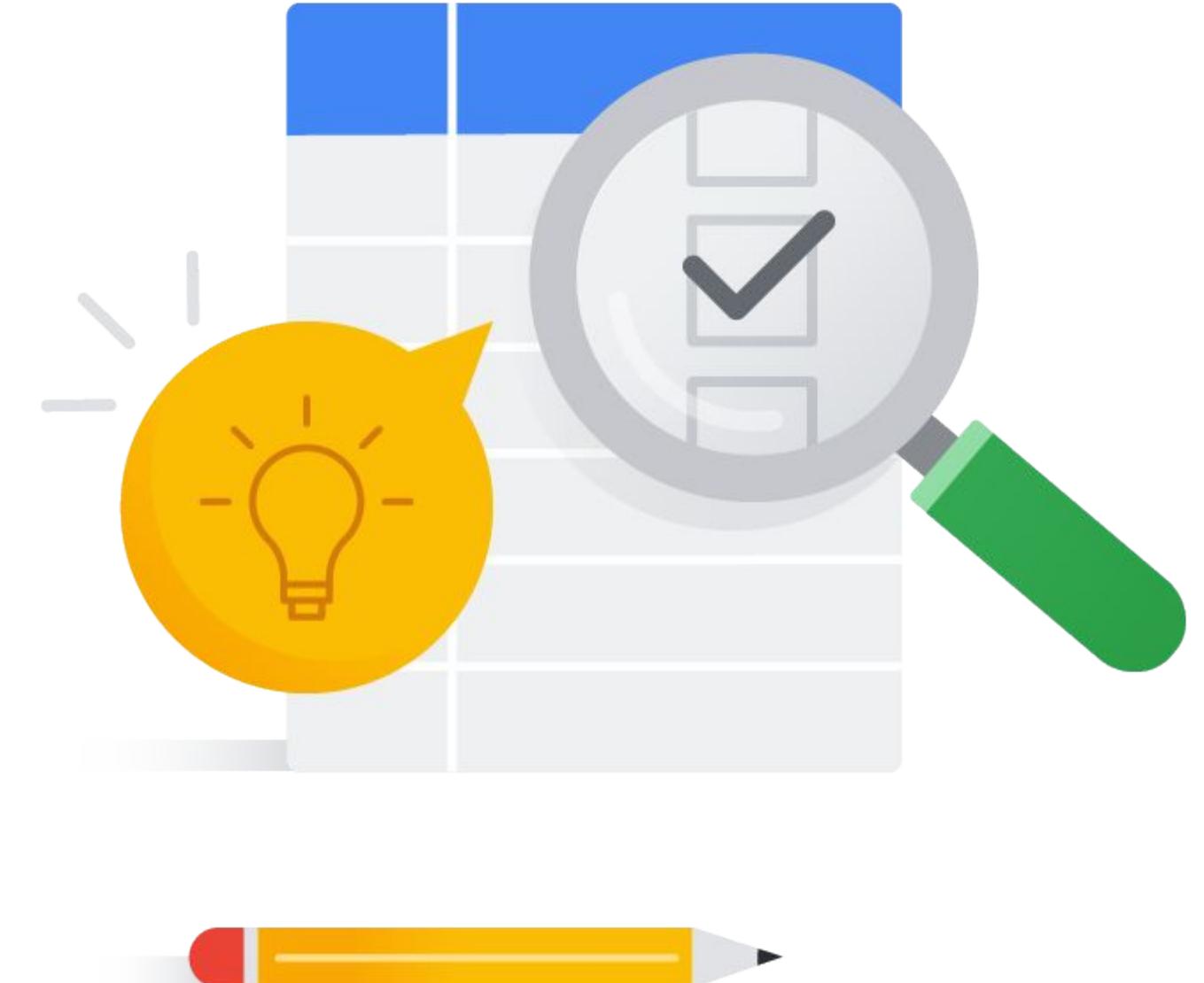
# Making a request to the PaLM API (continued)



# Lab

⌚ 30 min ⚙️

Integrating the PaLM API into Chat Applications



# In this module, you learned to ...

01

Use the Vertex AI API to generate content using the Pathways Language Model (PaLM)

02

Program Python applications that use PaLM to generate content

03

Integrate PaLM and GenAI into your applications



# Questions and answers



# Quiz question

- How do you authenticate a request to the PaLM API ?
- A: Using a API Key
  - B: Using an authorization token
  - C: With a Service Account
  - D: All of the above would work

# Quiz question

- How do you authenticate a request to the PaLM API ?
- A: Using a API Key
  - B: Using an authorization token
  - C: With a Service Account
  - D: All of the above would work

# Quiz question

What programming languages are  
Supported by the PaLM API ?

- A: Python
- B: Node.js
- C: Swift
- D: Java
- E: All of the above

# Quiz question

What programming languages are  
Supported by the PaLM API ?

- A: Python
- B: Node.js
- C: Swift
- D: Java
- E: All of the above





# Best Practices for Interactive Chat Applications

# In this module, you learn to ...

01

Adjust model settings and parameters for different use cases

02

Incorporate responsible AI into your ML and gen AI applications



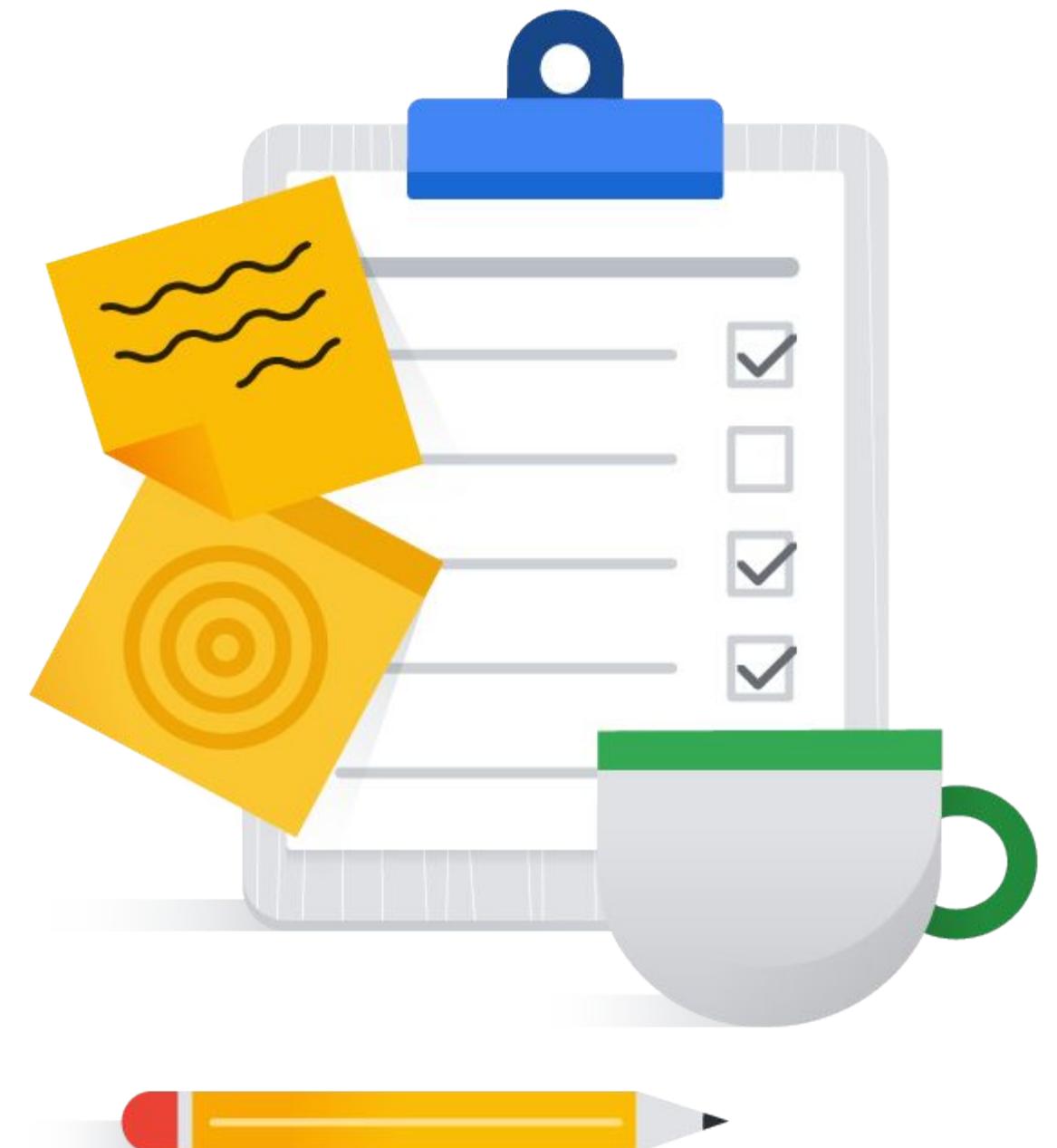
# Topics

01

Chat AI Best Practices

02

Responsible AI



# Context is important

- Context helps the model generate output specific to your organization or use case
- Add information about your company that you want the model to return
  - Website
  - Phone
  - Address
  - Etc.
- Specify the style of output that you desire

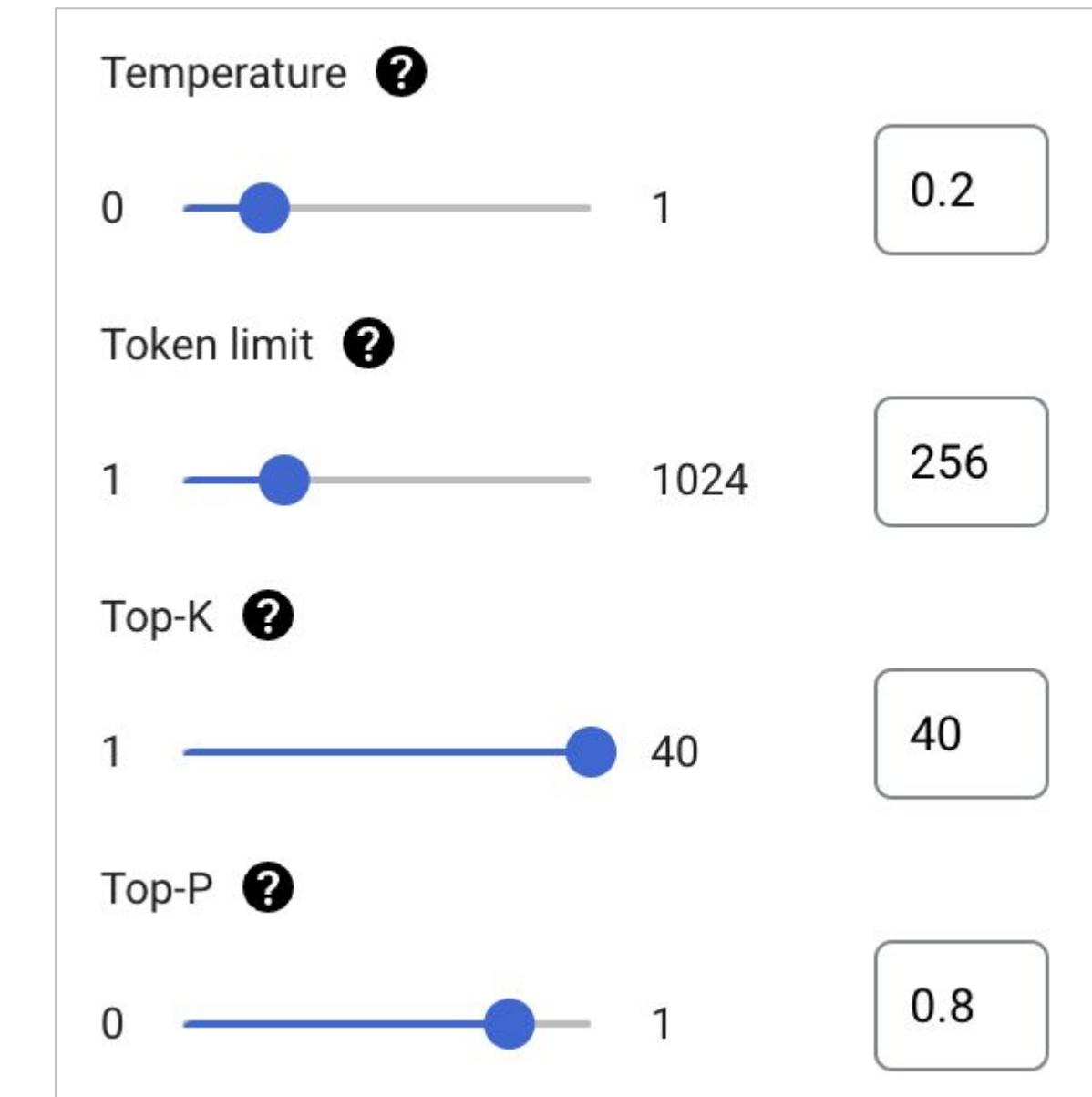


# Add examples

- Examples help the model return output that fits your parameters
- You don't want the model to just make things up
  - For example, you don't want the ChatBot to tell the user they have an appointment on Tuesday when it is not connected to any sort of database or calendar
- Anticipate typical questions, and provide acceptable responses

# Adjust parameters according to your use case

- Sometimes you want a creative chat bot
  - Producing marketing content
  - Generating social media posts
- Sometimes you want the bot to be restricted to a small set of acceptable answers
  - Customer service chat
  - Health advice
- For more creativity, set Temperature, Top-K, and Top-P higher and visa-versa



# Thoroughly test your applications

- Generative AI will often makes things up
- If interfacing with customers, you want to be sure they are given correct answers
- If using Gen AI for content generation, don't assume it is correct
  - Edit results for accuracy



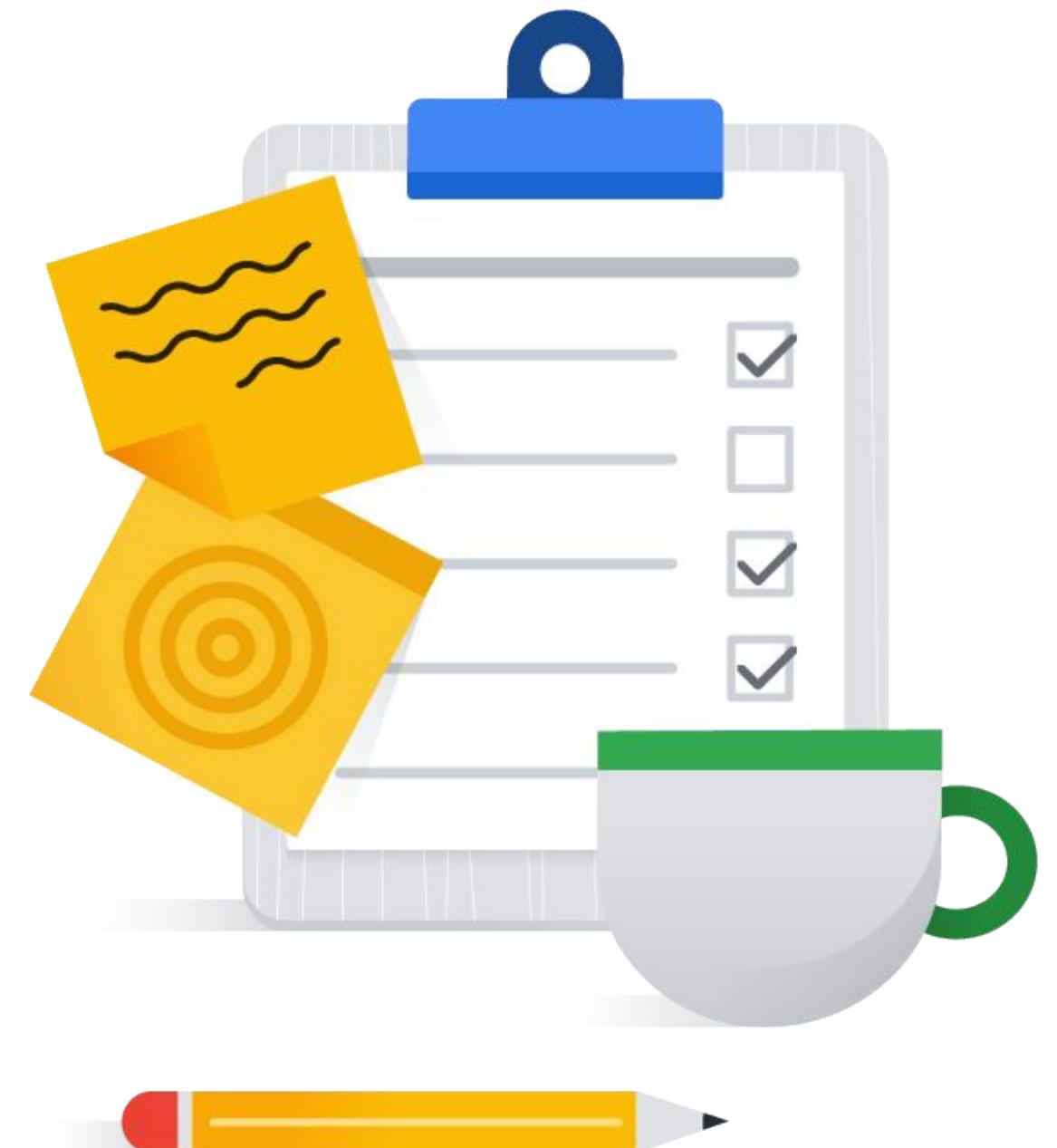
# Topics

01

Chat AI Best Practices

02

Responsible AI



# Google's seven AI principals

- 1 Be socially beneficial
- 2 Avoid creating or reinforcing unfair bias
- 3 Be built and tested for safety
- 4 Be accountable to people
- 5 Incorporate privacy design principles
- 6 Uphold high standards of scientific excellence
- 7 Be made available for uses that accord with these principles

# Google practices that ensure responsible AI

Use a human-centered design approach

Identify multiple metrics to assess training and monitoring

When possible, directly examine your raw data

Understand the limitations of your dataset and model

Test, Test, Test

Continue to monitor and update the system after deployment

# Content processing in the PaLM API is assessed against a list of safety attributes

- Scores are values from 0.0 to 1.0
  - Rounded to one decimal
- The scores are ML predictions
  - Thus, cannot be relied on for 100% accuracy
- If a response exceeds the safety threshold it is blocked
- If content is blocked, the model will return a canned response
  - e.g. "I'm not able to help with that, as I'm only a language model"

```
"predictions": [
  {
    "safetyAttributes": {
      "categories": [
        "Derogatory",
        "Toxic",
        "Violent",
        "Sexual",
        "Insult",
        "Obscene",
        "Death, Harm & Trage",
        "Firearms & Weapons",
        "Public Safety",
        "Health",
        "Religion & Belief",
        "Drugs",
        "War & Conflict",
        "Politics",
        "Finance",
        "Legal"
      ],
      "scores": [
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1,
        0.1
      ]
    }
  }
]
```

To adjust the likelihood of content being blocked, set the Safety filter threshold attribute in Gen AI Studio



# To use Gen AI responsibly follow these best practices

- Assess your application's security risks
- Consider adjustments to mitigate safety risks
- Perform safety testing appropriate to your use case
- Solicit user feedback and monitor content

# In this module, you learned to ...

01

Adjust model settings and parameters for different use cases

02

Incorporate responsible AI into your ML and gen AI applications







# Summary

# In this course, you learned ...

- 01 The Vertex AI generative AI options for your applications
- 02 To use Gen AI Studio to interact with foundation models
- 03 How to design and tune chat prompts for your Generative AI use cases
- 04 The effect of changing PaLM API parameters on model responses
- 05 How to implement the PaLM API into your applications using the Python SDK



# Take a structured approach to GenAI transformation



## Ramp your skills

Continue your learning journey and complete our [Generative AI Learning Path](#), free of charge, on Google Cloud Skills Boost



## Organizational readiness

Assess your organization's current status and business needs for generative & traditional AI capabilities



## Identify use cases

Select from one of our [Jumpstart GenAI offers](#), and work with Google Cloud to develop a technical design doc and sample code to solve the use case



## Test and scale

Purchase and implement generative AI solutions. Not all AI is built equal. POC often and fail fast to identify what works for your business.

[Contact your Google Cloud Representative to learn more](#)

# 4 GenAI Jumpstart offers - \$25k & 2 weeks per use case



## CREATE

Bring your thoughts and visions to life



## SUMMARIZE

Condense and summarize your knowledge base into a simple format



## DISCOVER

Help your customers and employees find what they need at the right time



## AUTOMATE

Automate your customer service across multiple channels

### Use cases:

- Images from text
- Product descriptions from images
- Blog post from content\*
- Email from content\*
- Release notes from content\*
- Report from content\*
- Press releases from content\*
- Personalized ads\*

### Use cases:

- Content/video summarization
- Intra-knowledge Q&A
- Explanations of code content\*
- External chatbot using internal data\*
- External chatbot using website data\*

### Use cases:

- Search for a document
- Machine-generated event monitoring
- File organization based on content\*
- Exam questions from content\*

### Use cases:

- Contract information extraction
- Feedback classification and ticket creation
- Sentiment analysis\*
- Content translation\*
- Structured data extraction from file\*
- Media tagging\*
- Product tagging\*
- Content moderation \*

\*may require Responsible AI Review

