

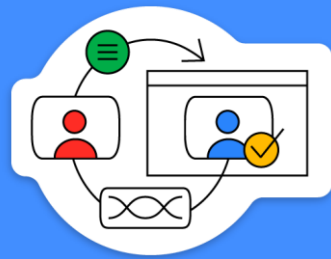
# Intro à Google Gemini API



Victor Pugliese (ele/dele)

Doutorando UNIFESP

<https://linktr.ee/victorpugliese>



# Quem sou?



Apaixonado por pesquisa e programação

- Doutorando em IA pela UNIFESP e Mestre em IA pelo ITA
- +10 anos na indústria
- +15 publicações acadêmicas, em áreas como finanças, saúde, meio ambiente e jogos
- Sou organizador do GDG Caraguatatuba
- Fã de Castelo Rá Tim Bum

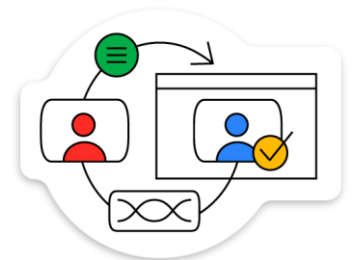
# Get started!

**Visite:** <https://github.com/google-gemini/cookbook>

## Iniciando o desenvolvimento

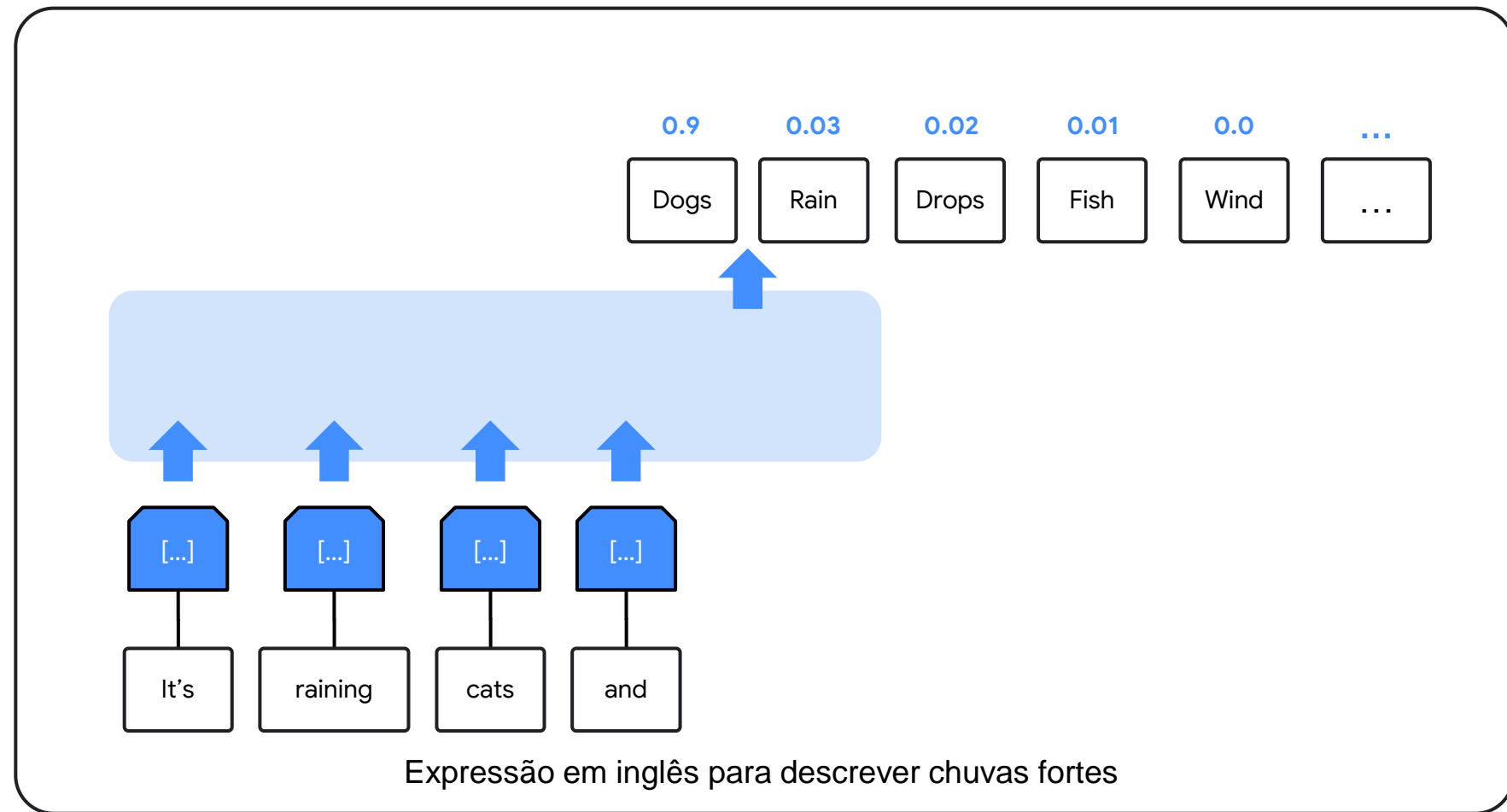
1. Acesse [Google AI Studio](#).
2. Entre com sua conta Google.
3. [Crie](#) uma chave API.
4. Utilize esse [quickstart](#) for Python.

**Leia mais sobre Google Gemini API**



O que é IA **generative**?

# O que é uma LLM?



K1ift K1oft Still,

Klift Kloft Still,  
a porta se abriu!



```
for(var i = 0;
```



```
for(var i = 0; i < 10; i++) {
```

LLMs são *enormes*.

# Problemas clássicos de linguagem natural



Extração de entidades



Classificação



Resumo



Análise de  
Sentimentos



Tradução



...

LLMs nos permitem  
prototipar *rápido*.

# Usuário:

Me dê cinco ideias para projetos em uma feira de tecnologia.

# Modelo:

## 1. **\*\* Doutor de Plantas com IA.\*\***

Construa um sistema que usa uma câmera e aprendizado de máquina para diagnosticar deficiências em plantas.

## 2. **\*\* Casa Inteligente para o Meio Ambiente.\*\***

Projete um sistema doméstico inteligente miniaturizado que monitora e otimiza o uso de energia e água. Use sensores para rastrear os níveis de temperatura, luz e umidade e construa um programa que ajusta automaticamente luzes, termostatos ou sprinklers.

## 3. **\*\*Aventura em Viagem Virtual em VR.\*\***

Crie uma experiência de realidade virtual que leva os espectadores a uma viagem educacional. Pode ser uma viagem a um marco histórico, às profundezas do oceano ou até mesmo ao corpo humano.

## 4. **\*\* O Gamer Reciclado.\*\***

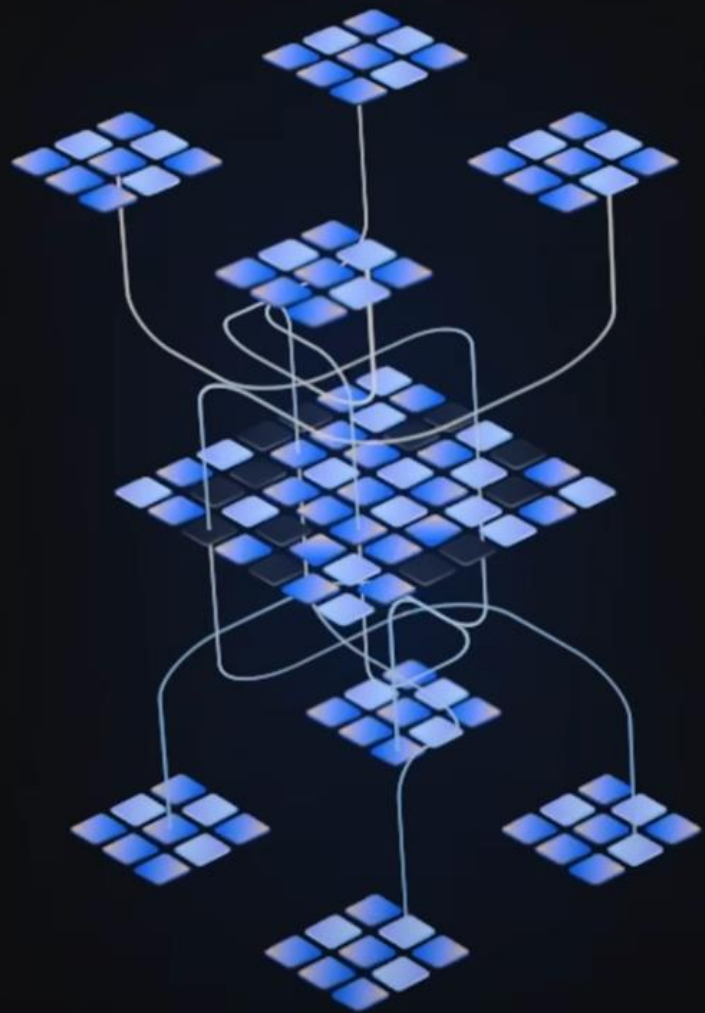
Projete um controle de videogame ou outro periférico de jogo feito de materiais reciclados.

## 5. **\*\*Identificador de fake news.\*\***

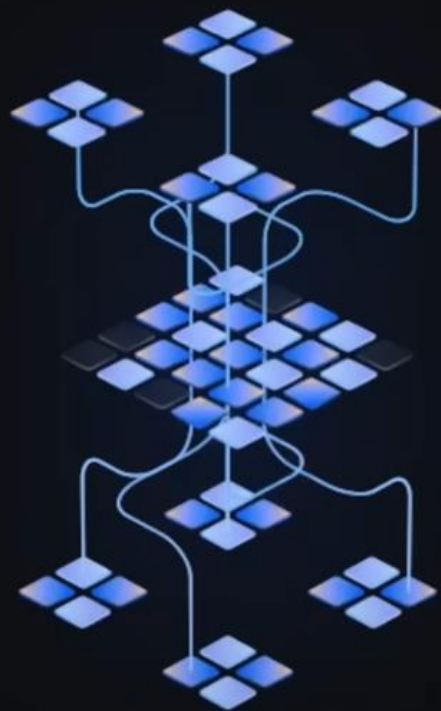
Desenvolva um programa ou site que ajuda os usuários a identificar desinformação online. Você pode treinar o programa para analisar o texto em busca de padrões de linguagem suspeitos, confiabilidade da fonte e precisão factual.

# Google é o pioneiro da indústria em IA

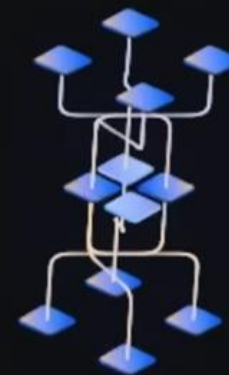




Ultra



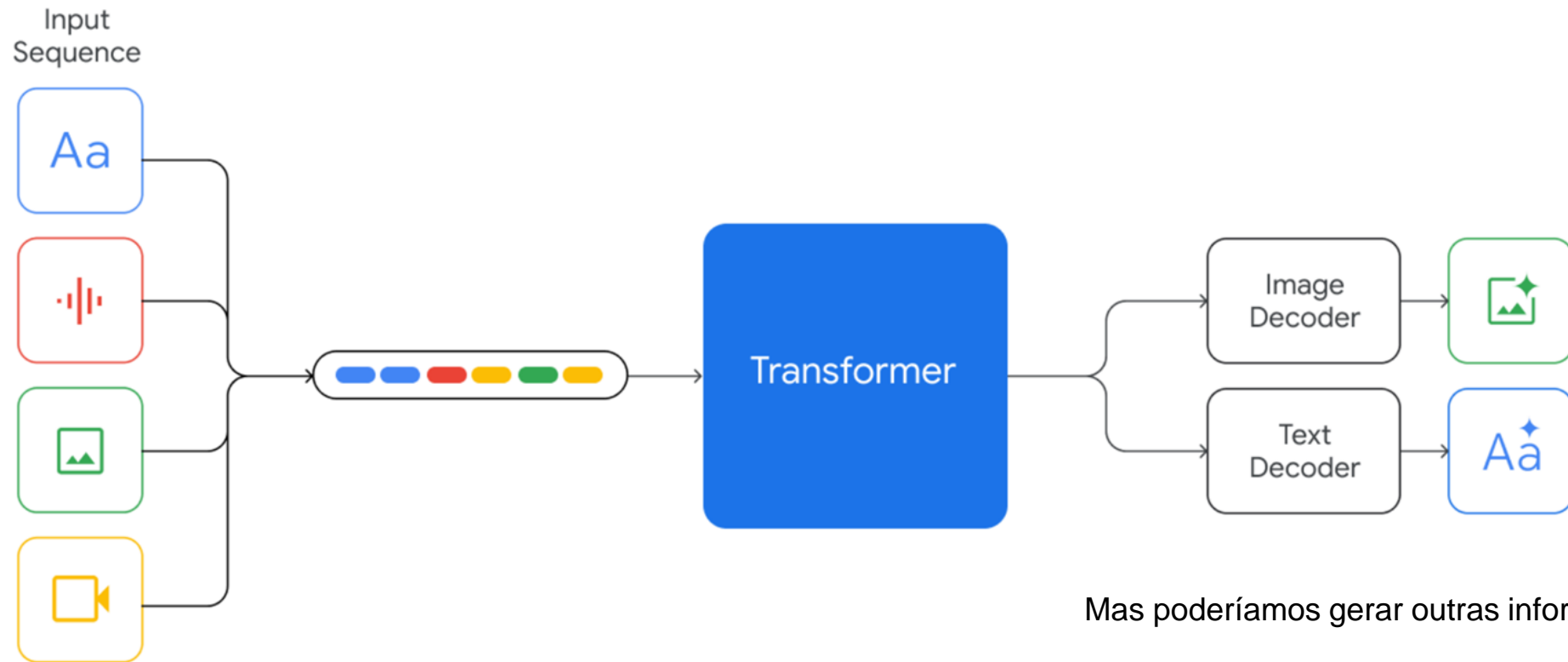
Pro



Nano

(Android AICore)

# Multimodalidade

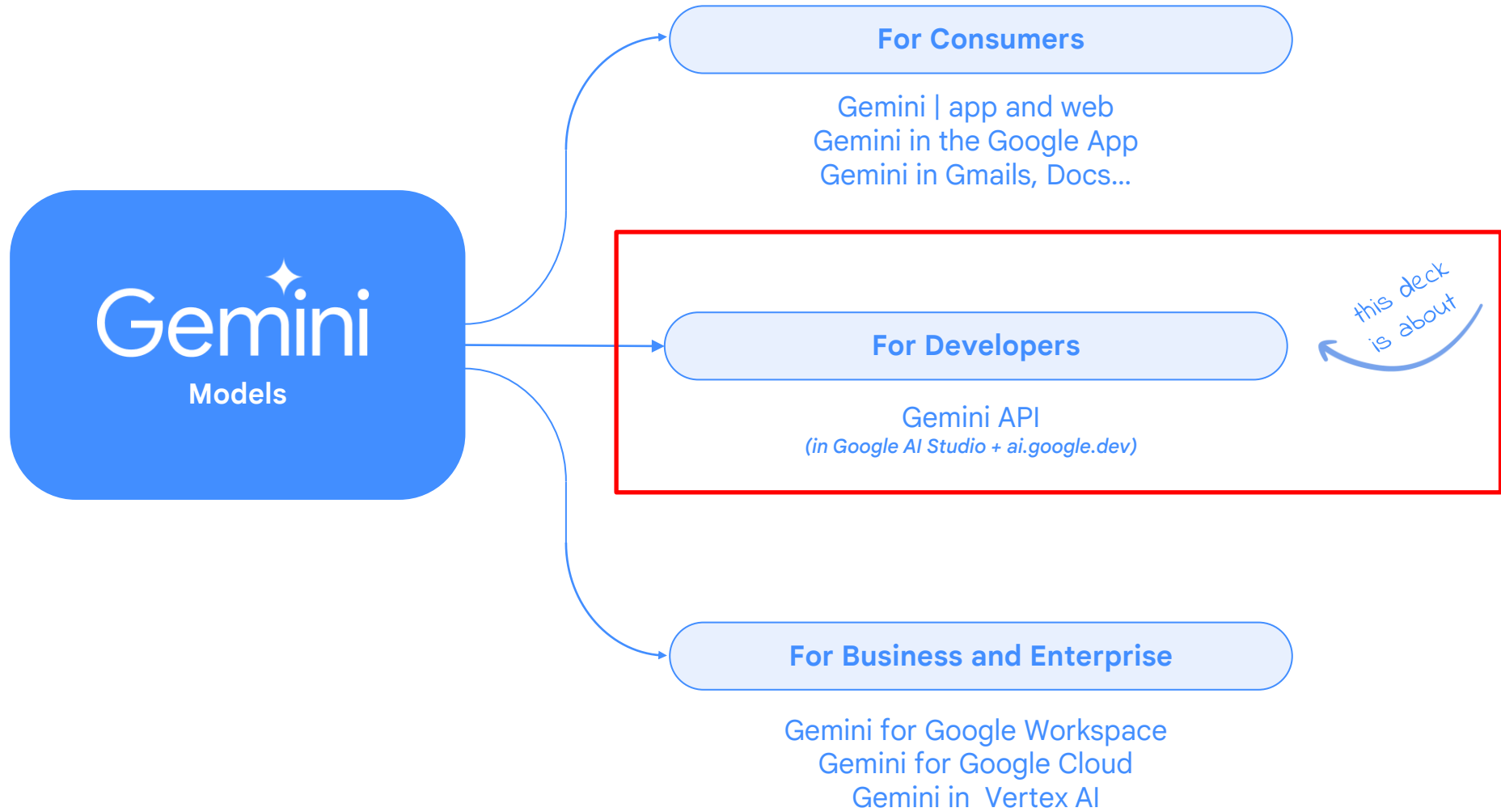


Mas poderíamos gerar outras informações?



# The Gemini Ecosystem

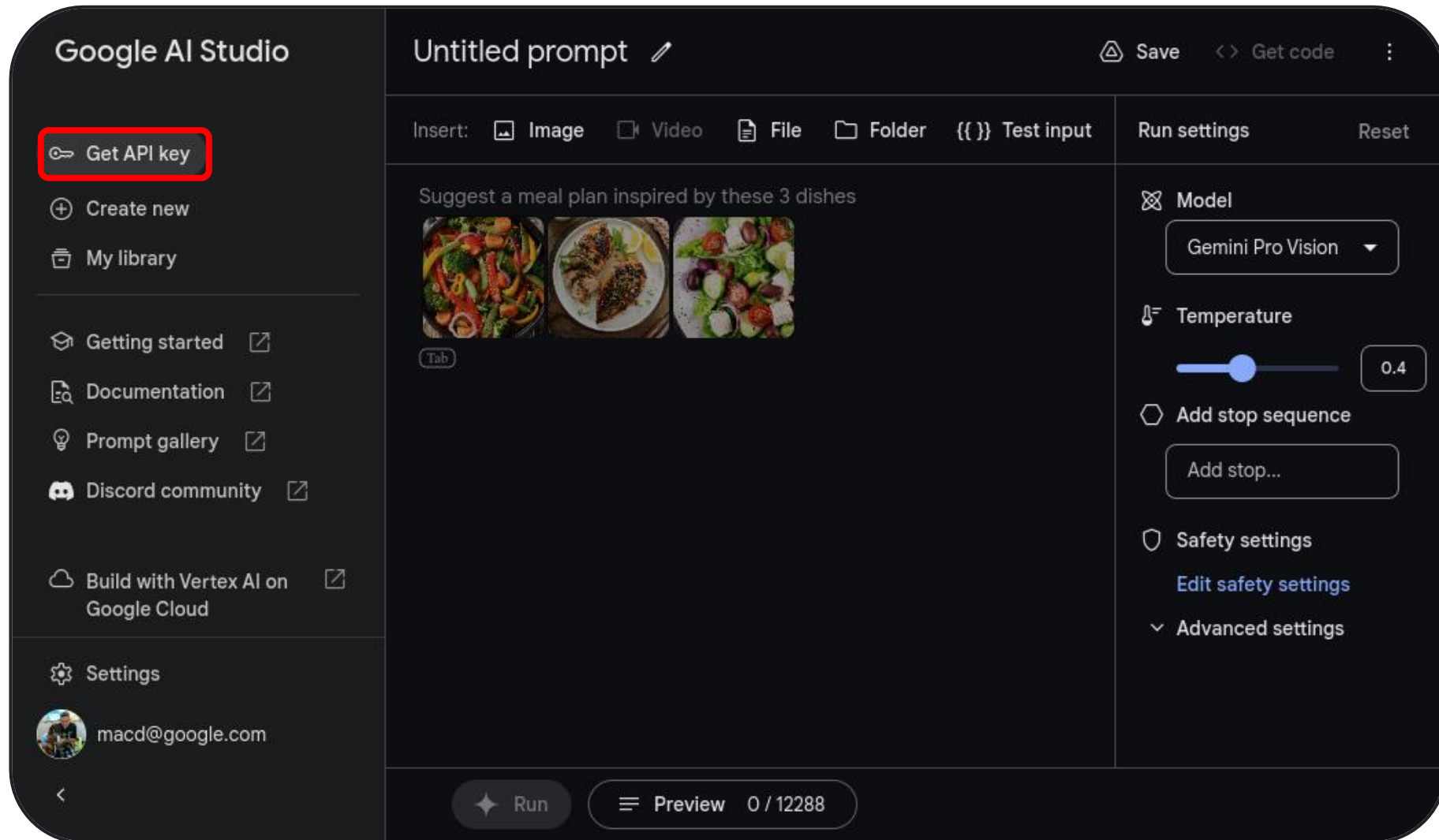
The most advanced AI from Google



[aistudio.google.com](https://aistudio.google.com)

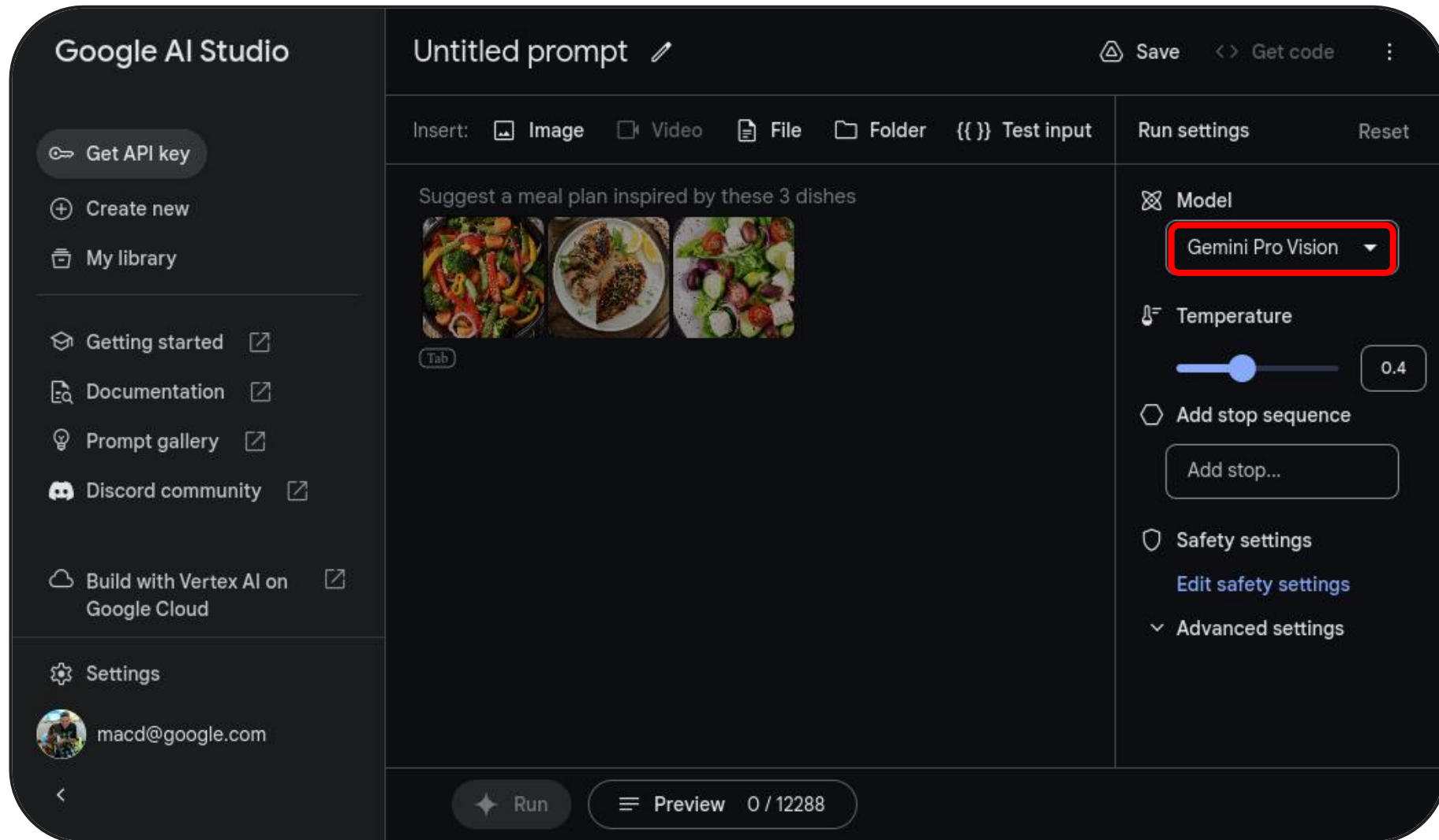
- Gerar chaves de API
- Crie, teste e salve prompts
- Personalize modelos em minutos
- Gerar código inicial

## AI Studio



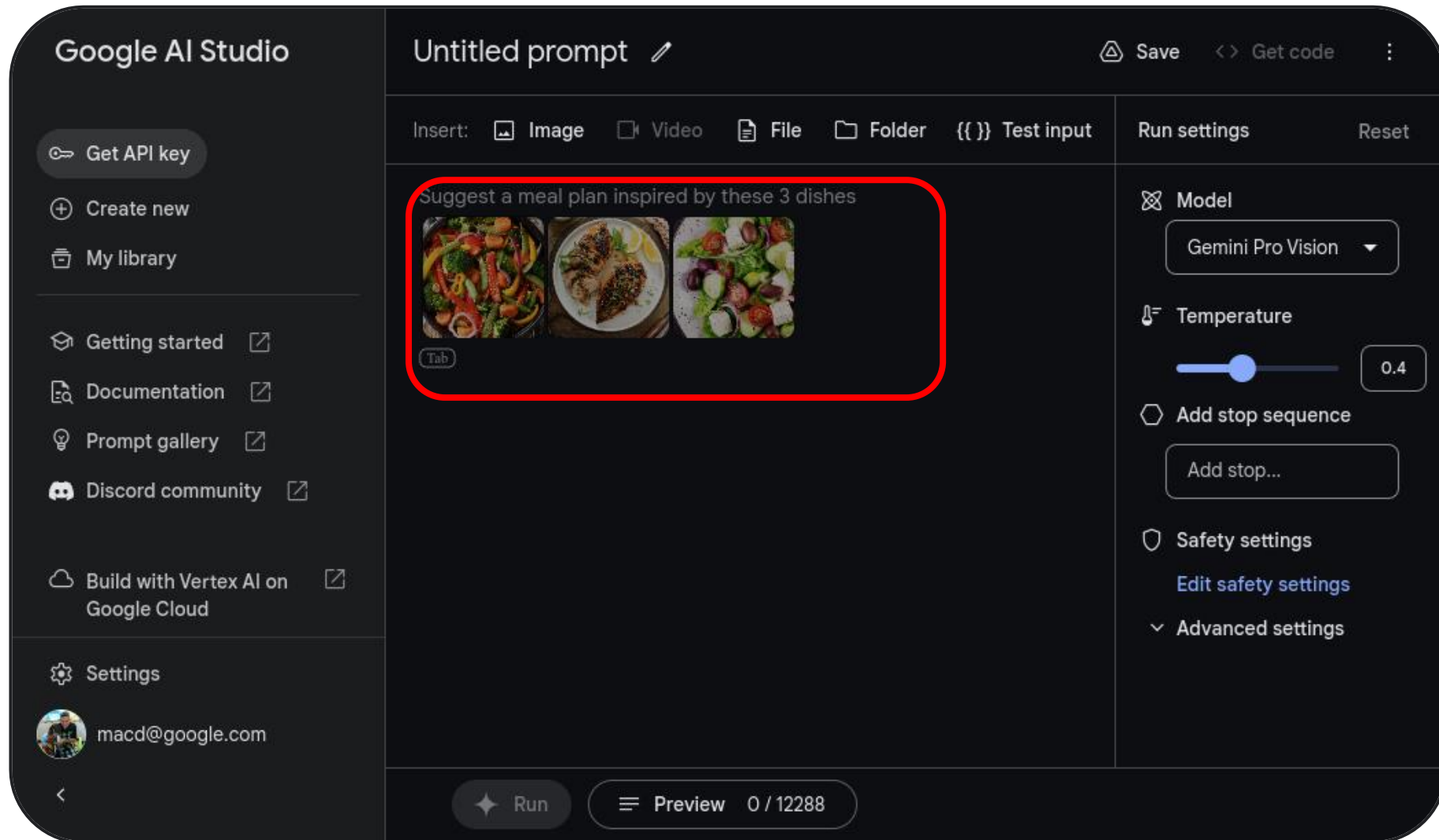
[aistudio.google.com](https://aistudio.google.com)

## AI Studio



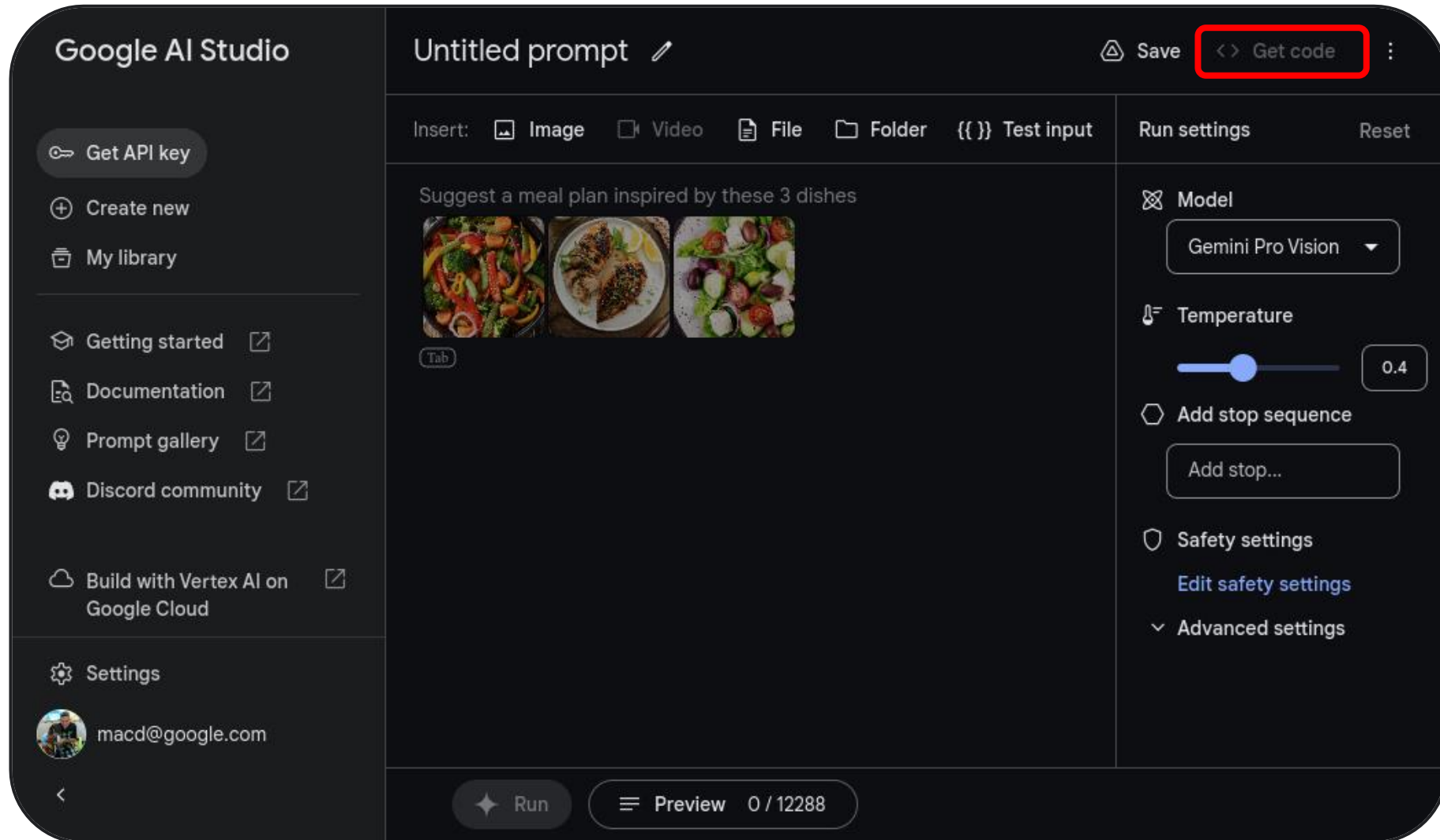
[aistudio.google.com](https://aistudio.google.com)

## AI Studio



[aistudio.google.com](https://aistudio.google.com)

## AI Studio



[aistudio.google.com](https://aistudio.google.com)

## AI Studio

Get code



Create your API key before using the code in your project

You can call this prompt from the Gemini API by copying the following code into your project


cURL


JavaScript

Python

Android (Kotlin)

Swift

 Open in Colab

 Copy

```
1  """
2  At the command line, only need to run once to install the package via pip:
3
4  $ pip install google-generativeai
5  """
6
7  import google.generativeai as genai
8
9  genai.configure(api_key="YOUR_API_KEY")
10
11  # Set up the model
12  generation_config = {
13      "temperature": 2,
14      "top_p": 0.4,
```

SDKs

# ai.google.dev

REST API + Client libraries para Python, Node, Java, and Swift





# Vertex AI

## Gemini API e Vertex AI

Ambos dão acesso aos modelos da família Gemini.

## Vertex AI



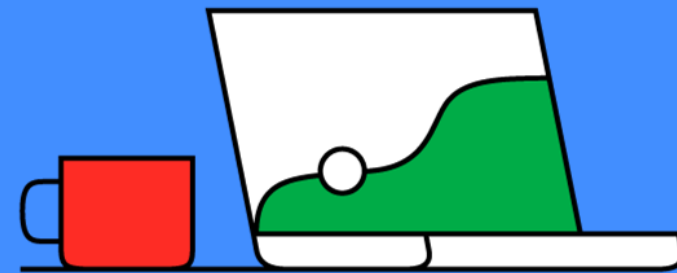
Suporte de nível empresarial.  
MLOps completos (exemplos:  
avaliação de modelo,  
monitoramento, registro)

## Vertex AI



Confira quando estiver  
pronto para produção

[ai.google.dev/docs/migrate\\_to\\_cloud](https://ai.google.dev/docs/migrate_to_cloud)



# Prompt Básico (demonstração)

[aistudio.google.com](https://aistudio.google.com)

Write a poem about a magic backpack.

# aistudio.google.com

Write a poem about a magic backpack.

In a world of wonder, behold,  
A backpack with tales yet untold.  
Its fabric shimmers, alive with might,  
A portal to realms, day and night.  
...

Generate a short tip about JavaScript. Good tips are short and contain a code example showing the tip.

Generate a short tip about JavaScript. Good tips are short and contain a code example showing the tip.

**\*\*Tip:** Use `Array.prototype.flat()` to flatten an array of arrays.\*\*

```
```javascript
const array = [[1, 2], [3, 4], [5, 6]];
console.log(array.flat());
```

```
// Output: [1, 2, 3, 4, 5, 6]
```
```

Generate a short tip about JavaScript. Good tips are short and contain a code example showing the tip.

**topic:** let and const

**tip:** Prefer using `let` and `const` to initialise variables, instead of `var`, as `var` has no block scope or protection against mutability.

e.g. `const myVar = "EXAMPLE";`

**topic:** arrow syntax

**tip:**

## Prompting

Generate a short tip about JavaScript based on the provided topic. Good tips are short and contain a code example showing the tip.

3 / 500 examples

Actions ▾

| <input type="checkbox"/> INPUT topic:     | <input type="checkbox"/> OUTPUT tip:  |
|---|---|
| <input type="checkbox"/> let and const    | <p>Prefer using `let` and `const` to initialise variables, instead of `var`, as `var` has no block scope or protection against mutability.</p> <p>e.g.</p> <pre>const myVar = "EXAMPLE";</pre>                        |
| <input type="checkbox"/> arrow syntax     | <p>Use the arrow syntax `=&gt;` instead of defining anonymous functions. Your code will be clearer and easier to read.</p> <p>e.g.</p> <pre>const doubles = input.map(x =&gt; x + x);</pre>                           |
| <input type="checkbox"/> template strings | <p>Template strings allow you to build multi-line strings using a template and variables from the outer scope.</p> <p>e.g.</p> <pre>const emailHeader = `To: \${recipient} From: \${sender} Subject: Welcome!`;</pre> |

The user's input

The model's response



```
import google.generativeai as genai
```

```
model = genai.GenerativeModel('models/gemini-pro')
```

```
resp = model.generate_content(  
    'Write the first paragraph of a story about a magic backpack')
```

```
>>> print(resp.text)
```

In a bustling city, amidst the vibrant tapestry of human existence, there existed a peculiar entity named Archie. Archie, however, was no ordinary backpack...

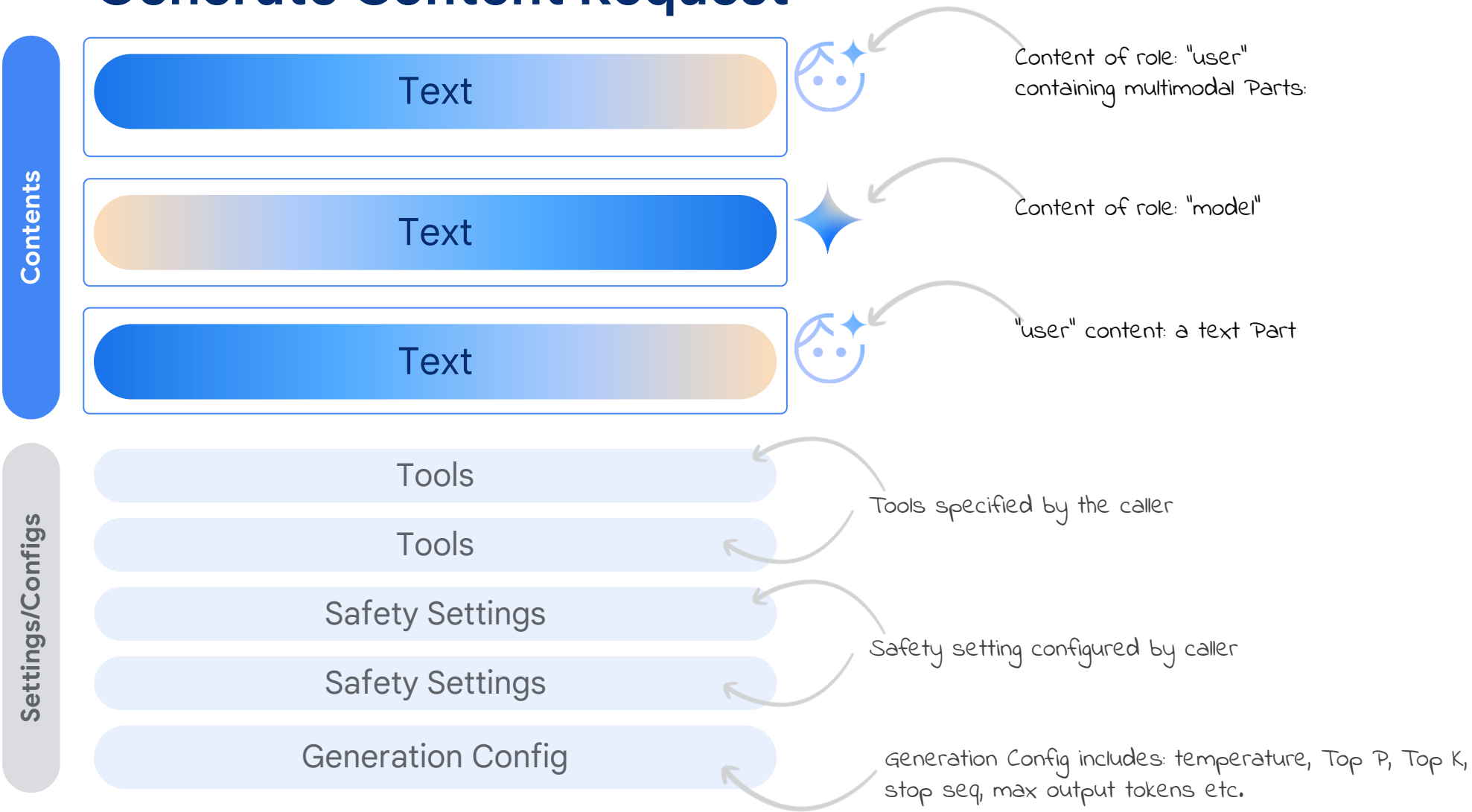
```
chat = model.start_chat()

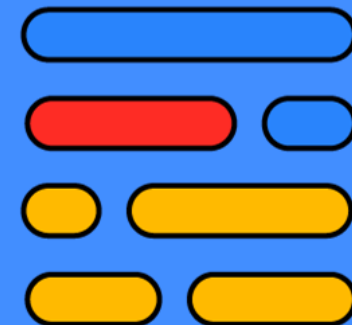
response = chat.send_message(
    "Hello, what should I have for dinner?")

print(response.text)
    # 'Here are some suggestions...'




response = chat.send_message(
    "How do I cook the first one?")
```

# Generate Content Request





# Geração de código e dados

Insert:  Image  Video  File  Folder

Chat



User

Generate a table that has all the countries in the world listed alphabetically, their capitals, and what continent they belong to.

Model



| Country             | Capital          | Continent     |
|---------------------|------------------|---------------|
| Afghanistan         | Kabul            | Asia          |
| Albania             | Tirana           | Europe        |
| Algeria             | Algiers          | Africa        |
| Andorra             | Andorra la Vella | Europe        |
| Angola              | Luanda           | Africa        |
| Antigua and Barbuda | Saint John's     | North America |
| Argentina           | Buenos Aires     | South America |
| Armenia             | Yerevan          | Asia          |



Generate a SQL query that will query for capitals in countries of Asia, following this format: Table countries, columns = [CountryID, CapitalID, Capital]

Model

```
SELECT c.Capital
FROM Capitals c
JOIN Countries co ON c.CountryID = co.CountryID
WHERE co.ContinentID = (
    SELECT ContinentID
    FROM Continents
    WHERE Continent = 'Asia'
);
```



Use code with caution.

SQL

What is the output from this command and table?

Model



The output of the following command:





```
SELECT c.Capital
FROM Capitals c
JOIN Countries co ON c.CountryID = co.CountryID
WHERE co.ContinentID = (
    SELECT ContinentID
    FROM Continents
    WHERE Continent = 'Asia'
);
```

 Use code [with caution](#).

SQL

when executed against the provided table data will be:

Kabul  
Yerevan  
Dhaka

Insert:  Image  Video  File  Folder

## Chat



Generate a Docker script to create a simple linux machine that has python 3.10 installed with following libraries: pandas, tensorflow, numpy

## Model



```
# Create a new Dockerfile
FROM ubuntu:20.04

# Update the package manager
RUN apt-get update -y

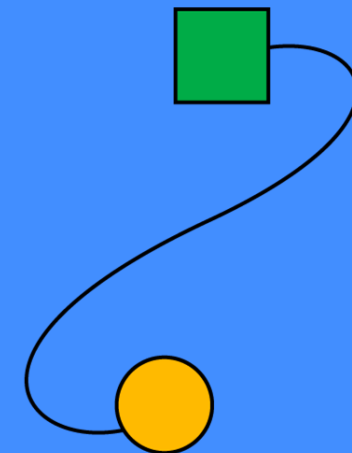
# Install Python 3.10
RUN apt-get install -y python3.10

# Install necessary libraries
RUN pip install pandas tensorflow numpy

# Set the default Python version to 3.10
RUN update-alternatives --install /usr/bin/python python /usr/bin/python3.10 1

# Copy the user's code to the container
WORKDIR /usr/src/app
COPY . .
```





# Síntese de Busca e Informação

# BYO Data

- Os modelos têm limites de conhecimento
- LLMs não são mecanismos de fatos, eles são geradores de texto
- Não tem acesso a dados privados

# Use the prompt's context window

- Instruções + Contexto + Pergunta tudo no prompt
- Fácil de implementar
  - Nenhum código extra, basta perguntar.

```
model = genai.GenerativeModel('gemini-pro')
```

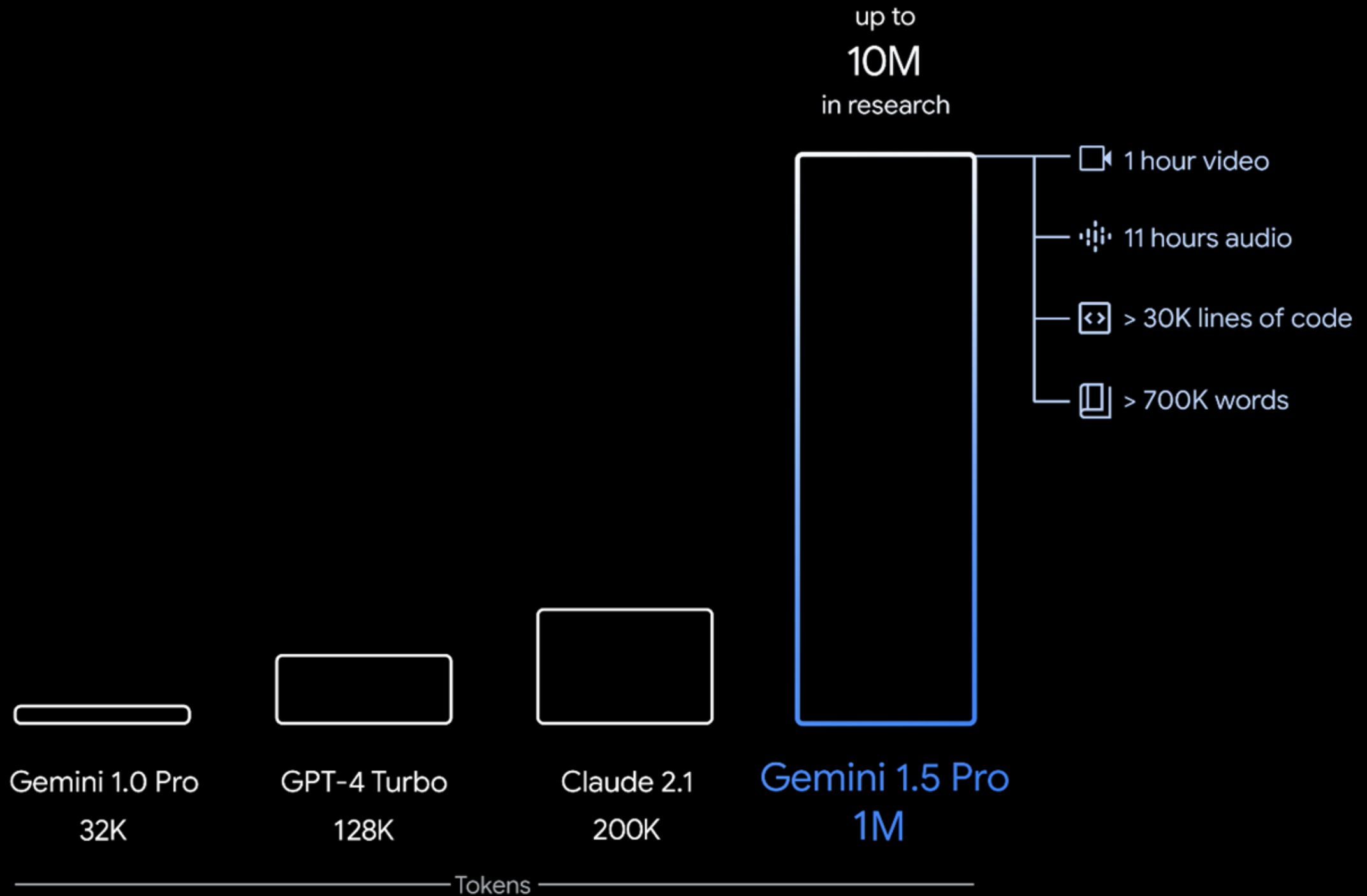
```
document = pathlib.Path('document.txt').read_text()
```

```
result = model.generate_content(f"""  
    Explain how deep-sea life survives.
```


```
    Please answer based on the following document:  
    {document}""")
```

# Use a janela de contexto do prompt

- Limitado pelo comprimento do contexto do modelo
  - gemini-1.0-pro: 30K tokens.





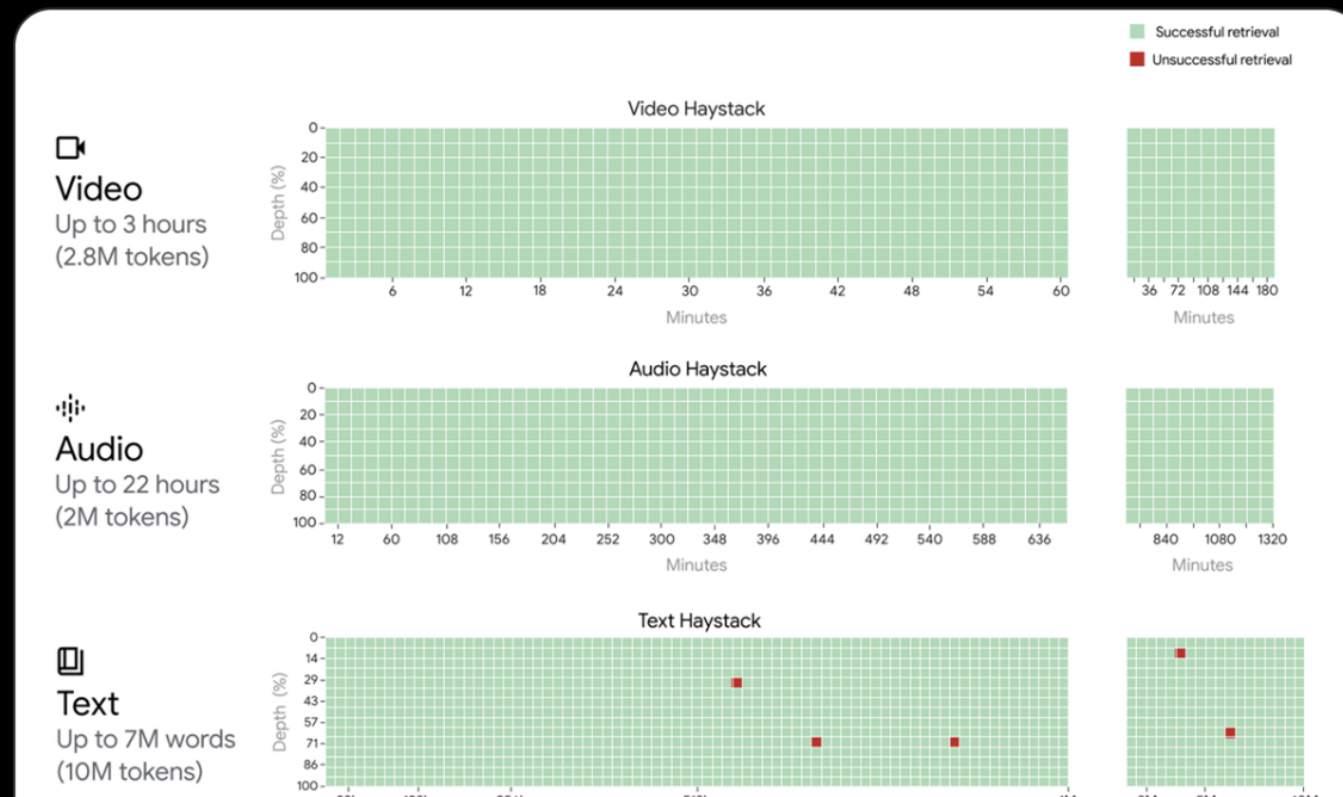
Jeff Dean (@ )

@JeffDean

...

## Needle in a Haystack Tests Out to 10M Tokens

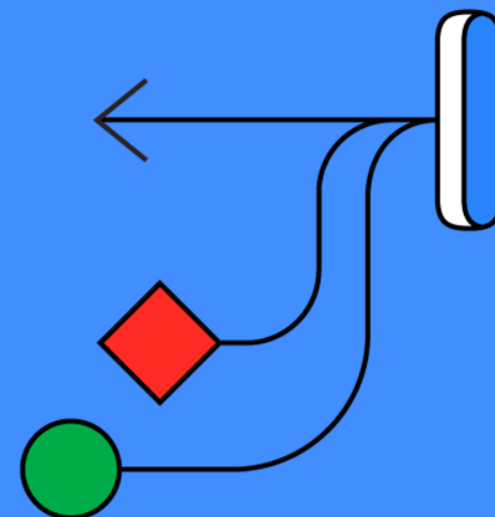
First, let's take a quick glance at a needle-in-a-haystack test across many different modalities to exercise Gemini 1.5 Pro's ability to retrieve information from its very long context. In these tests, green is good, and red is not good, and these are almost entirely green (>99.7% recall), even out to 10M tokens. Great! A bit more on needle-in-a-haystack tests later in the thread.



# Use the prompt's context window

- **gemini-1.5-pro: 1M tokens.**
  - Remember: Everything is tokens.
- Using context may be more flexible than Retrieval Augmented Generation (RAG).
- More input context means higher latency
- Join the waitlist:  
<https://aistudio.google.com/app/waitlist/97445851>





# Chamada de Funções

# Chamada de Funções

- Descrever funções externas ao modelo.
- O modelo pode pedir que você chame a função para ajudá-lo a responder às suas consultas.
- Possibilidades infinitas para integração de ferramentas externas.

```
def multiply(a: float, b: float):  
    """Returns a * b."""  
    return a*b
```

```
model = genai.GenerativeModel(  
    model_name='gemini-1.0-pro',  
    tools=[multiply])
```

# Function calling - More Examples

- Wikipedia research aid
  - Integrates a search tool.
  - Uses the Gemini API inside the function call to summarize pages.

```
def wikipedia_search(queries:list[str]) -> list[str]:  
    ...
```

```
model = genai.GenerativeModel('gemini-pro', tools=[wikipedia_search])
```

```
chat = model.start_chat(enable_automatic_function_calling=True)
```

```
query = "Explain how deep-sea life survives."
```

```
res = chat.send_message(instructions.format(query=query))
```

```
model = genai.GenerativeModel('gemini-pro', tools=[wikipedia_search])
```

```
chat = model.start_chat(enable_automatic_function_calling=True)
```

```
query = "Explain how deep-sea life survives."
```

```
res = chat.send_message(instructions.format(query=query))
```

```
# Searching for "How do deep-sea creatures survive the extreme pressure?"  
# Related search terms: ['Deep sea', 'Deep-sea community', 'Deep-sea fish']  
# Fetching page: "Deep sea"  
# Information Source: https://en.wikipedia.org/wiki/Deep\_sea  
# Fetching page: "Deep-sea community"  
# Information Source: https://en.wikipedia.org/wiki/Deep-sea\_community  
# Fetching page: "Deep-sea fish"  
# Information Source: https://en.wikipedia.org/wiki/Deep-sea\_fish  
# Searching for "How do deep-sea creatures survive the cold temperatures?"  
# Related search terms: ['Deep-sea community', 'Deep sea', 'Deep-water coral']  
# Fetching page: "Deep-water coral"  
# Information Source: https://en.wikipedia.org/wiki/Deep-water\_coral
```



# Interpretando imagens

## Gemini: Guess the movie



Insert:



Image

{{ }} Test input



Guess the name of this movie



# Gemini: Guess the movie

 Save



Insert:




Image

{{ }} Test input



Guess the name of the movie. Explain your reasoning.

Gemini: Guess the book 

Insert:  Image  Video  File  Folder  Test input



Guess the name of this book?

## Gemini: Guess the movie



Insert:



Image

{{ }}

Test input



Guess the name of this movie

The correct answer is 2001: A Space Odyssey.

## Animal puzzle

Insert:  Image  Video  File  Folder `{{ }}` Test input

Solve the puzzle. Explain it step by step.

*What do they weigh?*

$$\text{🐧} + \text{🐧} = 60\text{kg}$$

$$\text{🐧} + \text{🦆} + \text{🦆} = 36\text{kg}$$

$$\text{🦆} + \text{🐓} + \text{🐓} = 23\text{kg}$$

Let's assume the weight of the penguin is P, duck is D and hen is H.

So, as per the given picture, we can write the following equations:

$$P + P = 60$$

$$P + D + D = 36$$

$$D + H + H = 23$$

Let's solve the first two equations to find the value of P and D.

$$P + P = 60$$

$$2P = 60$$

$$P = 30$$

$$P + D + D = 36$$

$$30 + 2D = 36$$

$$2D = 36 - 30$$

$$2D = 6$$

$$D = 3$$

Now, substitute the values of P and D in the third equation to find the value of H.

$$D + H + H = 23$$

$$3 + H + H = 23$$

$$2H = 23 - 3$$

$$2H = 20$$

$$H = 10$$

Therefore, the weight of the penguin is 30 kg, the weight of the duck is 3 kg and the weight of the hen is 10 kg.

```
import google.generativeai as genai

# Use the Gemini vision model.
PRO_VISION = 'models/gemini-pro-vision'
model = genai.GenerativeModel(PRO_VISION)
```

```
!wget -O instrument.jpg -q https://goo.gle/instrument-img
```

```
import PIL.Image
```

```
img = PIL.Image.open('instrument.jpg')
```

```
# Preview the image
```

```
(thumb := img.copy()).thumbnail((200, 200))
```

```
thumb
```



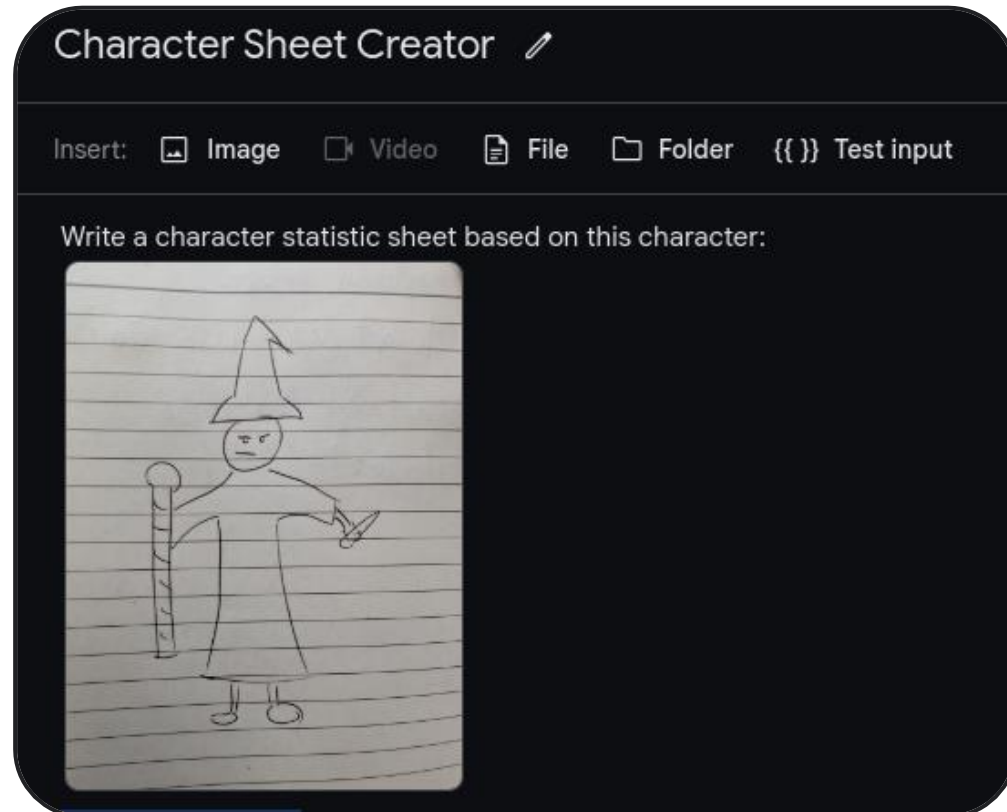
```
response = model.generate_content(  
    ['What instrument is this?',  
     img,  
     'What kinds of music would use it?'])  
  
print(response.text)
```

This is a pipe organ. It is a musical instrument that produces sound by driving pressurized air through pipes. Organs are often used in churches, concert halls, and other large venues. They can be used to play a wide variety of music, from classical to contemporary.

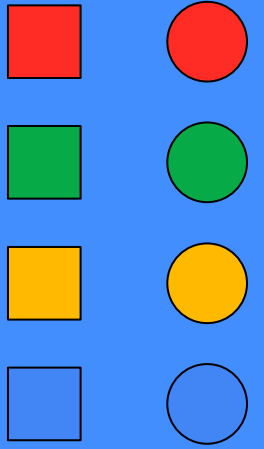


# Condicionamento de imagem

- Gere texto ou dados estruturados a partir de imagens




Name: Stickmancer  
Race: Human  
Class: Wizard  
Level: 1  
Strength: 10  
Dexterity: 14  
Stamina: 12  
Intelligence: 18  
Wisdom: 16  
Charisma: 10  
Equipment: Staff, Dagger, Robe





# Tuning

Train the Trainer in AI


 Get API key



 Create new



 New tuned model

 My library

Loading...

 Getting started 

 Documentation 

 Prompt gallery 


 Discord community 

 Build with Vertex AI on Google Cloud 


## Select data for tuning

You can tune a model from an existing structured prompt or create one by importing from Google Sheets or a CSV. Tuning only works with text at this time. We recommend using 100-500 examples. [Tuning Guide](#).

Structured prompts

 Create a Structured prompt

or

 Import

## Provide details

Tuned model name

Description


## Choose base model

Model


models/gemini-1.0-pro-... ▼

 Create a Structured prompt

or

 Import

## Import examples

 Tuning

### Assign column labels



Use first row as headers



Prefix column names

| Header | inputs | outputs |
|--------|--------|---------|
| Row 1  | 1      | 2       |
| Row 2  | 3      | 4       |

Assign to

New input column



New output column



Avoid importing sensitive or personal information.  
Only text is imported. You can add images directly  
through the UI.

× Close

 Import 15 examples

# Choose base model

## Model

models/gemini-1.0-pro-... ▼

## ^ Advanced settings

Tuning epochs ⓘ

5

Learning rate multiplier ⓘ

1

Batch size ⓘ

4



Tune

# Tuned model results

Tuning details

Increment

Model ID: tunedModels/tuning-9x6pbjld1zo

Base model: Gemini 1.0 Pro 001 (Tuning)

Total training time: Less than 1 min


Tuned examples: 20 examples

Epochs: 5

Batch size: 4

Learning rate: 0.001

Loss / Epochs ⓘ



| Epoch | Loss |
|-------|------|
| 0.0   | 18.5 |
| 0.5   | 20.0 |
| 1.0   | 10.5 |
| 1.5   | 5.0  |
| 2.0   | 2.5  |
| 2.5   | 0.5  |
| 3.0   | 0.0  |
| 3.5   | 0.0  |
| 4.0   | 0.0  |
| 4.5   | 0.0  |
| 5.0   | 0.0  |

## Use your tuned model

☰

Use in freeform prompt

☐

Use in structured prompt

# Tuning

- Utilizando a API:

[https://ai.google.dev/tutorials/tuning\\_quickstart\\_python](https://ai.google.dev/tutorials/tuning_quickstart_python)



```
1 training_data=[
2     {
3         'text_input': '1',
4         'output': '2',
5     }, {
6         'text_input': '3',
7         'output': '4',
8     }, ...
9 ],
```

```
[ ] 1 import random
2
3 name = f'Increment-{random.randint(0,10000)}'
4 operation = genai.create_tuned_model(
5     # You can use a tuned model here too. Set `source_model="tunedModels/..."`
6     source_model=base_model.name,
7     training_data=training_data,
8     id = name,
9     epoch_count = 100,
10    batch_size=4,
11    learning_rate=0.001,
12 )
```



You can use the `genai.generate_text` method and specify the name of your model to test your model performance.

```
[ ] 1 model = genai.GenerativeModel(model_name=f'tunedModels/{name}')
```

```
[ ] 1 result = model.generate_content('55')
    2 result.text
```

```
⇒ '56'
```

```
[ ] 1 result = model.generate_content('123455')
    2 result.text
```

```
⇒ '123456'
```

```
[ ] 1 result = model.generate_content('four')
    2 result.text
```

```
⇒ 'five'
```

```
[ ] 1 result = model.generate_content('quatre') # French 4
    2 result.text                               # French 5 is "cinq"
```

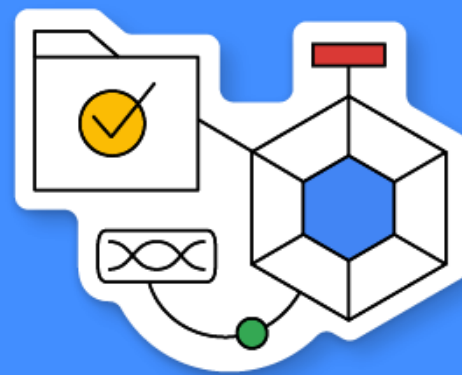
```
⇒ 'cinq'
```

```
▶ 1 result = model.generate_content('III')      # Roman numeral 3
    2 result.text                               # Roman numeral 4 is IV
```

```
⇒ 'IV'
```

```
[ ] 1 result = model.generate_content('七')      # Japanese 7
    2 result.text                               # Japanese 8 is 八!
```

```
⇒ '八'
```



# IA Responsável

## Responsible AI

Toxic content

Block some  
Block medium or high  
probability of being unsafe



Sexual content

Block some



Violent content

Block some



Dangerous content

Block some



Medical content

Block some



[Restore default settings](#)

Learning more

# ai.google.dev

<https://ai.google.dev/docs/discord>

# Obrigado!



Victor Pugliese Ele/Dele

Doutorando UNIFESP

<https://linktr.ee/victorpugliese>

