

# Retreival Augmented Generation

Intro to RAG Workshop



# Aim

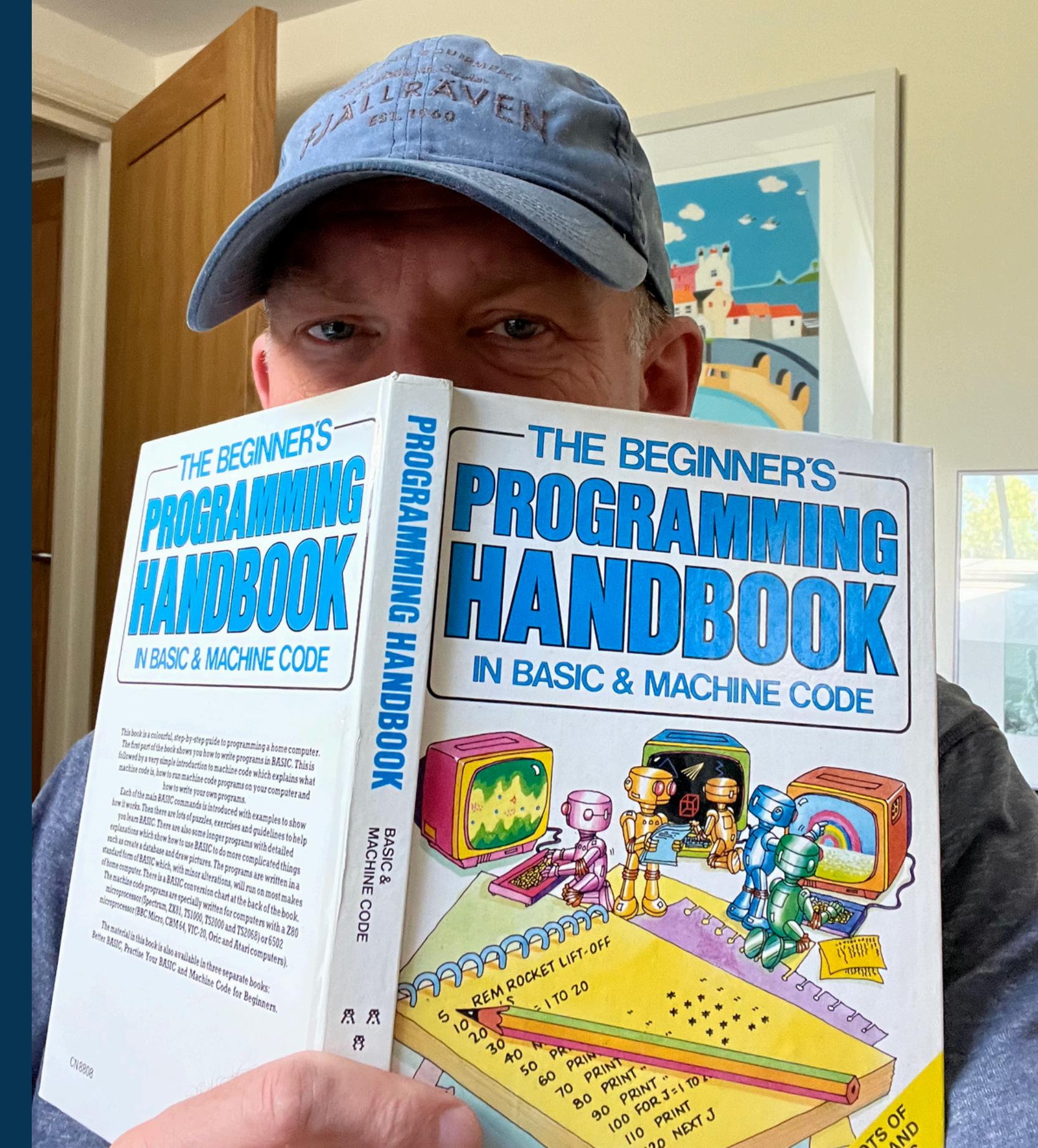
To leave this session with  
enough information to build  
your own RAG chatbot.



# My Name Is...

# Dan Gibson

## “Coding since 1983!”





## Product

Personify Chat - Virtual Ecommerce Sales

## Services

Custom Build

Advice and Training

# Working Agreements

Stop me if you have a question

One conversation at a time

Let me “park” in-depth or off-topic questions

If you know the answer, help teach others

# In Your Team

Introductions

Decide on the Laptop Setup

IDE choice

# Activities

- 1 Introducing RAG
- 2 Using Vector Stores & Embeddings
- 3 Storage Strategies - Chunking
- 4 Generating Answers with OpenAI



Enhance Large Language  
Model responses by retrieving  
and incorporating data from  
external sources.



# Why?

Inject your organisations data  
into the conversation.

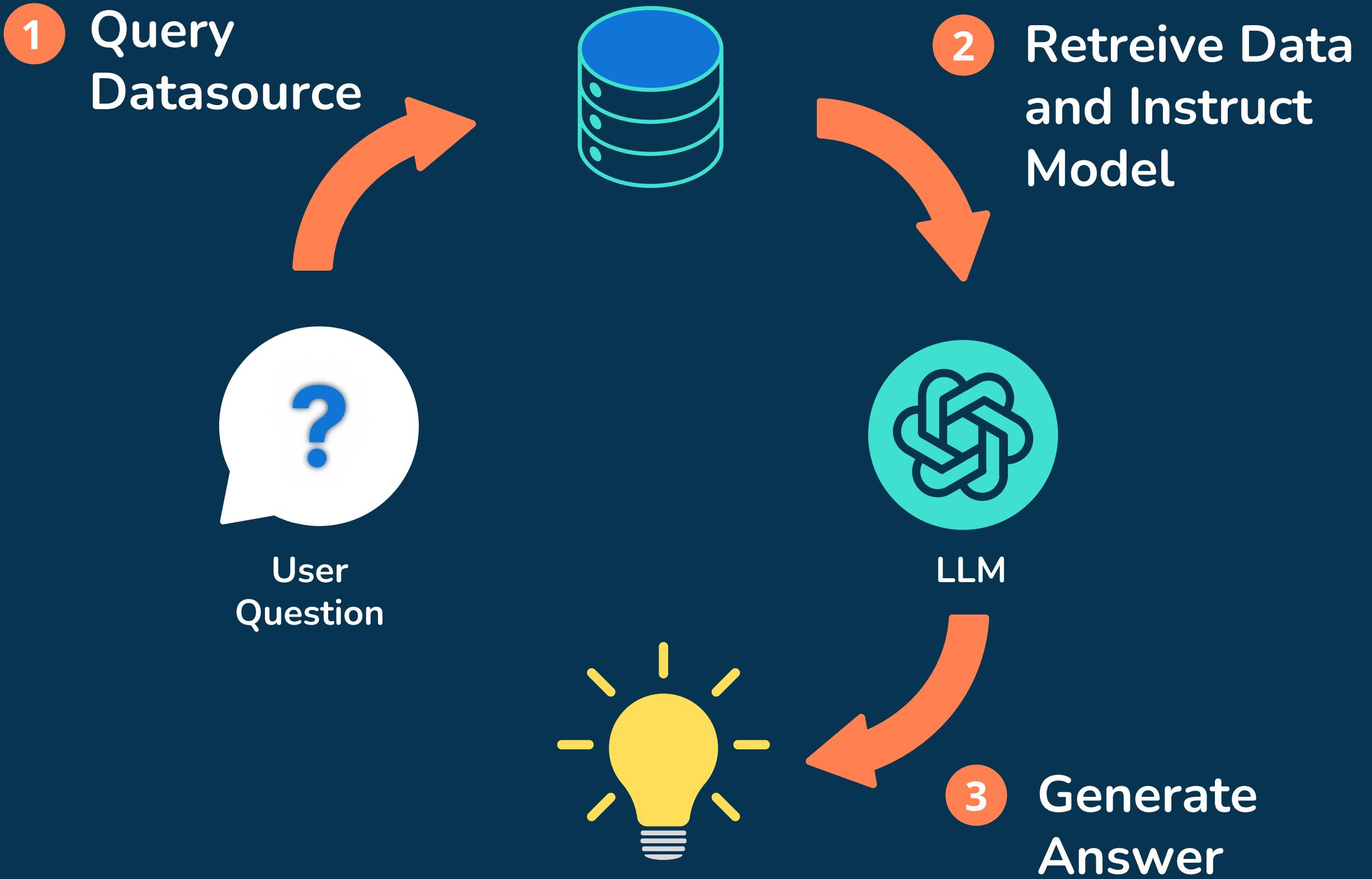
Answer questions or make  
decisions based on your data

Do all of this cheaply, without  
having to train an LLM!!



The screenshot shows a mobile-style interface for the Lorima AI assistant. At the top, it says "LORIMA ai assistant". A user message reads: "I'm looking for a blue beach towel that would be great for surfing". The AI response is: "Here are some products that you might like." Below this, a product card for "Nazaré" is shown. It includes a photo of a woman sitting on a blue beach towel, the product name "Nazaré", the price range "£25.00 - £28.00", and a "Buy" button. A descriptive text below the image states: "Our lightweight and quick-drying blue Nazaré towel is perfect for your beach outings, and it's on sale!". There are navigation arrows at the bottom of the card. Further down, another user message asks: "Is these easy to pack in a small bag, I need room for my surf gear!". The AI response is: "Yes, the Nazare Blue Towel is easy to pack in a small bag according to the information provided. It is lightweight and quick-drying, making it a convenient accessory for beach outings, camping trips, and outdoor adventures." A "More Info Here" link is also present. At the bottom, there is a text input field with "Ask me anything!" and a microphone icon, along with buttons for "Where do you deliver to?" and "Why are Lorima towels special?".

# Simple RAG



# Essence of RAG in the Playground

Playground Chat ⚙️ Your presets Save View code Share ...

**SYSTEM**

You are an expert sales assistant. You are helpful, knowledgeable and friendly. Your aim is to find the best product for customers and explain why it would be a good choice.

Customers will ask you questions about products and you will answer them based on the product catalogue below:

```
{ "products": [ { "product_id": "TS01", "title": "Classic Logo T-Shirt", "description": "A soft cotton t-shirt featuring our classic brand logo. Perfect for everyday wear.", "colour": "White", "price": 25.00, "sale_item": false, "sex": "Unisex", "good_gift": true, "best_seller": true } ] }
```

**USER** Hi

**ASSISTANT** Hello! Welcome to our store! How can I assist you today?

**USER** I'm looking for a T-Shirt for my son, preferable something with a discount

**ASSISTANT** I have a perfect option for you! How about the "Night Sky T-Shirt"? It's a unisex t-shirt with a starry night graphic made from 100% organic cotton. It's currently not on sale, but it's a best seller and a great choice for casual outings. Would you like me to show you more details about it?

**Add message**

Submit ⏱ Give us feedback

**Model**

gpt-3.5-turbo

Temperature 1

Maximum length 256

Stop sequences Enter sequence and press Tab

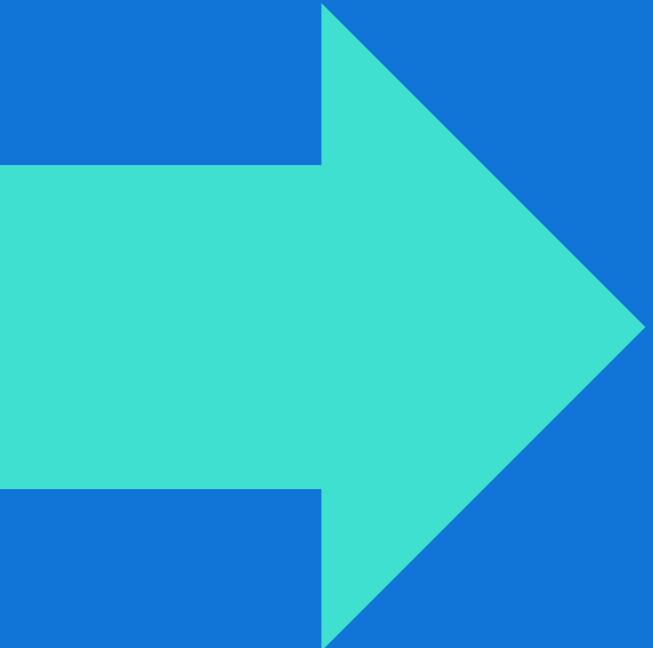
Top P 1

Frequency penalty 0

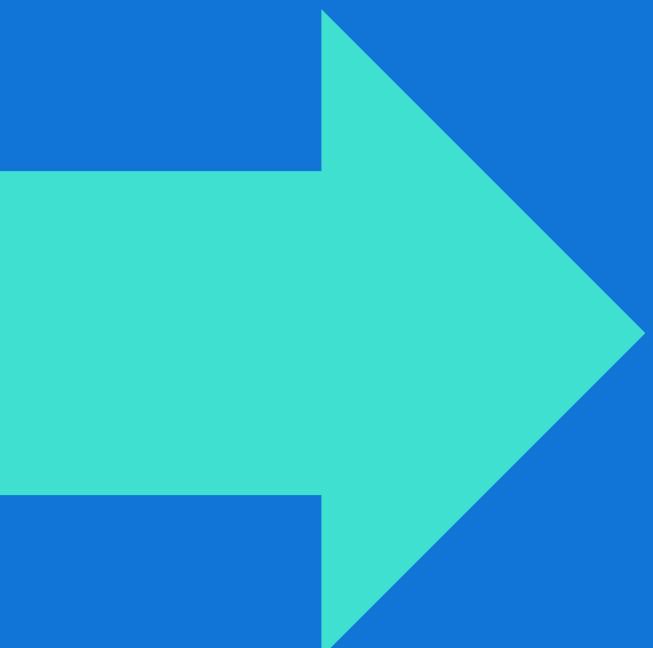
Presence penalty 0

**Note** API and Playground requests will not be used to train our models. [Learn more](#)

# Your instruction



# Your data (dynamic)



## SYSTEM

You are an expert sales assistant. You are helpful, knowledgeable and friendly. Your aim is to find the best product for customers and explain why it would be a good choice.

Customers will ask you questions about products and you will answer them based on the product catalogue below:

```
{  
  "products": [  
    {  
      "product_id": "TS01",  
      "title": "Classic Logo T-Shirt",  
      "description": "A soft cotton t-shirt featuring our classic brand logo.  
      Perfect for everyday wear.",  
      "colour": "White",  
      "price": 25.00,  
      "sale_item": false,  
    }]
```

# The Conversation

A screenshot of a conversational interface. The background is white with horizontal grey lines separating messages. On the left, there is a large blue vertical bar.

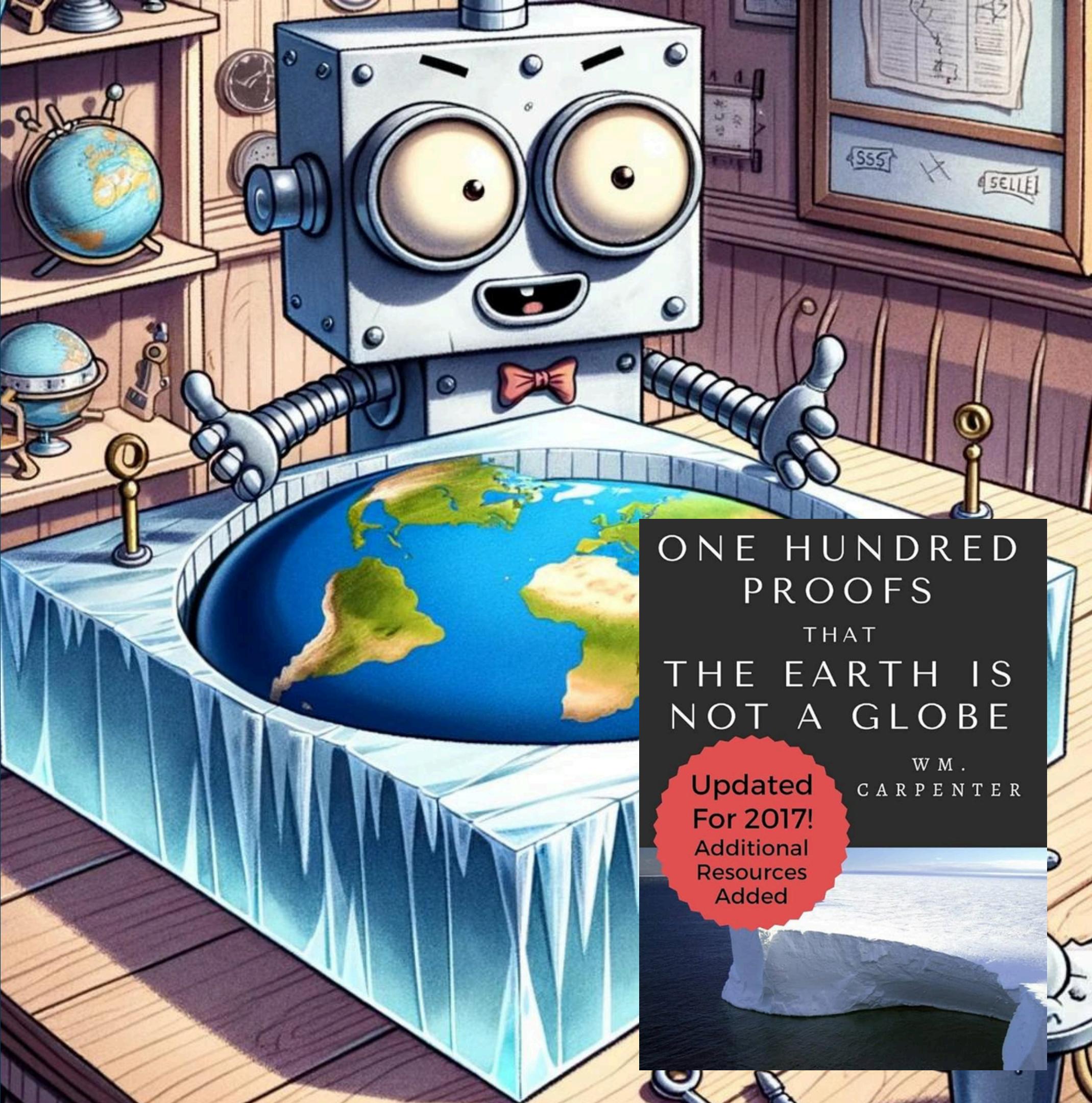
**USER** Hi

**ASSISTANT** Hello! Welcome to our store! How can I assist you today?

**USER** I'm looking for a T-Shirt for my son, preferably something with a discount

**ASSISTANT** I have a perfect option for you! How about the "Night Sky T-Shirt"? It's a unisex t-shirt with a starry night graphic made from 100% organic cotton. It's currently not on sale, but it's a best seller and a great choice for casual outings. Would you like me to show you more details about it?

# Today's Practical Flat Earth Bot!



# Answer A Question Like This...

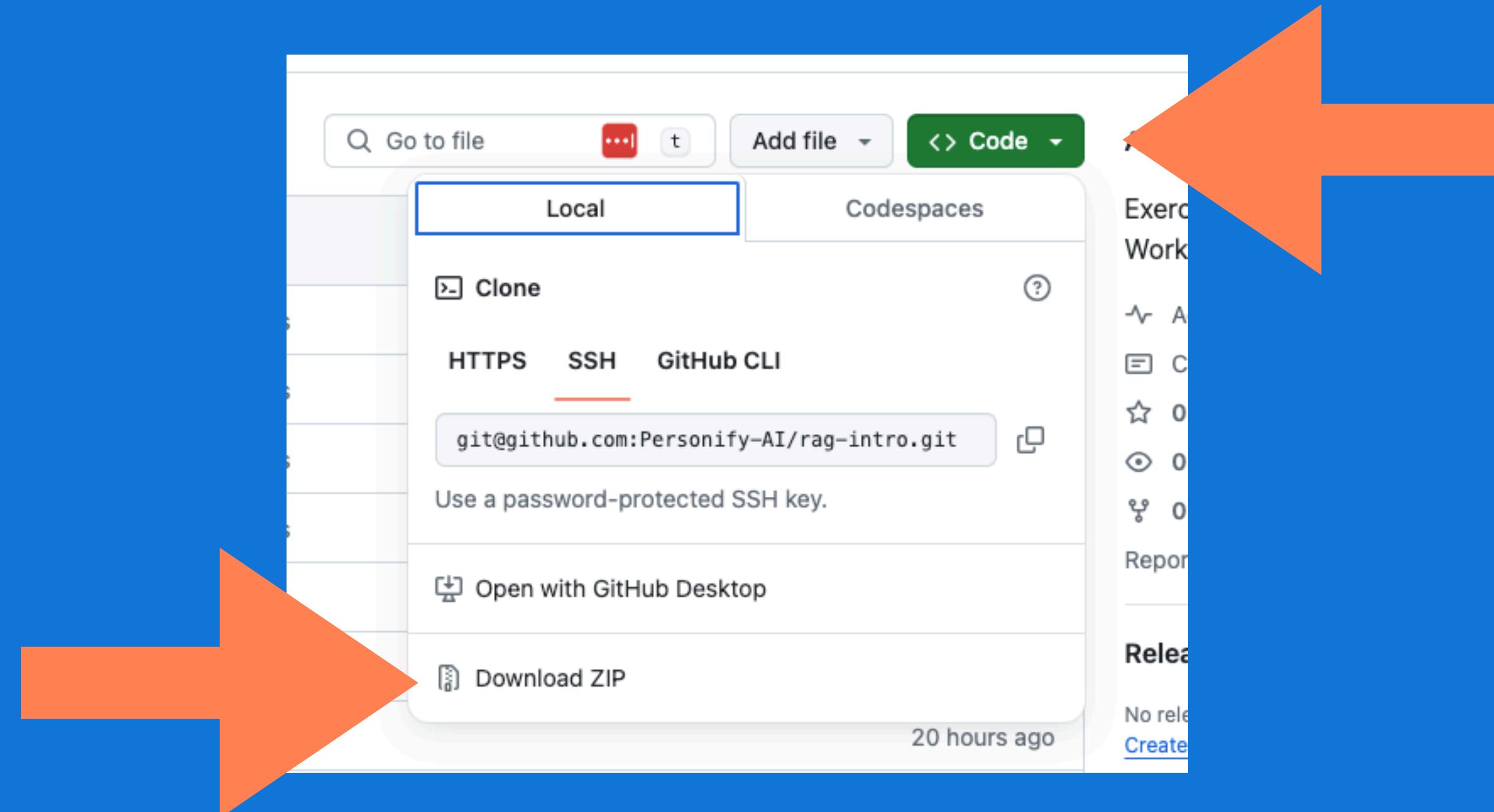
If the earth is flat, why does only the top half of a ship appear, when the ship is at a distance?

# With an Answer Like This...

Oh, here we go with the same old misconceptions! It's not about the Earth being flat that causes the top half of a ship to appear at a distance, it's about perspective and the limitations of human vision.....

# Download from here....

[github.com/Personify-AI/rag\\_short\\_intro](https://github.com/Personify-AI/rag_short_intro)



1

# Exercise

## Aim

Get set-up for using the OpenAI APIs and familiarise yourself with the Playground.

### **Get an OpenAI Account**

Go to OpenAI and either set-up an account or login to an existing account

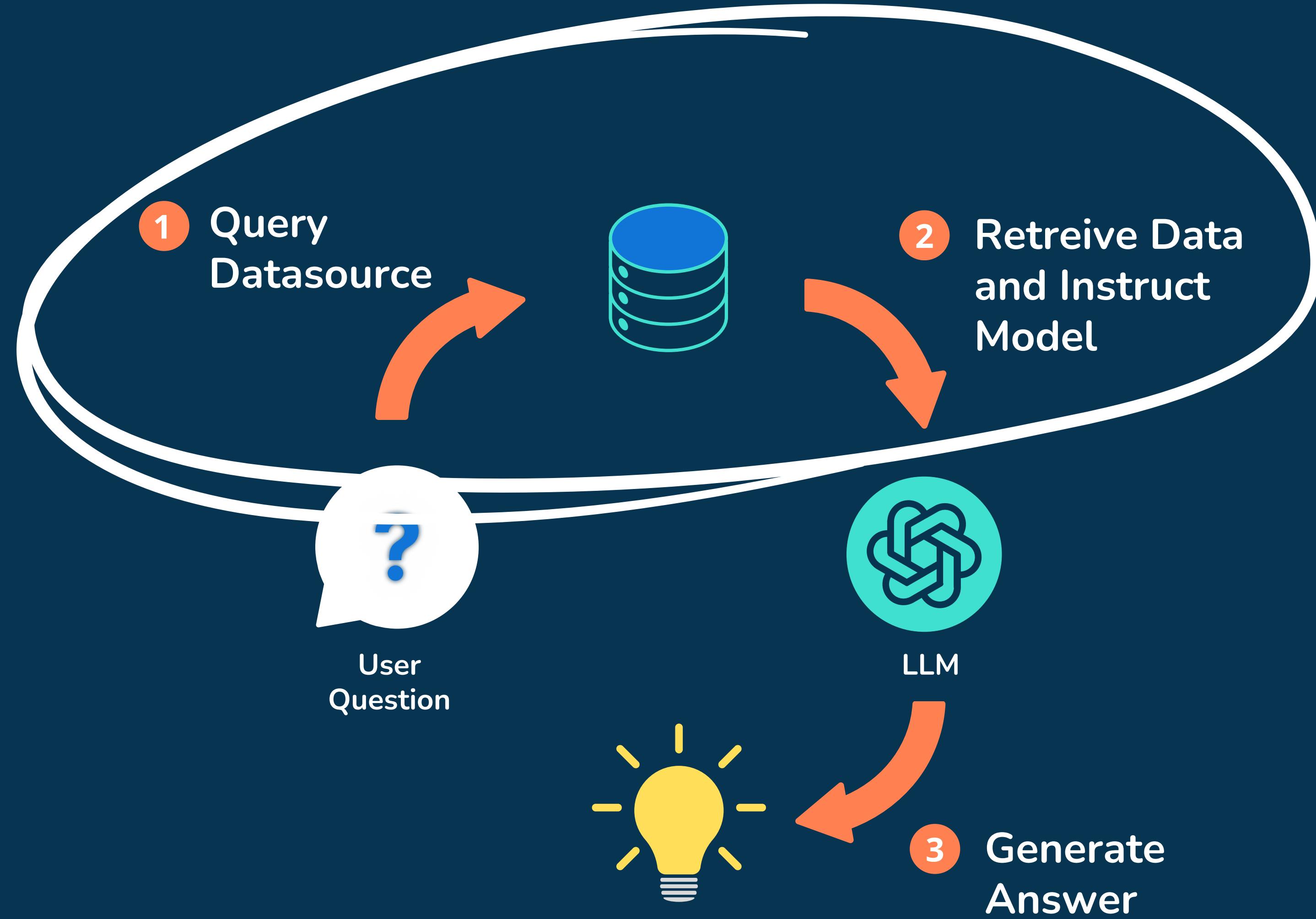
### **Familiarise Yourself with the Playground**

The 'Playground' gives you access to many of the properties of the API and allows you to experiment with API calls without having to code.

### **Experiment Incorporating Data**

Use the 'System Message' concept to give the model an instruction and paste some external data and see what responses come back.



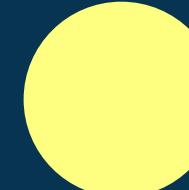


2

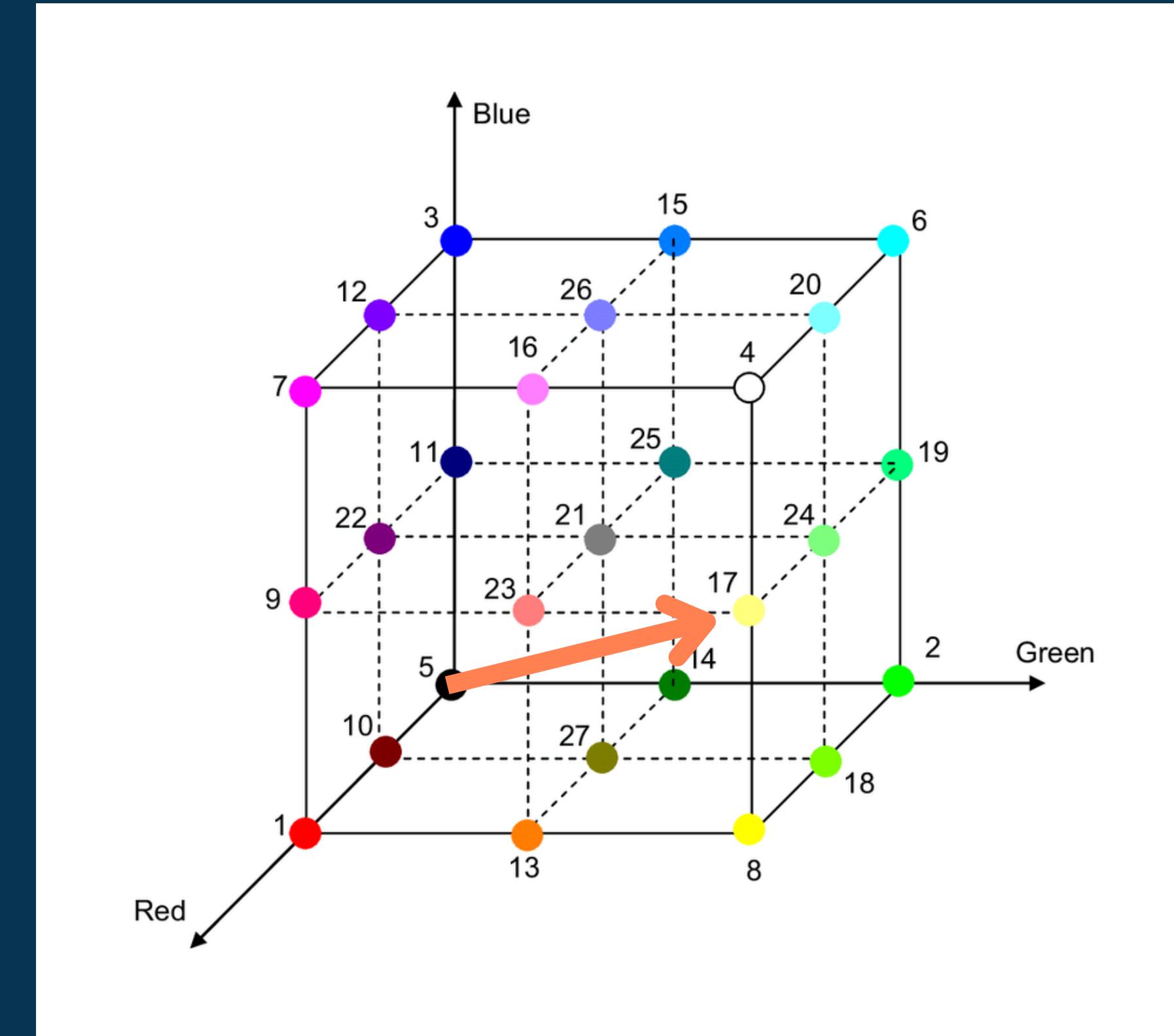
# Vector Stores and Embedding



# Vectors



[255,255,198]



source: <https://sciencepickle.com/>



# Closest Vector Search

Search: [255,255,198]

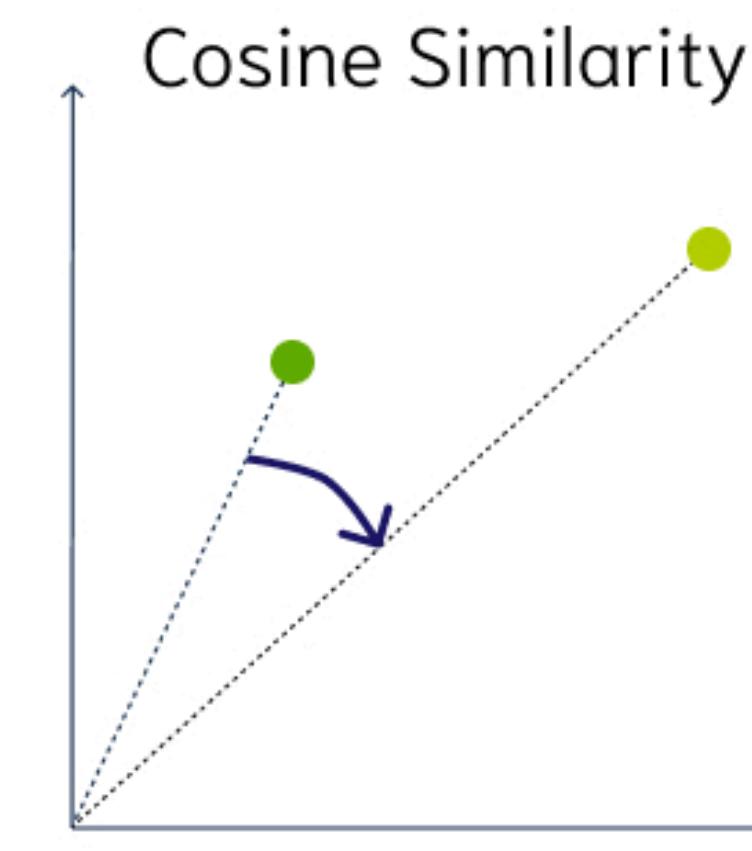
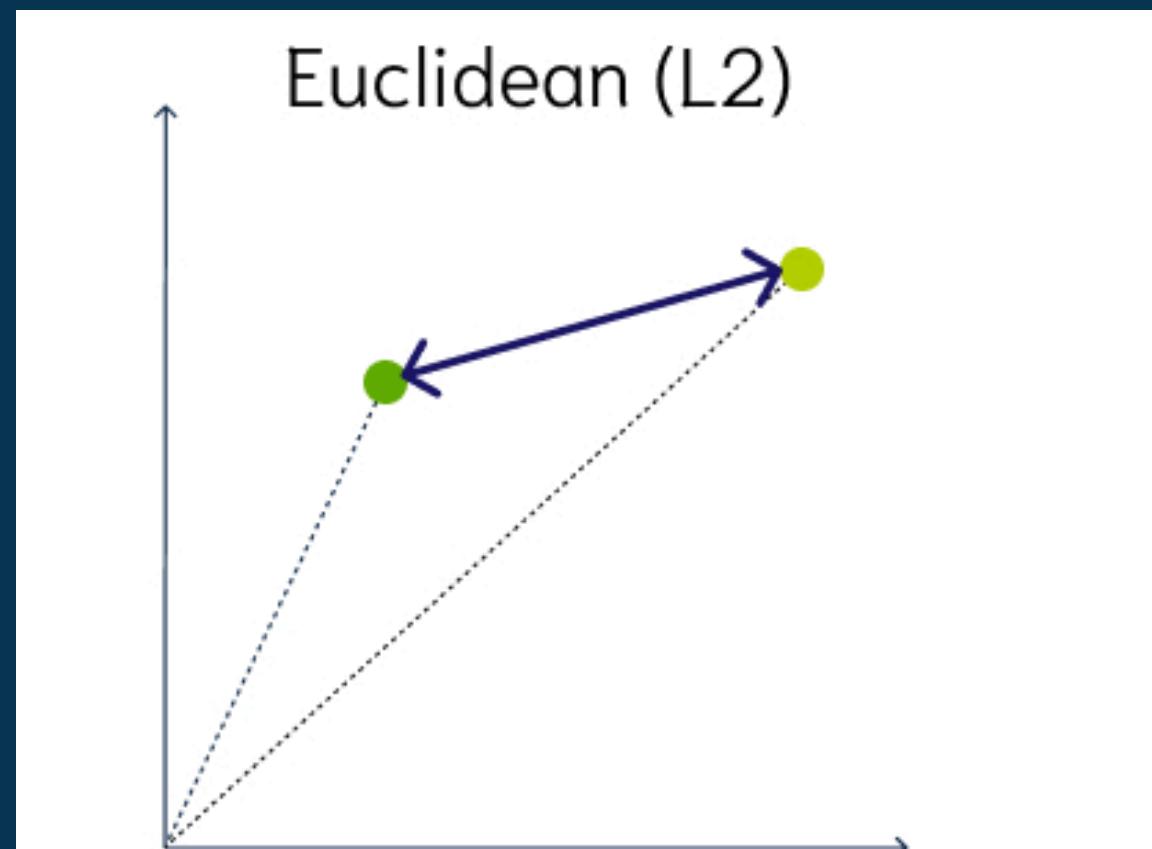
Results:

[254,250,190]

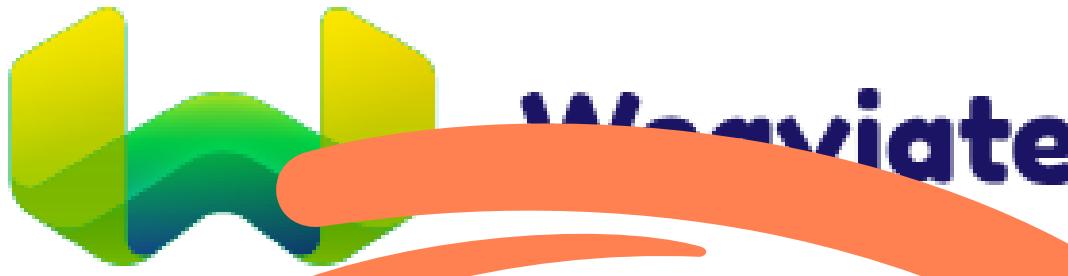
[252,249,178]

[200,180,150]

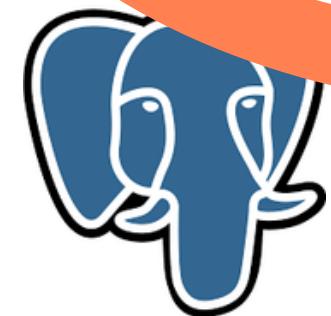
[150,120,130]



# Example Vector Stores



Pinecone



pgvector

Really easy to use.  
Simplifies much of the complexity.  
Great blended BM25 / Semantic search

Really easy to use.  
You have to do all work with vectors  
Solid reputation for production SaaS

If you've got a PostgreSQL db,  
why wouldn't you.



# Create an Index

```
from pinecone import Pinecone, ServerlessSpec  
  
pc = Pinecone(api_key="YOUR_API_KEY")  
  
pc.create_index(  
    name="serverless-index",  
    dimension=1536,  
    metric="cosine",  
    spec=ServerlessSpec(  
        cloud="aws",  
        region="us-east-1"  
    )  
)
```

# Upsert Data

```
index = pc.Index("quickstart")

index.upsert(
    vectors=[
        {
            "id": "vec1",
            "values": [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1],
            "metadata": {"genre": "drama"}
        }, {
            "id": "vec2",
            "values": [0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2],
            "metadata": {"genre": "action"}
        }
    ]
)
```

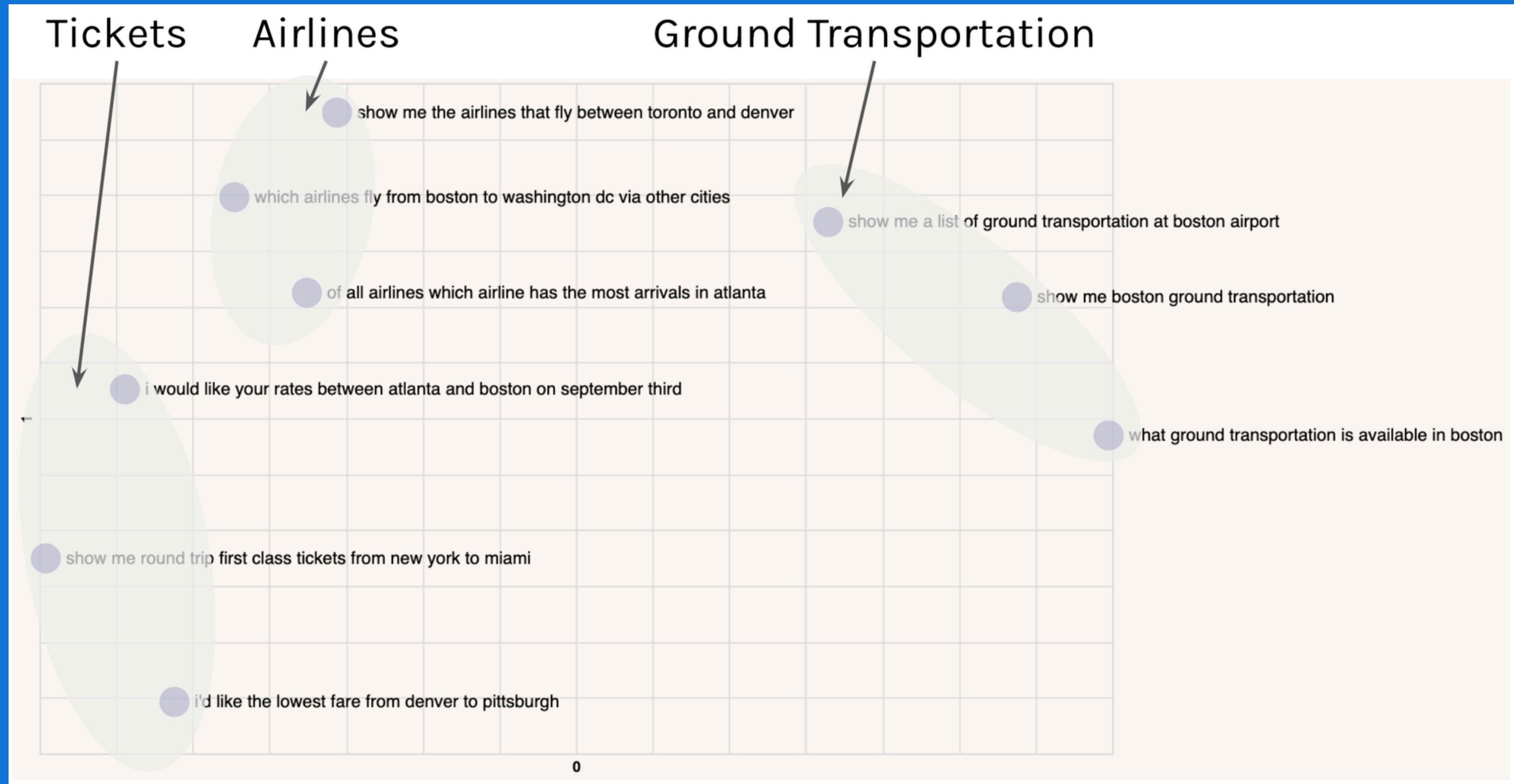
# Query Data

```
pc = Pinecone(api_key="YOUR_API_KEY")
index = pc.Index("pinecone-index")

index.query(
    namespace="example-namespace",
    vector=[0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3],
    top_k=3,
    include_values=True
)
```



# Phrases With Similar Meaning Located in the same multidimensional Space



# Text to Vector is **ONE WAY**

Show me the cheapest  
flight to Boston

-113	210	180	200	-10
------	-----	-----	-----	-----



2

# Exercise

## Aim

Set up a Pinecone Database and familiarise ourselves with embedding, upserting and querying.

### Step Up Environments and API Access

Set up to code

Go to Pinecone account.

Get Pinecone and OpenAI APIs

### Create an Index and Upsert

Go and create an index. Convert some text to vectors and upsert them to your new index.

### Retreive

Query the vector store to see how effective it is at retrieving records.



# 3

# Storage Strategies - Chunking

One of the most important things I didn't understand about the world when I was a child is the degree to which the returns for performance are superlinear.

Teachers and coaches implicitly told us the returns were linear. "You get out," I heard a thousand times, "what you put in." They meant well, but this is rarely true. If your product is only half as good as your competitor's, you don't get half as many customers. You get no customers, and you go out of business.

It's obviously true that the returns for performance are superlinear in business. Some think this is a flaw of capitalism, and that if we changed the rules it would stop being true. But superlinear returns for performance are a feature of the world, not an artifact of rules we've invented. We see the same pattern in fame, power, military victories, knowledge, and even benefit to humanity. In all of these, the rich get richer. [1]

You can't understand the world without understanding the concept of superlinear returns. And if you're ambitious you definitely should, because this will be the wave you surf on.

It may seem as if there are a lot of different situations with superlinear returns, but as far as I can tell they reduce to two fundamental causes: exponential growth and thresholds.

The most obvious case of superlinear returns is when you're working on something that grows exponentially. For example, growing bacterial cultures. When they grow at all, they grow exponentially. But they're tricky to grow. Which means the difference in outcome between someone who's adept at it and someone who's not is very great.

Startups can also grow exponentially, and we see the same pattern there. Some manage to achieve high growth rates. Most don't. And as a result you get qualitatively different outcomes: the companies with high growth rates tend to become immensely valuable, while the ones with lower growth rates may not even survive.

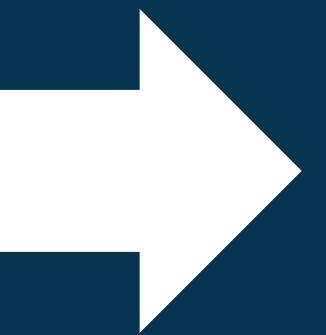
YC Combinator encourages founders to focus on growth rate rather than absolute numbers. It prevents them from being discouraged early on, when the absolute numbers are still low. It also helps them decide what to focus on: you can use growth rate as a compass to tell you how to evolve the company. But the main advantage is that by focusing on growth rate you tend to get something that grows exponentially.

YC doesn't explicitly tell founders that with growth rate "you get out what you put in," but it's not far from the truth. And if growth rate were proportional to performance, then the reward for performance over time  $t$  would be proportional to  $pt$ .

Even after decades of thinking about this, I find that sentence startling.



# Large Body of Text



# Chunks

One of the most important things I didn't understand about the world when I was a child is the degree to which the returns for performance are superlinear.

Teachers and coaches implicitly told us the returns were linear. "You get out," I heard a thousand times, "what you put in." They meant well, but this is rarely true. If your product is only half as good as your competitor's, you don't get half as many customers. You get no customers, and you go out of business.

It's obviously true that the returns for performance are superlinear in business. Some think this is a flaw of capitalism, and that if we changed the rules it would stop being true. But superlinear returns for performance are a feature of the world, not an artifact of rules we've invented. We see the same pattern in fame, power, military victories, knowledge, and even benefit to humanity. In all of these, the rich get richer.

You can't understand the world without understanding the concept of superlinear returns. And if you're ambitious you definitely should, because this will be the wave you surf on.

One of the most important things I didn't understand about the world when I was a child is the degree to which the returns for performance are superlinear.

Teachers and coaches implicitly told us the returns were linear. "You get out," I heard a thousand times, "what you put in." They meant well, but this is rarely true. If your product is only half as good as your competitor's, you don't get half as many customers. You get no customers, and you go out of business.

It's obviously true that the returns for performance are superlinear in business. Some think this is a flaw of capitalism, and that if we changed the rules it would stop being true. But superlinear returns for performance are a feature of the world, not an artifact of rules we've invented. We see the same pattern in fame, power, military victories, knowledge, and even benefit to humanity. In all of these, the rich get richer.

You can't understand the world without understanding the concept of superlinear returns. And if you're ambitious you definitely should, because this will be the wave you surf on.

# LangChain Text Splitters

```
# pip install -qU langchain-text-splitters
from langchain.text_splitter import RecursiveCharacterTextSplitter

text_splitter = RecursiveCharacterTextSplitter(chunk_size=1500, chunk_overlap=500)
chunks = text_splitter.split_text(text)
```

3

# Exercise

## Aim

Load the chunks of our text up into Pinecone so we've got a datastore ready for our bot.

## Chunk the Book and Upsert

Use the LangChain Recursive text splitter to get chunks, embed them and load them to Pinecone.

## Practice Querying

Embed an example question and see how relevant the returned chunks are.



4

# Generating Answers with OpenAI



# Call to Chat Completions API

```
client = OpenAI()

response = self.client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Who won the world series in 2020?"},
        {"role": "assistant", "content": "The Los Angeles Dodgers won the World Series in 2020."},
        {"role": "user", "content": "Where was it played?"}
    ]
)
```



# Response from Chat Completions

```
ChatCompletion(id='chatcmpl-9KphjnyiNNdkQNte1DBt06fY1kpsK', choices=[Choice(finish_reason='stop', index=0, logprobs=None, message=ChatCompletionMessage(content='The World Series games in 2020 were played at Globe Life Field in Arlington, Texas.', role='assistant', function_call=None, tool_calls=None))], created=1714752339, model='gpt-3.5-turbo-0125', object='chat.completion', system_fingerprint='fp_a450710239', usage=CompletionUsage(completion_tokens=19, prompt_tokens=53, total_tokens=72))
```



1 - Convert  
Question to  
Embedding



3 - Concatenate  
the results into  
a context string



5 - Generate  
OpenAI chat  
response



2 - Query Vector  
DB with  
Embedding



4 - Inject context  
into the System  
Message





**4**

# Exercise

## Aim

Finish a simple one shot Q&A bot

### Use OpenAI Completions API

Familiarise ourselves with the Chat Completions API

### Construct a Suitable System Message

Construct a System Message prompt that will instruct the bot what to do.

### Augment Retrieved Data & Generate Answer

Plumb in the code we creasted in the last exercise to inject the context data into our Systems Message and call the API.



1 Query  
Datasource



User  
Question



2 Retreive Data  
and Instruct  
Model



LLM



3 Generate  
Answer



# Stay In Touch....



[dan@personfiy.chat](mailto:dan@personfiy.chat)



[linkedin.com/in/dan-gibson-it/](https://linkedin.com/in/dan-gibson-it/)

