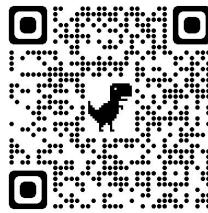


# Intro to With Gemini

Rabimba Karanjai



GDE - AI&ML



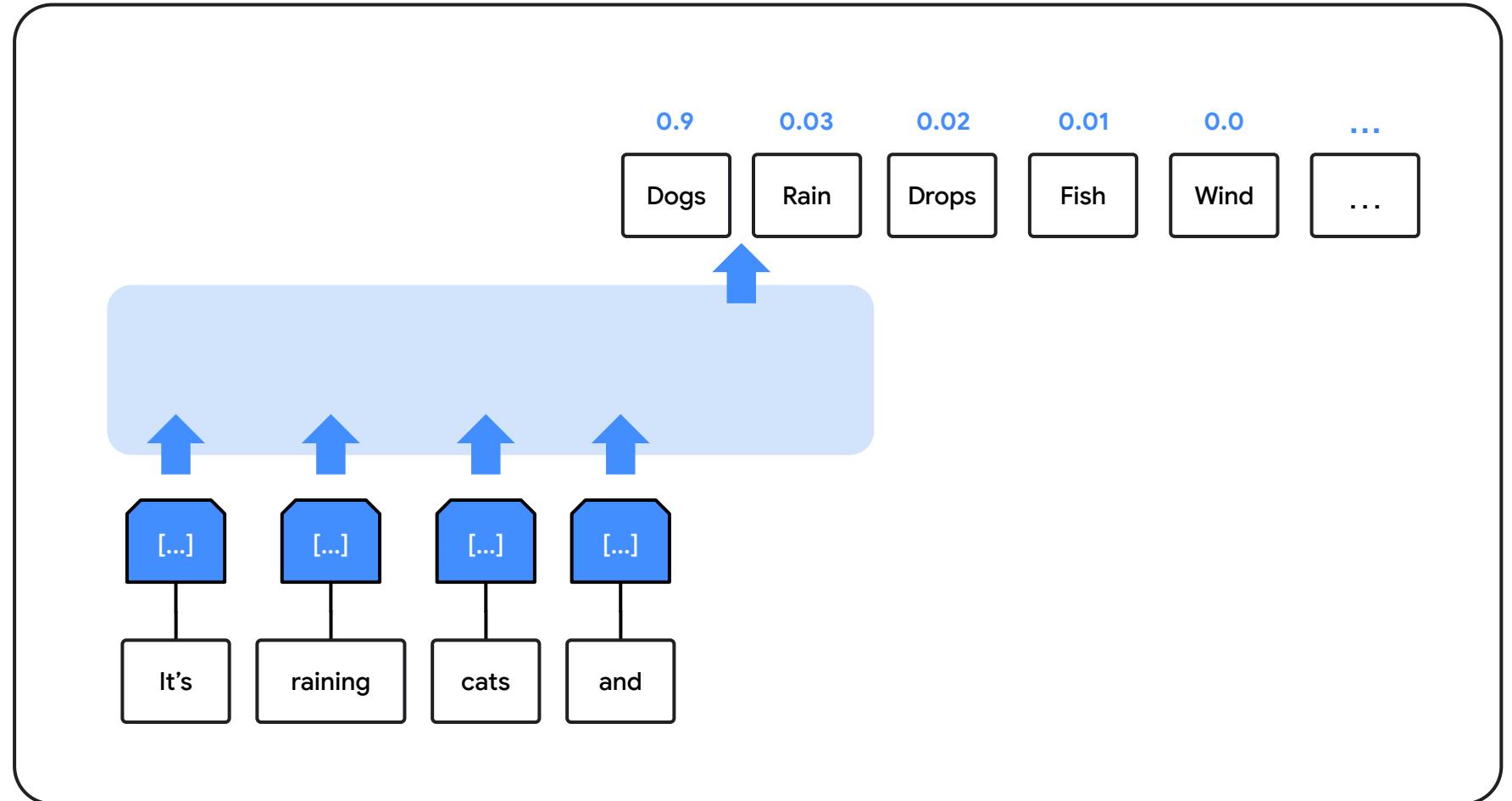
# Rabimba Karanjai

Google Developer Expert - AI&ML  
Cloud Champions Innovator - AI&ML  
Google Cloud Research Innovator

Graduate Researcher - Security & Privacy  
I2C Lab - University Of Houston  
Habanero Lab- Rice University

what is generative ai?

# What is an LLM?



Roses are red,

Roses are red,  
Violets are blue,  
Sugar is sweet,

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

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

Modern LLMs are  
*large.*

# Classic Natural Language Problems

 Entity extraction

 Classification

 Summarization

 Sentiment Analysis

 Translation

 ...

LLMs let us  
prototype *fast*.

## User:

**Explain this joke: “I used to be addicted to soap, but I'm clean now.”**

## Model:

**Model:** The joke is a pun on the word "clean". It has two meanings:

To be free of dirt or other unwanted substances.  
To be free of addiction.

The joke is funny because it is unexpected. The listener expects the speaker to be talking about the first meaning of "clean", but they are actually talking about the second meaning. This creates a surprise and a laugh.

# User:

Give me five neat ideas for science fair projects.

# Model:

## 1. **\*\*Build a volcano.\*\***

This is a classic science fair project that is both fun and educational. You can learn about the science of volcanoes, earthquakes, and plate tectonics.

## 2. **\*\*Make a solar oven.\*\***

This is a great project to learn about renewable energy and how to harness the power of the sun. You can use your solar oven to cook food, dry fruit, or even melt chocolate.

## 3. **\*\*Create a 3D printer.\*\***

This is a more challenging project, but it is also very rewarding. You can learn about 3D printing technology and design your own custom objects.

## 4. **\*\*Build a robot.\*\***

This is a great project for those who are interested in engineering and robotics. You can learn about basic electronics, mechanics, and programming.

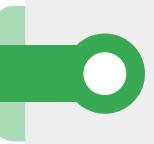
## 5. **\*\*Design a water filtration system.\*\***

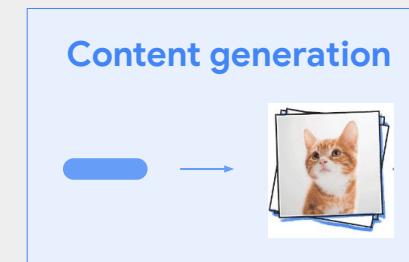
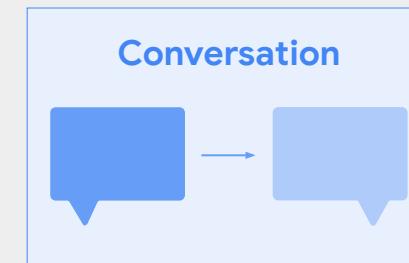
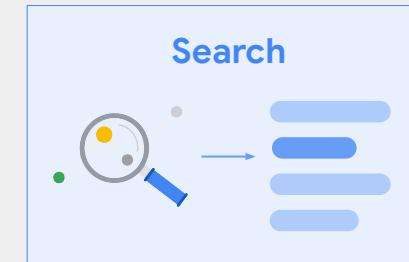
This is a project that can help you learn about the importance of clean water and how to filter it. You can use your water filtration system to clean water from a stream or river.

# Why are large language models different?

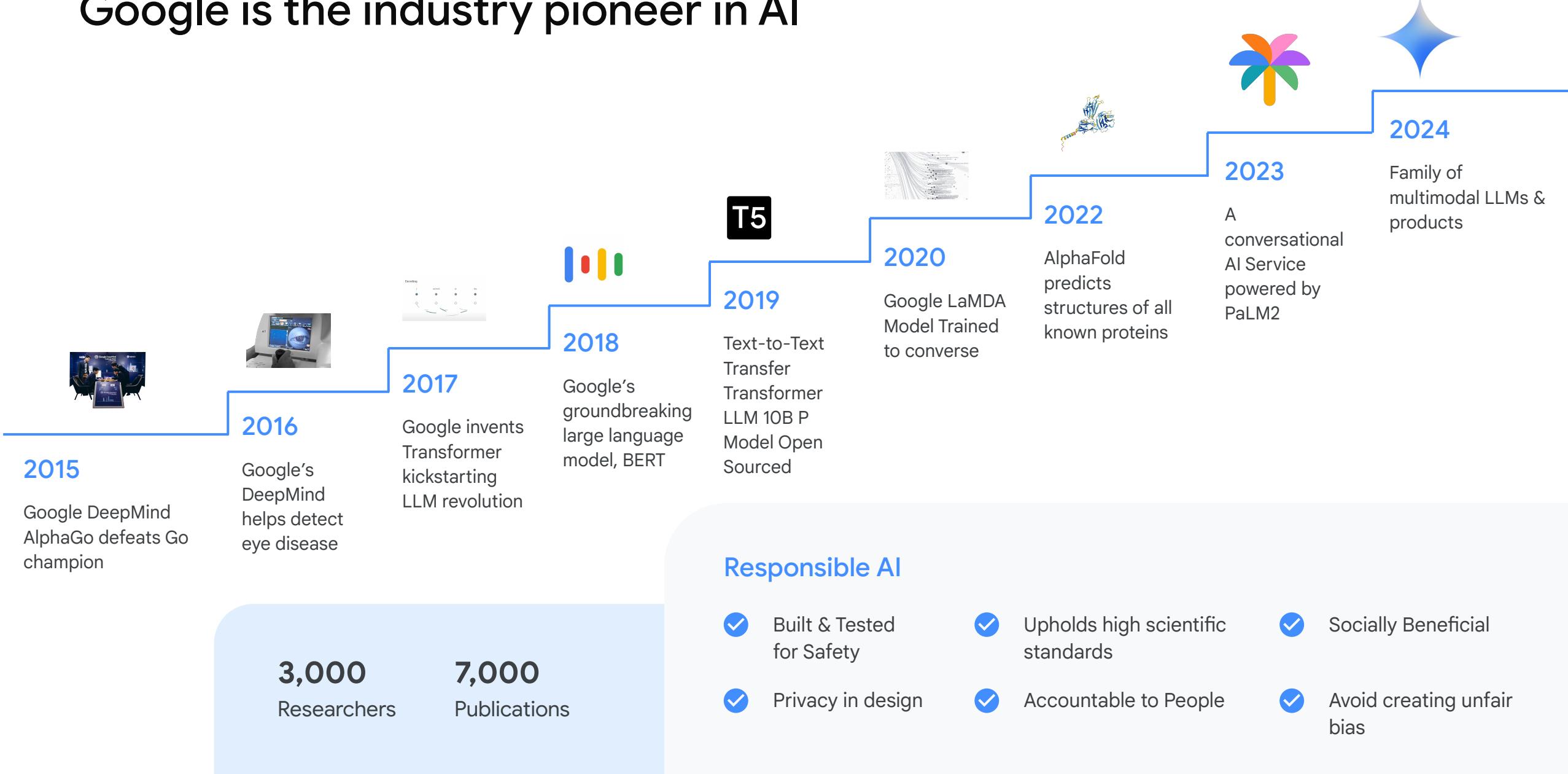
 LLMs are characterized by **emergent abilities**, or the ability to perform tasks that were not present in smaller models.

 LLMs contextual understanding of human language **changes how we interact** with data and intelligent systems.

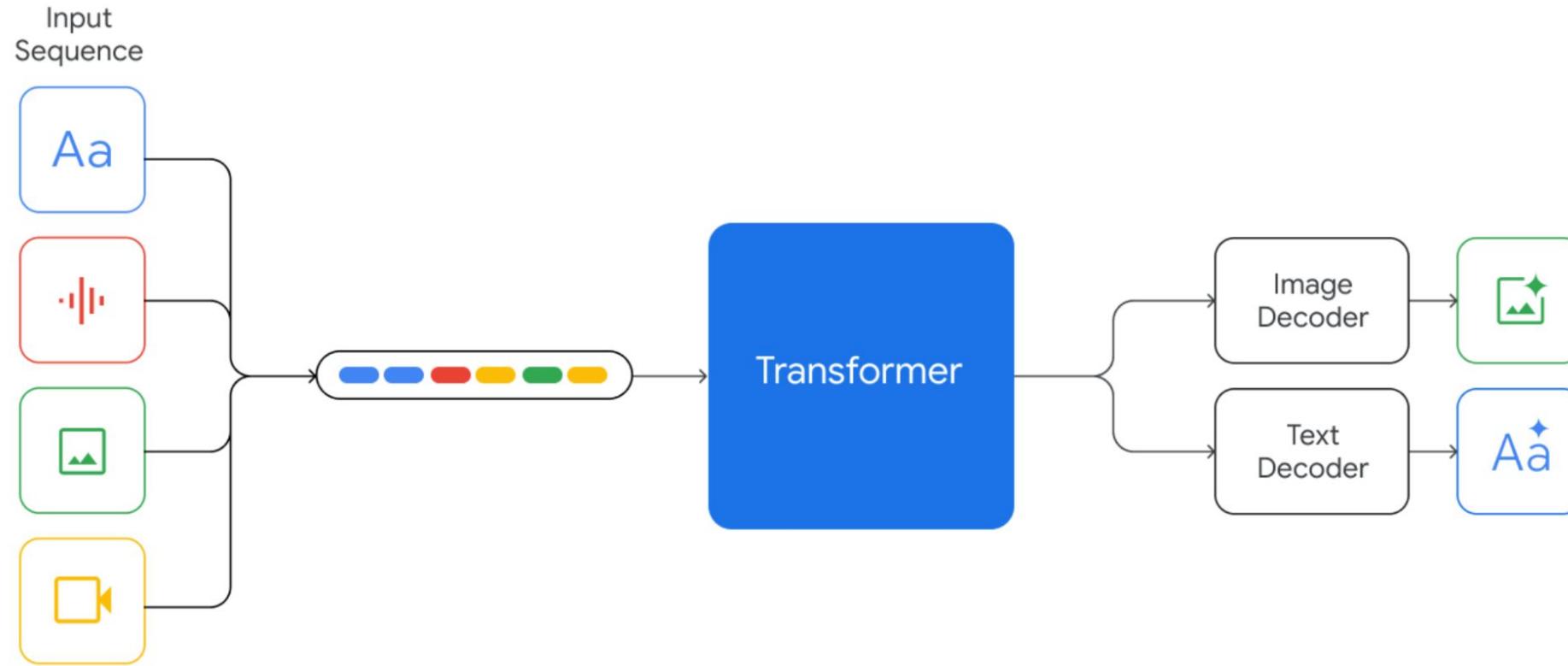
 LLMs can find patterns and connections in **massive, disparate data corpora**.



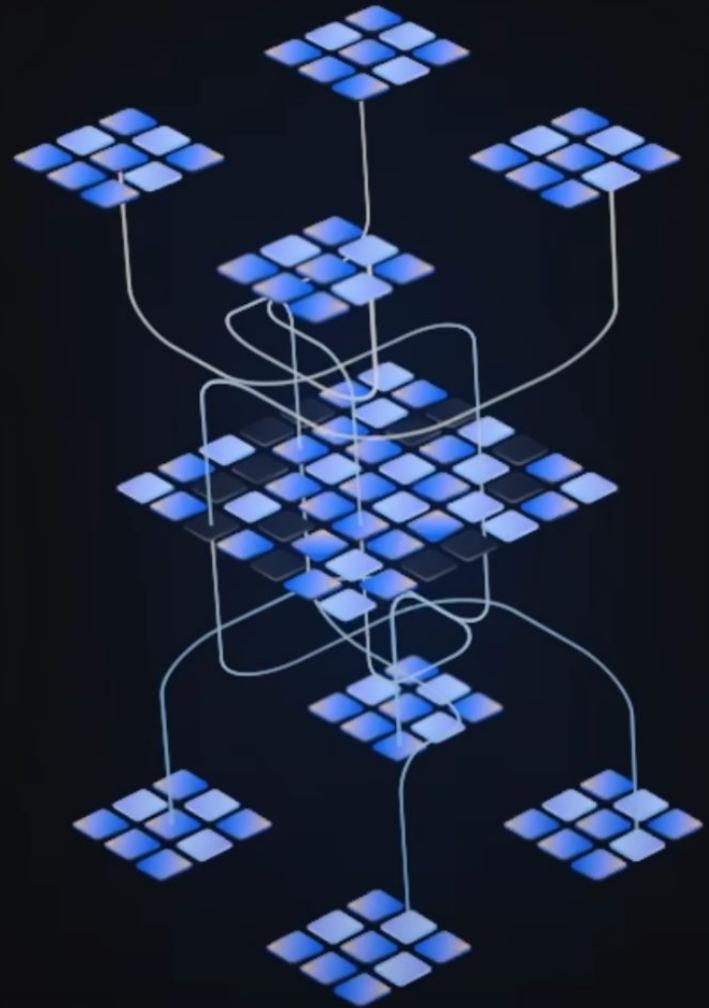
# Google is the industry pioneer in AI



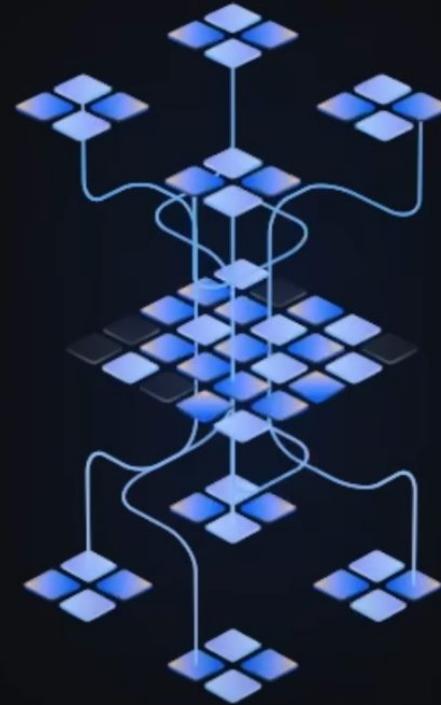
# Multimodality



Welcome to  
the Gemini era

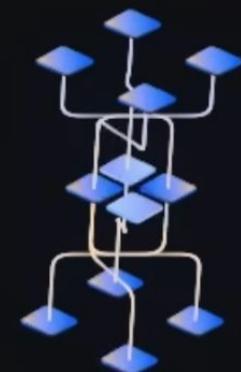


Ultra



Pro

(Android AICore)



Nano

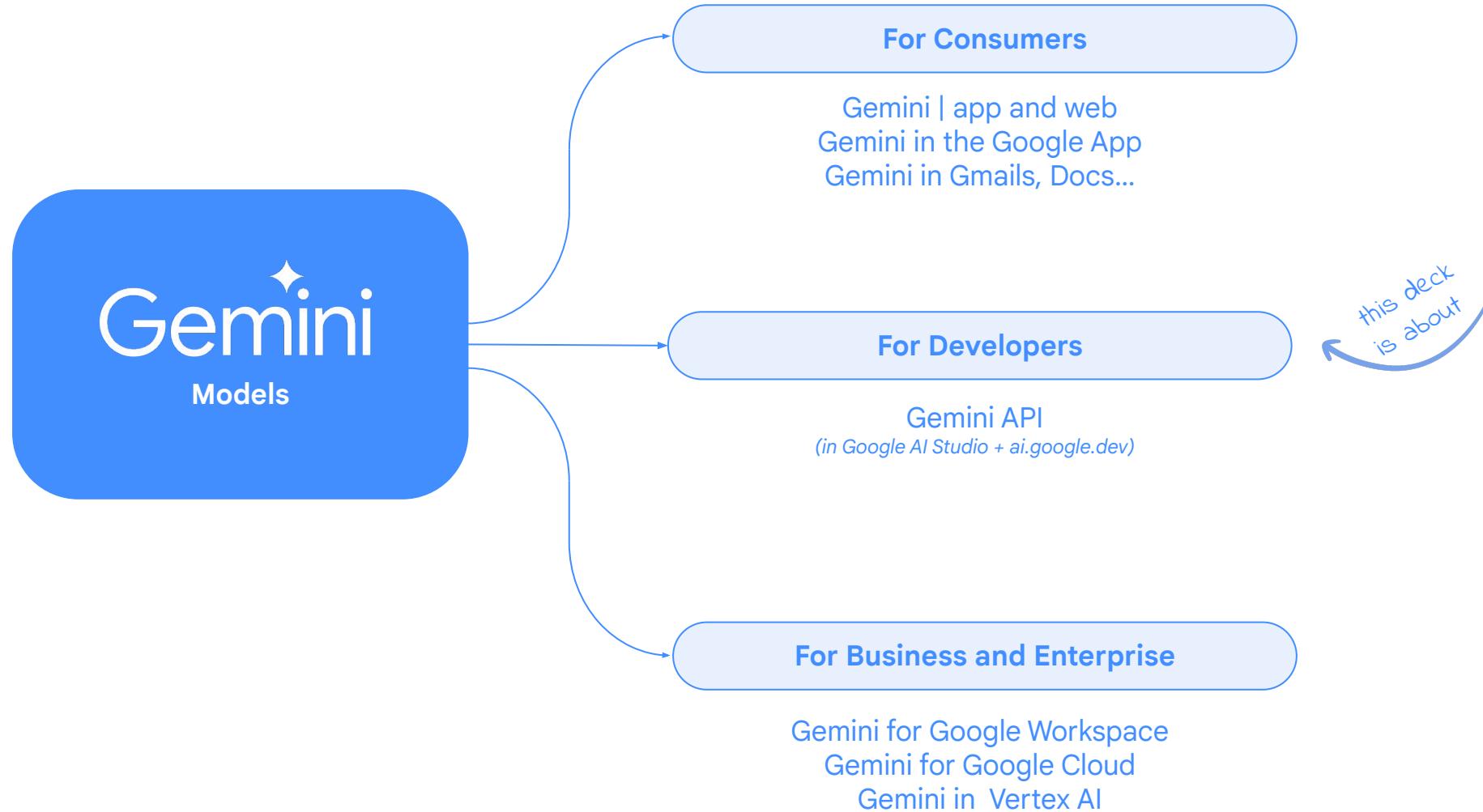


# Gemma Open Models

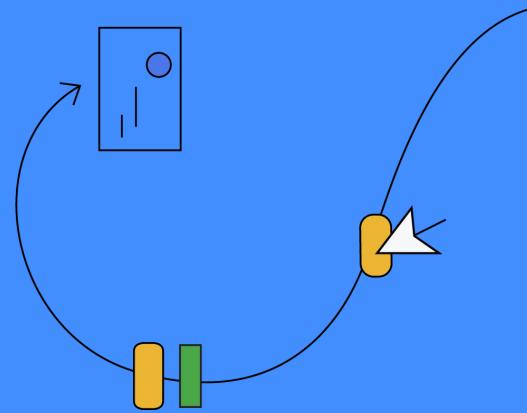
[ai.google.dev/gemma](https://ai.google.dev/gemma)

# The Gemini Ecosystem

The most advanced AI from Google



# Getting started with the Gemini API



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

- Generate API Keys
- Create, test, and save prompts
- Customize models in minutes
- Generate starter code

## AI Studio

Google AI Studio

Untitled prompt

Save Get code :

Insert: Image Video File Folder Test input

Run settings Reset

Suggest a meal plan inspired by these 3 dishes

Model: Gemini Pro Vision

Temperature: 0.4

Add stop sequence: Add stop...

Safety settings: Edit safety settings

Advanced settings

Get API key Create new My library

Getting started Documentation Prompt gallery Discord community

Build with Vertex AI on Google Cloud

Settings

macd@google.com

Run Preview 0 / 12288

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

## AI Studio

### Google AI Studio

Untitled prompt edit

Insert: Image Video File Folder Test input

Run settings Reset

Suggest a meal plan inspired by these 3 dishes



Model: Gemini Pro Vision

Temperature: 0.4

Add stop sequence: Add stop...

Safety settings: Edit safety settings

Advanced settings

Save Get code

Get API key Create new My library

Getting started Documentation Prompt gallery Discord community

Build with Vertex AI on Google Cloud

Settings macd@google.com

Run Preview 0 / 12288

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

## AI Studio

### Google AI Studio

Untitled prompt edit

Save Get code ⋮

Insert: Image Video File Folder Test input

Run settings Reset

Suggest a meal plan inspired by these 3 dishes

Model: Gemini Pro Vision

Temperature: 0.4

Add stop sequence: Add stop...

Safety settings: Edit safety settings

Advanced settings

Getting started Documentation Prompt gallery Discord community

Build with Vertex AI on Google Cloud

Settings

macd@google.com

Run Preview 0 / 12288

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

Get code X

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,
15     ".": 0.0
```

## AI Studio

Google AI Studio

Untitled prompt edit

Insert: Image Video File Folder Test input

Save Get code ⋮

Run settings Reset

Model Gemini Pro Vision

Temperature 0.4

Add stop sequence Add stop...

Safety settings Edit safety settings

Advanced settings

Suggest a meal plan inspired by these 3 dishes

Getting started Documentation Prompt gallery Discord community

Build with Vertex AI on Google Cloud

Settings

macd@google.com

Run Preview 0 / 12288

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

SDKs

# ai.google.dev

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



# Vertex AI

## Gemini API and Vertex AI

Both give access Gemini family models.

## Vertex AI

Enterprise grade support.  
Full MLOps (Examples: Model evaluation, monitoring, registry)



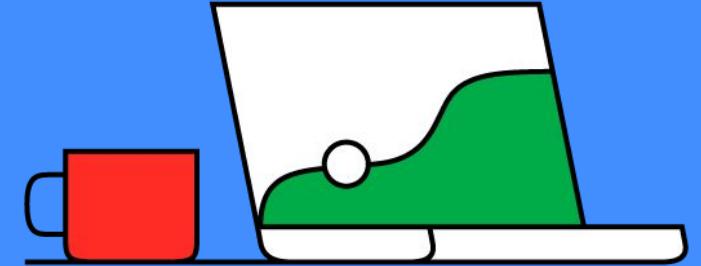
## Vertex AI

Check it out when you're ready for production



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

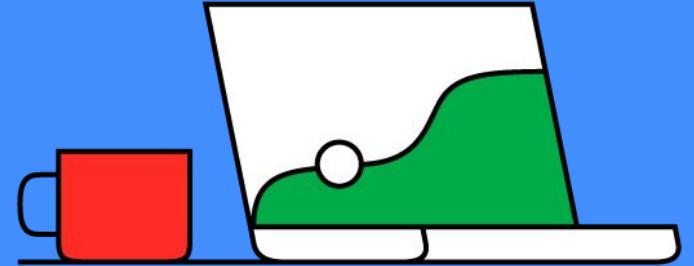
# Usage patterns



# We'll cover

- Basic Prompting
- Code generation
- Search, retrieval and information synthesis
- Function calling
- Image understanding
- Tuning

# Basic prompting



aistudio.google.com

Write a poem about a magic backpack.

[aistudio.google.com](http://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 topic:

: OUTPUT tip:

⋮

let and const

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";
```

arrow syntax

Use the arrow syntax `=>` instead of defining anonymous functions. Your code will be clearer and easier to read.

e.g.

```
const doubles = input.map(x => x + x);
```

template strings

Template strings allow you to build multi-line strings using a template and variables from the outer scope.

e.g.

```
const emailHeader = `To: ${recipient}\nFrom: ${sender}\nSubject: Welcome!`;
```

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...
```

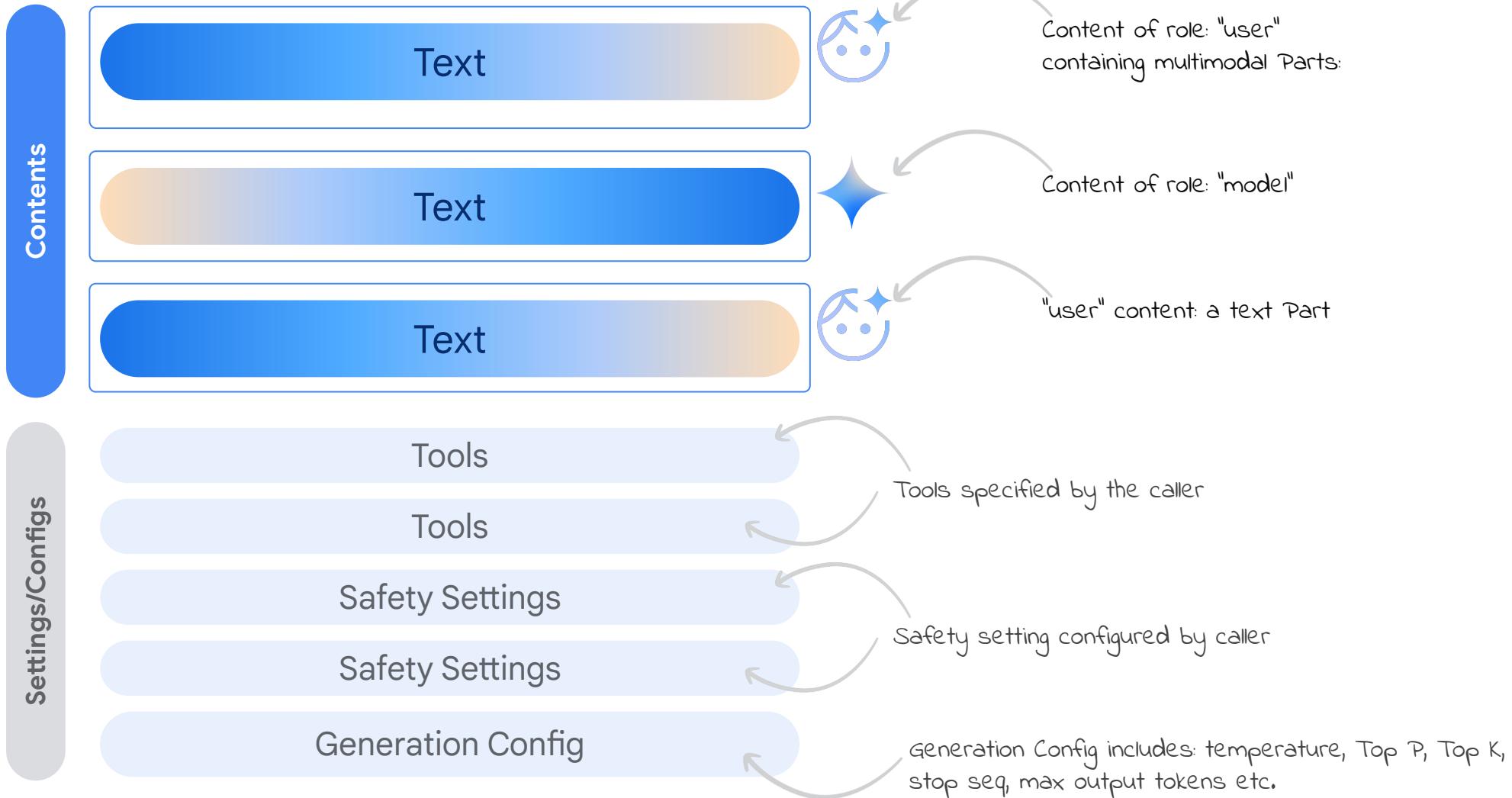
```
chat = model.start_chat()

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



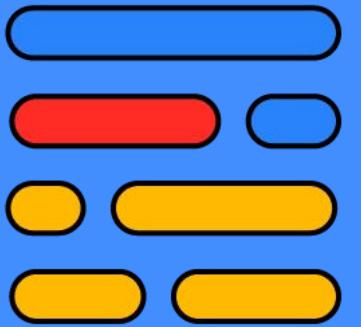
# Generate Content Response

|              |                |  |
|--------------|----------------|--|
| Candidate(s) | Text           | Candidate "Content"<br>note: only one candidate returned today |
|              | Finish Reason  | why the model stopped generating                               |
|              | Finish Message | set if finish reason is present                                |
|              | Safety Ratings | how safe is the response                                       |
|              | Safety Ratings |  |
|              | ...            |  |
|              | Block Reason   | feedback on the prompt   |
| Feedback     | Safety Ratings |  |

## Prompting

- Chained prompts - make a plan, then execute it
- Context - Few shot prompts
- Generation parameters - Temperature, Safety settings, Top-P, TopK

[https://ai.google.dev/docs/prompt\\_best\\_practices](https://ai.google.dev/docs/prompt_best_practices)



# Code generation

Content generation

# Code Generation

Generate code based on a user prompt

Convert human language requests to machine requests

Rapid tool building

# Examples

- Generate data

Insert: Image Video File Folder

## Chat

C



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          |

# Examples

- Generate data
- Generate a SQL query



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

# Examples

- Generate data
- Generate a SQL query
- Simulate Execution

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

# Examples

- Generate data
- Generate a SQL query
- Simulate Execution
- Quick coding help - Smart pair programmer.

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 . .
```

Write a unit test for this Python function:

```
def is_diver_neutrally_buoyant(density_water, density_diver):
    """Check if a diver is neutrally buoyant."""
    if density_water == density_diver:
        return True
```

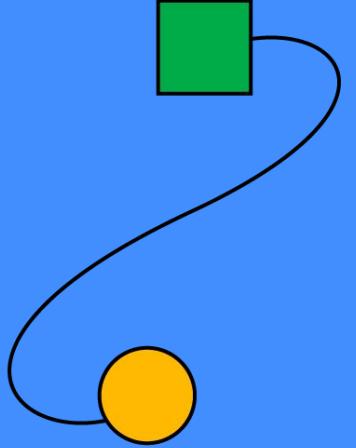
```
import unittest

class IsDiverNeutrallyBuoyantTest(unittest.TestCase):

    def test_diver_is_neutrally_buoyant(self):
        self.assertTrue(is_diver_neutrally_buoyant(1000, 1000))

    def test_diver_is_not_neutrally_buoyant(self):
        self.assertFalse(is_diver_neutrally_buoyant(1000, 1010))

    def test_invalid_input_types(self):
        with self.assertRaises(TypeError):
            is_diver_neutrally_buoyant("1000", 1000)
        with self.assertRaises(TypeError):
            is_diver_neutrally_buoyant(1000, "1000")
```



# Search and Information Synthesis

Train the Trainer in AI

# BYO Data

- Models have knowledge cut-offs
- LLMs are not fact engines
- No exposure to private data

# Use the prompt's context window

- Instructions + Context + Question all in the prompt
- Easy to implement
  - No extra code, just ask.

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

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

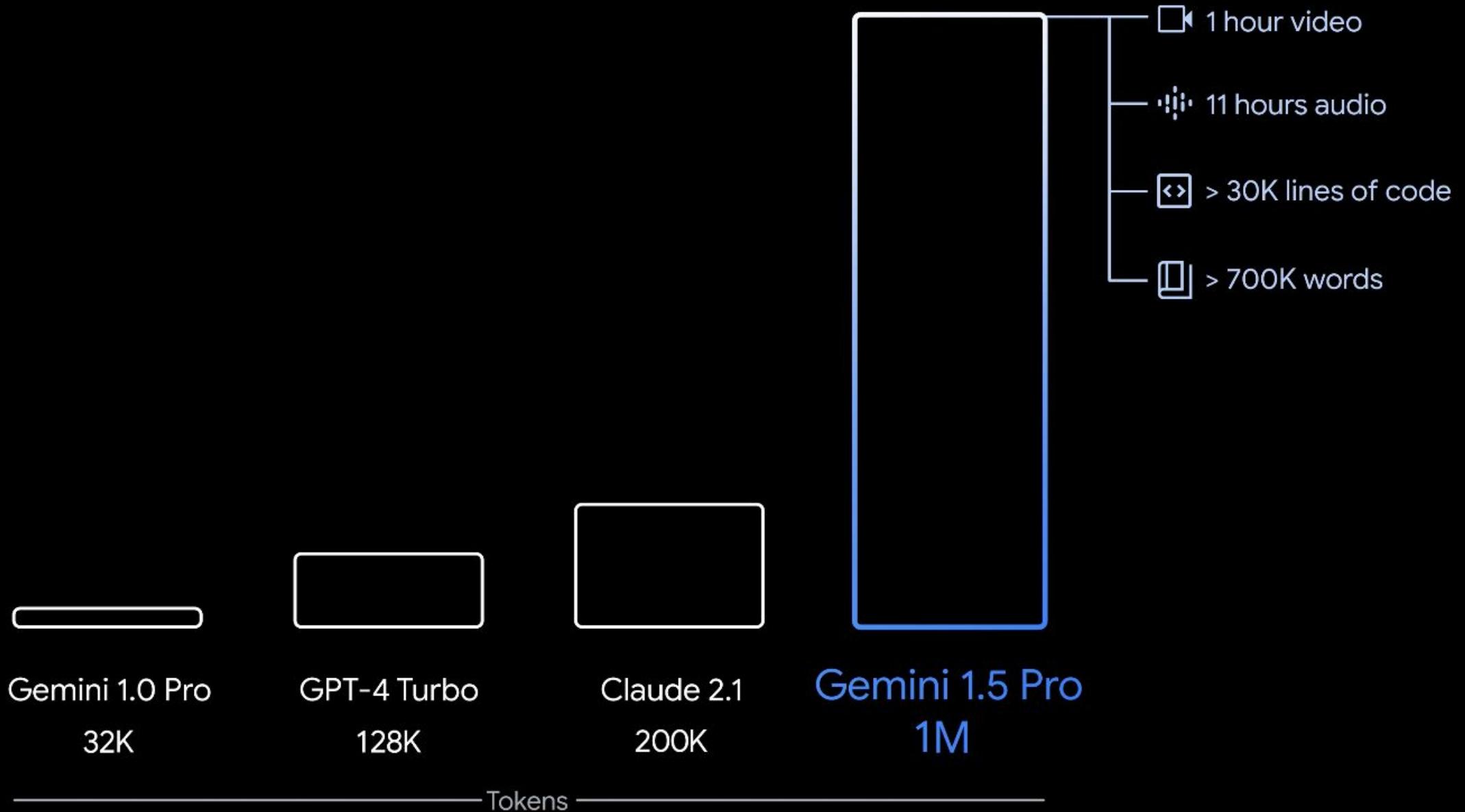
result = model.generate_content(f"""
    Explain how deep-sea life survives.

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

# Use the prompt's context window

- Limited by the model's context length
  - gemini-1.0-pro: 30K tokens.

up to  
**10M**  
in research



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.

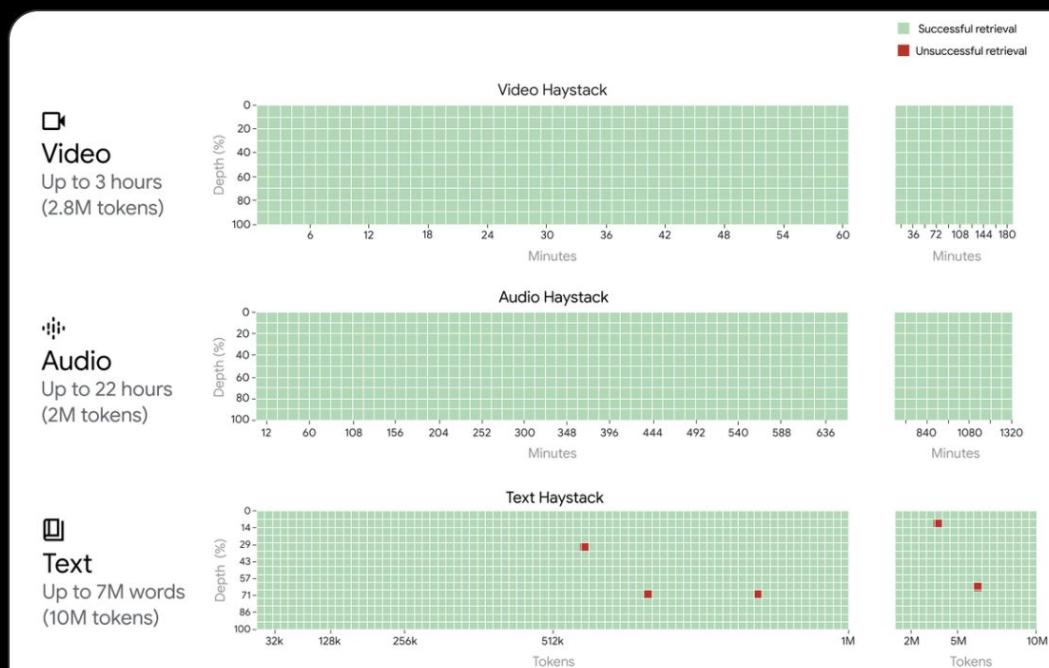


Figure 1 | Gemini 1.5 Pro achieves near-perfect “needle” recall (>99.7%) up to 1M tokens of “haystack” in all modalities, i.e., text, video and audio. And even maintaining this recall performance when extending to 10M tokens in the text modality (approximately 7M words); 2M tokens in the audio modality (up to 22 hours); 2.8M tokens in the video modality (up to 3 hours). The x-axis represents the context window, and y-axis the depth percentage of the needle placed for a given context length. The results are color-coded to indicate: green for successful retrievals and red for unsuccessful ones.

# 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>

# Join the Gemini 1.5 Pro model waitlist

We're excited to share our next generation model that achieves a context window of up to 1 million tokens and introduces a breakthrough experimental capability of long-context understanding. Due to high demand, we're gradually rolling out access to some of our newer models. Register below and we'll send you an email when it's your turn.

**Country/Region\***

Select your country/region



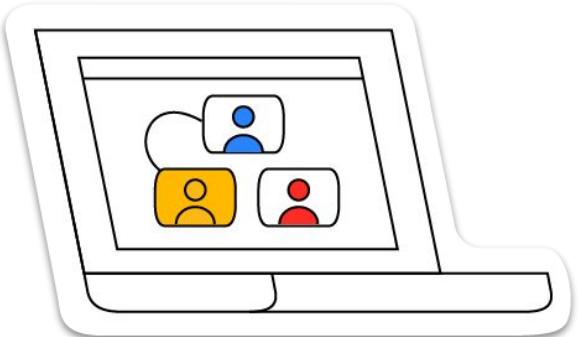
Select your country/region.

**Profession**

(select all that apply)

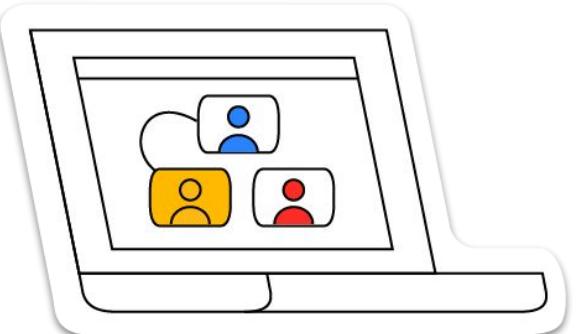
- Academic/Researcher
- Data Scientist/Analyst
- Designer
- Executive/Business owner
-

Learning more



# ai.google.dev

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



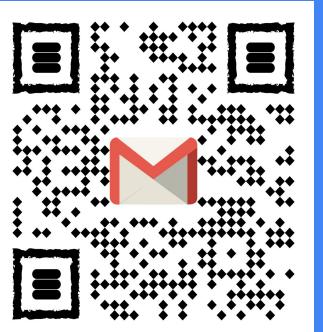
## Docs:

- <https://cloud.google.com/vertex-ai/generative-ai/docs/multimodal/function-calling>
- <https://github.com/GoogleCloudPlatform/generative-ai/>
- <https://github.com/rabimba/GDE-ML-Artifacts/>

Connect with me



Train the  
Trainer in AI



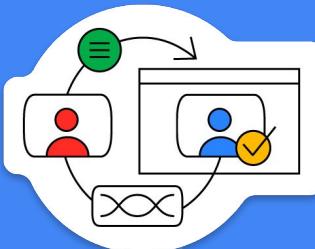
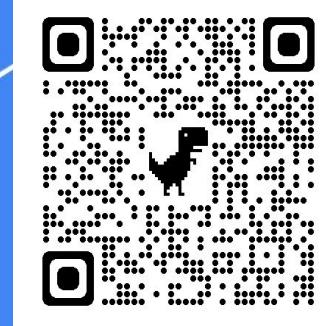
# Thank you!



Rabimba Karanjai

[rabimba@gmail.com](mailto:rabimba@gmail.com)

Link or socials, maybe?



Train the Trainer in AI