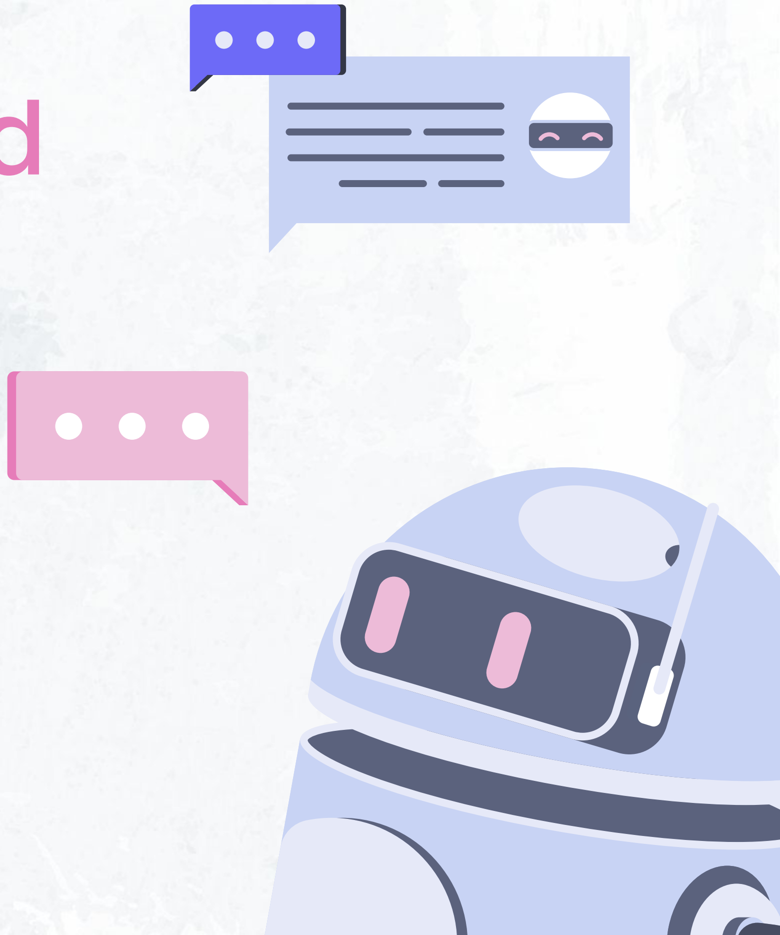
Building Fine-Tuned Chatbots with No-code



FINE-TUNER.AI

Agenda

01 → Fine-Tuner.ai (2 min)

02 → Ingestion (20 min)

03 → Retrieval & Generation (20 min)

04 → Integration (20 min)

05 → Interactive Q&A (20 min)

FINE-TUNER.AI

01 →

Fine-Tuner.ai

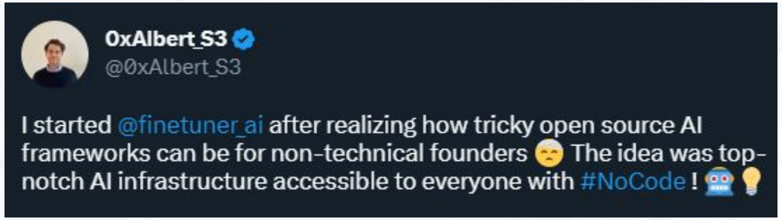
FINE-TUNER.AI

What’s Fine-Tuner.ai?

The ultimate no-code solution for building and deploying customized AI agents.

Makes it effortless for both **technical and non-technical builders** to leverage the latest AI capabilities.

- 🧠 Pre-built **customizable AI Agents**
- 🌐 Model Fine-Tuning and Text, Image Embeddings
- 🚀 No Coding Required
- 🔌 Deploy via **API Endpoints, Widgets or Plugins**
- 🛡️ Unlimited Secure Vector Data Storage on Pinecone



0xAlbert_S3
@0xAlbert_S3

I started @finetuner_ai after realizing how tricky open source AI frameworks can be for non-technical founders 😒 The idea was top-notch AI infrastructure accessible to everyone with #NoCode ! 🤖💡

FINE-TUNER.AI

High Level

At a high level, there are two components to setting up LLM over your own data:

(1) ingestion of the data

(2) chatbot over the data

FINE-TUNER.AI

02 →

Ingestion

FINE-TUNER.AI

Data Ingestion - Under the Hood

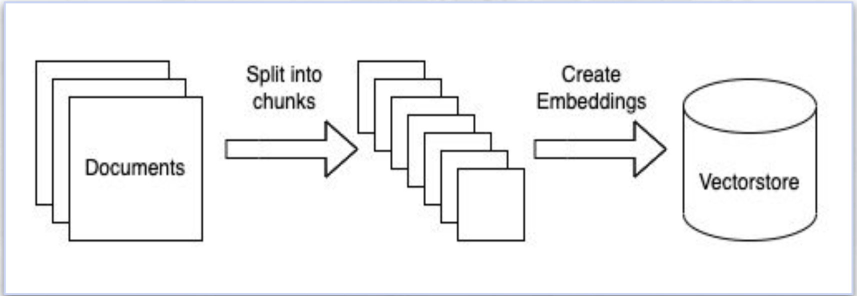
This can be broken in a few sub steps. Each step is highly modular and would require Python or JS knowledge combined with tech architecture knowledge to set them up. The steps are:

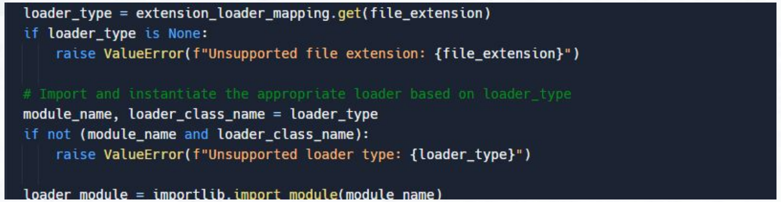
(a) Load data sources to text
this involves loading your data from arbitrary sources to text in a form that it can be used downstream.

(b) Chunk text
this involves chunking the loaded text into smaller chunks. This is necessary because language models generally have a limit to the amount of text they can deal with

(c) Embed text
this involves creating a numerical embedding for each chunk of text.

(d) Load embeddings to Vector DB
this involves putting embeddings and documents into a Vector DB. Vecstorstores help us find the most similar chunks in the embedding space quickly and efficiently.





```
loader_type = extension_loader_mapping.get(file_extension)
if loader_type is None:
    raise ValueError(f"Unsupported file extension: {file_extension}")

# Import and instantiate the appropriate loader based on loader_type
module_name, loader_class_name = loader_type
if not (module_name and loader_class_name):
    raise ValueError(f"Unsupported loader type: {loader_type}")

loader_module = importlib.import_module(module_name)
```

FINE-TUNER.AI

Data Ingestion - Using Fine-Tuner

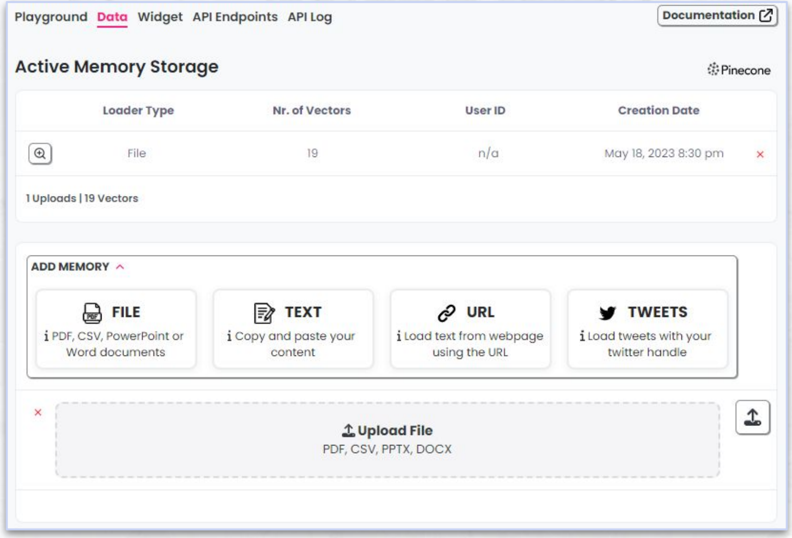
Provides visual no-code interface to store & manage vector data - can be segmented and hosted for multiple users with their specific files:

(a) Load data sources to text
this involves loading your data from arbitrary sources to text in a form that it can be used downstream.

(b) Chunk text
this involves chunking the loaded text into smaller chunks. This is necessary because language models generally have a limit to the amount of text they can deal with

(c) Embed text
this involves creating a numerical embedding for each chunk of text.

(d) Load embeddings to Vector DB
this involves putting embeddings and documents into a Vector DB. Vecstorstores help us find the most similar chunks in the embedding space quickly and efficiently.



Active Memory Storage

Loader Type	Nr. of Vectors	User ID	Creation Date
File	19	n/a	May 18, 2023 8:30 pm

1 Uploads | 19 Vectors

ADD MEMORY

FILE: PDF, CSV, PowerPoint or Word documents

TEXT: Copy and paste your content

URL: Load text from webpage using the URL

TWEETS: Load tweets with your twitter handle

Upload File
PDF, CSV, PPTX, DOCX

FINE-TUNER.AI

03 →

Retrieval & Generation

FINE-TUNER.AI

Retrieval & Generation - Under the Hood

LLMs do not know about information they were not trained on. If you want to use an LLM to answer questions about documents it was not trained on, you have to give it information about those documents. The most common way to do this is through "retrieval augmented generation".

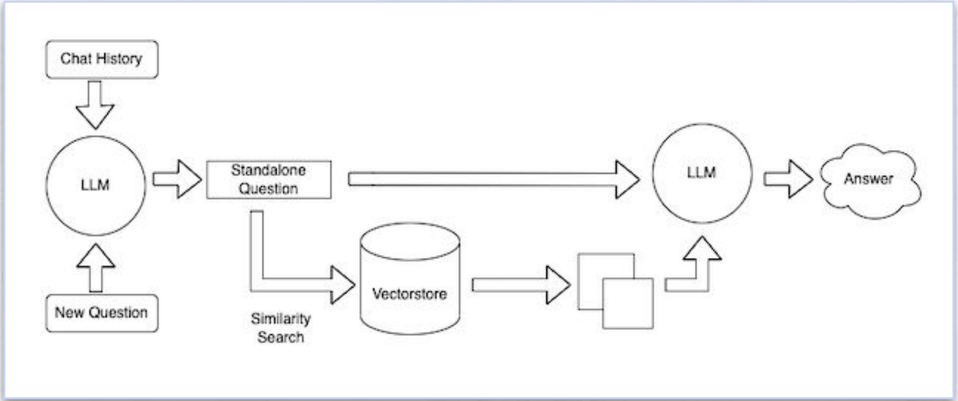
(a) Receive user question

(b) Lookup documents in the index relevant to the question (similarity search)

(c) Construct a PromptValue from the question and relevant documents

(d) Pass the PromptValue to LLM

(e) Get back the result and return to the user.

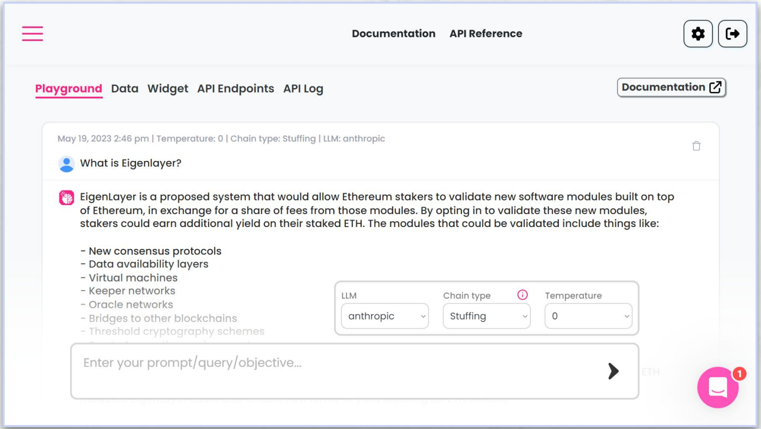


FINE-TUNER.AI

Retrieval & Generation - Using Fine-Tuner

Fine-Tuner wraps the Retrieval & Generation chain into a simple workflow with a visual interface:

- **Customizable** - Parameters such as LLM, Chain type, Prompt Template (coming soon 👀)
- **Visual playground** for setting up and testing
- **Can be easily integrated** via ready API Endpoints, Widget, Bubble plugin.
- **Visual log** with API calls (for each user)



May 19, 2023 2:46 pm | Temperature: 0 | Chain type: Stuffing | LLM: anthropic

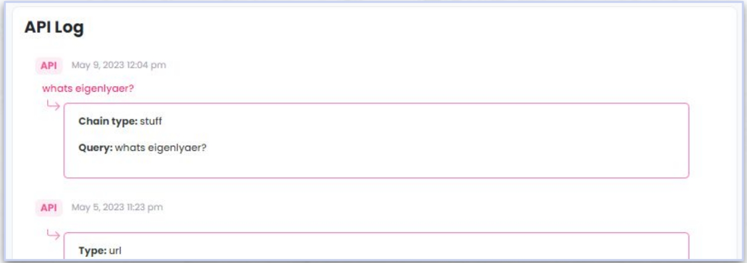
What is Eigenlayer?

EigenLayer is a proposed system that would allow Ethereum stakers to validate new software modules built on top of Ethereum, in exchange for a share of fees from those modules. By opting in to validate these new modules, stakers could earn additional yield on their staked ETH. The modules that could be validated include things like:

- New consensus protocols
- Data availability layers
- Virtual machines
- Keeper networks
- Oracle networks
- Bridges to other blockchains
- Threshold cryptography schemes

LLM: anthropic
Chain type: Stuffing
Temperature: 0

Enter your prompt/query/objective...



API Log

API May 9, 2023 12:04 pm

whats eigenlyaer?

Chain type: stuff

Query: whats eigenlyaer?

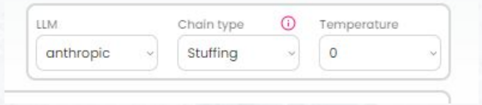
API May 5, 2023 11:23 pm

Type: url

FINE-TUNER.AI

Type of LLM chains

Different chains, used for interacting with indexes by combining user data with LLMs. The choice of method being context-specific and ranging from simplest to most complex.


LLM: anthropic
Chain type: Stuffing
Temperature: 0

(a) **Stuffing** Involves inputting all relevant data into a single prompt, giving the language model access to the entire context. However, this method is limited by the context length and becomes unfeasible for larger or multiple documents.

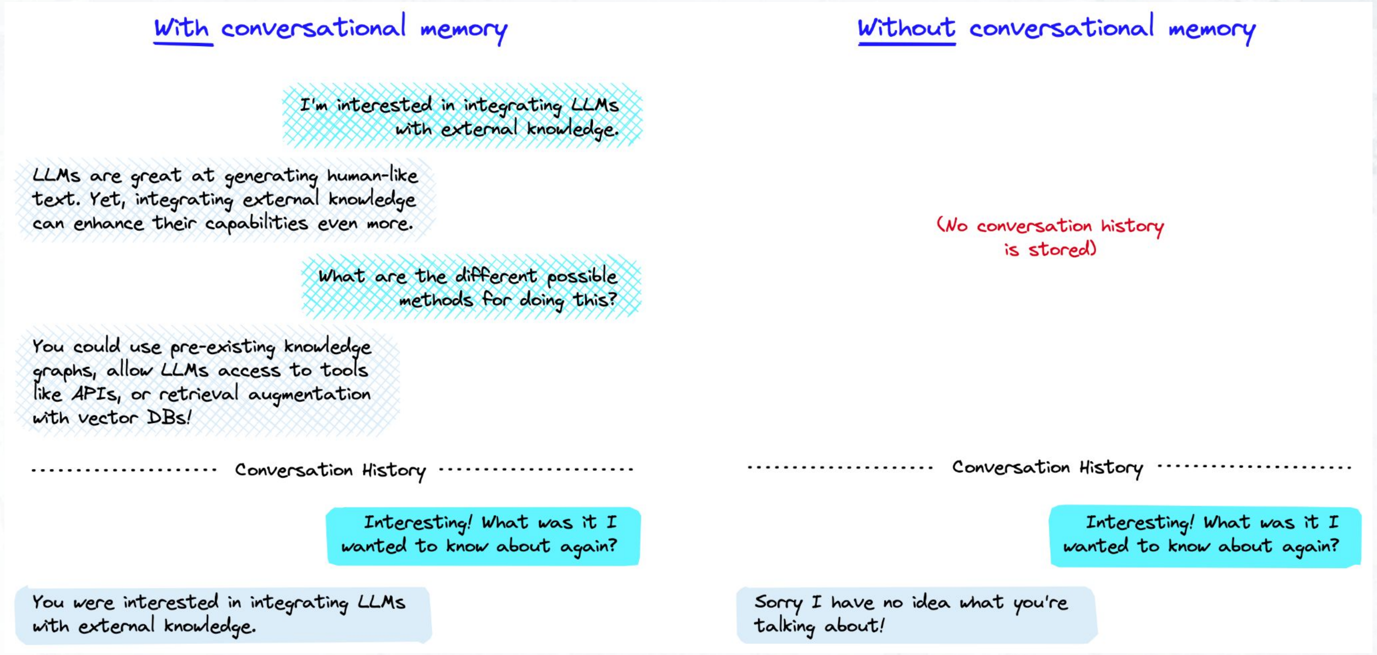
(b) **Map Reduce** Processes individual data chunks and combines their outputs, making it scalable for larger documents and allowing parallelization. The downside is the higher number of calls to the language model and potential loss of information during combination.

(c) **Refine** Iteratively refines output by processing one document at a time, potentially retaining more context. The drawback is the higher number of calls to the language model, and the inability to parallelize, as well as possible document ordering dependencies.

FINE-TUNER.AI

Conversational memory

The memory allows a Large Language Model (LLM) to remember previous interactions with the user.


With conversational memory
I'm interested in integrating LLMs with external knowledge.
LLMs are great at generating human-like text. Yet, integrating external knowledge can enhance their capabilities even more.
What are the different possible methods for doing this?
You could use pre-existing knowledge graphs, allow LLMs access to tools like APIs, or retrieval augmentation with vector DBs!
Conversation History
Interesting! What was it I wanted to know about again?
You were interested in integrating LLMs with external knowledge.
Without conversational memory
(No conversation history is stored)
Conversation History
Interesting! What was it I wanted to know about again?
Sorry I have no idea what you're talking about!

FINE-TUNER.AI

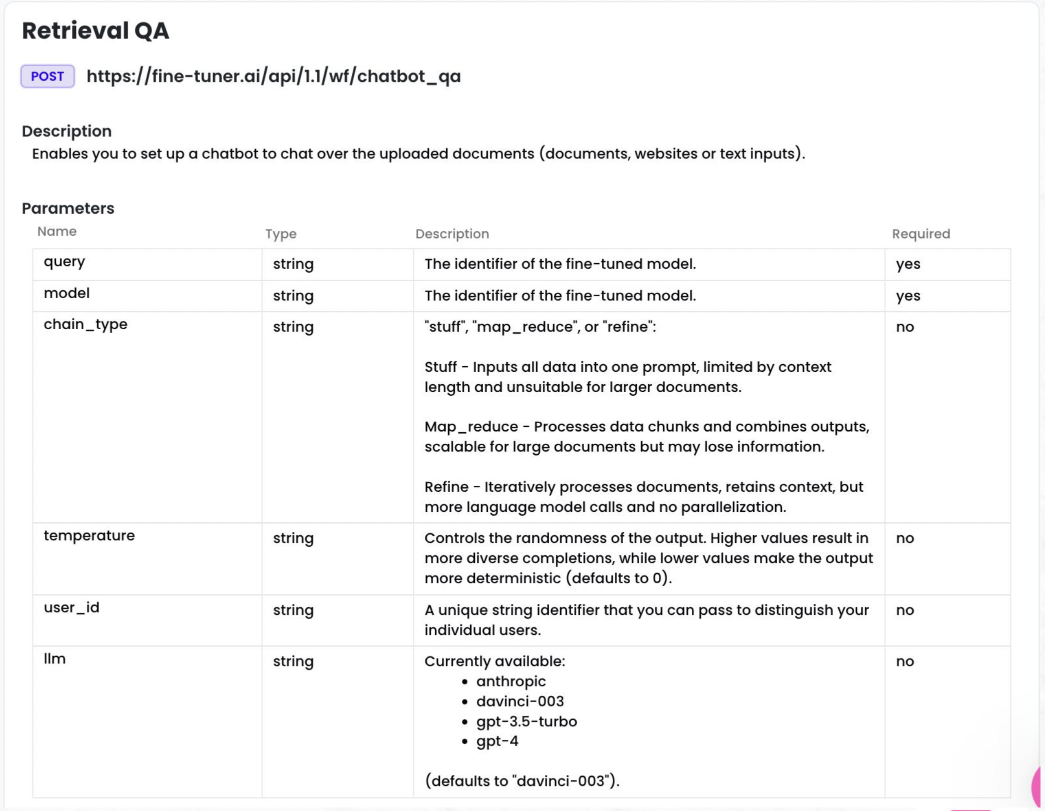
04 →

Integration

FINE-TUNER.AI

Using REST API

Enables you to set up a chatbot to chat over the uploaded documents (documents, websites or text inputs).



Retrieval QA

POST https://fine-tuner.ai/api/1.1/wf/chatbot_qa

Description

Enables you to set up a chatbot to chat over the uploaded documents (documents, websites or text inputs).

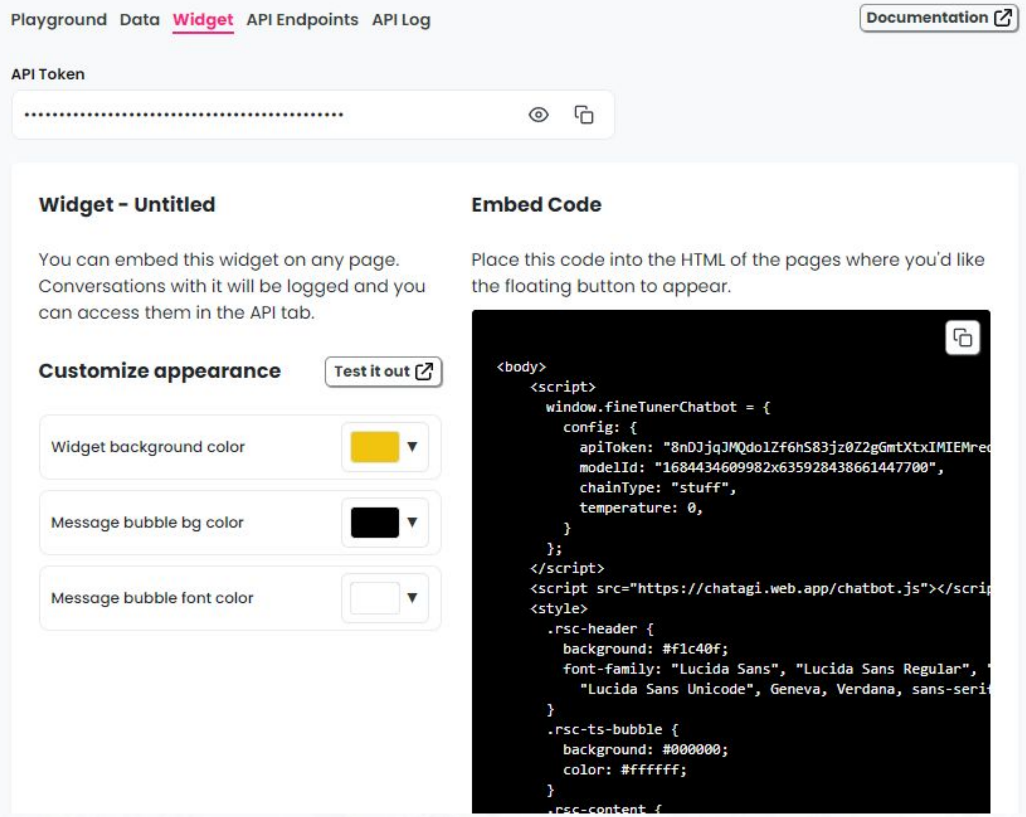
Parameters

Name	Type	Description	Required
query	string	The identifier of the fine-tuned model.	yes
model	string	The identifier of the fine-tuned model.	yes
chain_type	string	"stuff", "map_reduce", or "refine": Stuff - Inputs all data into one prompt, limited by context length and unsuitable for larger documents. Map_reduce - Processes data chunks and combines outputs, scalable for large documents but may lose information. Refine - Iteratively processes documents, retains context, but more language model calls and no parallelization.	no
temperature	string	Controls the randomness of the output. Higher values result in more diverse completions, while lower values make the output more deterministic (defaults to 0).	no
user_id	string	A unique string identifier that you can pass to distinguish your individual users.	no
llm	string	Currently available: • anthropic • davinci-003 • gpt-3.5-turbo • gpt-4 (defaults to "davinci-003").	no

FINE-TUNER.AI

Using Widget

You can embed this widget on any page. Conversations with it will be logged and you can access them in the API tab.



Widget - Untitled

You can embed this widget on any page. Conversations with it will be logged and you can access them in the API tab.

Customize appearance

Widget background color	🟨 ▼
Message bubble bg color	⬛ ▼
Message bubble font color	⬜ ▼

Embed Code

Place this code into the HTML of the pages where you'd like the floating button to appear.

```
<body>
    <script>
      window.fineTunerChatbot = {
        config: {
          apiToken: "8nDJjqJMQdolZf6hS83jz0Z2gGmtXtxIMIEMre
          modelId: "1684434609982x635928438661447700",
          chainType: "stuff",
          temperature: 0,
        }
      };
    </script>
    <script src="https://chatagi.web.app/chatbot.js"></scrip
    <style>
      .rsc-header {
        background: #f1c40f;
        font-family: "Lucida Sans", "Lucida Sans Regular",
          "Lucida Sans Unicode", Geneva, Verdana, sans-seri
      }
      .rsc-ts-bubble {
        background: #000000;
        color: #ffffff;
      }
      .rsc-content {
```

FINE-TUNER.AI

05 →

Interactive Q&A