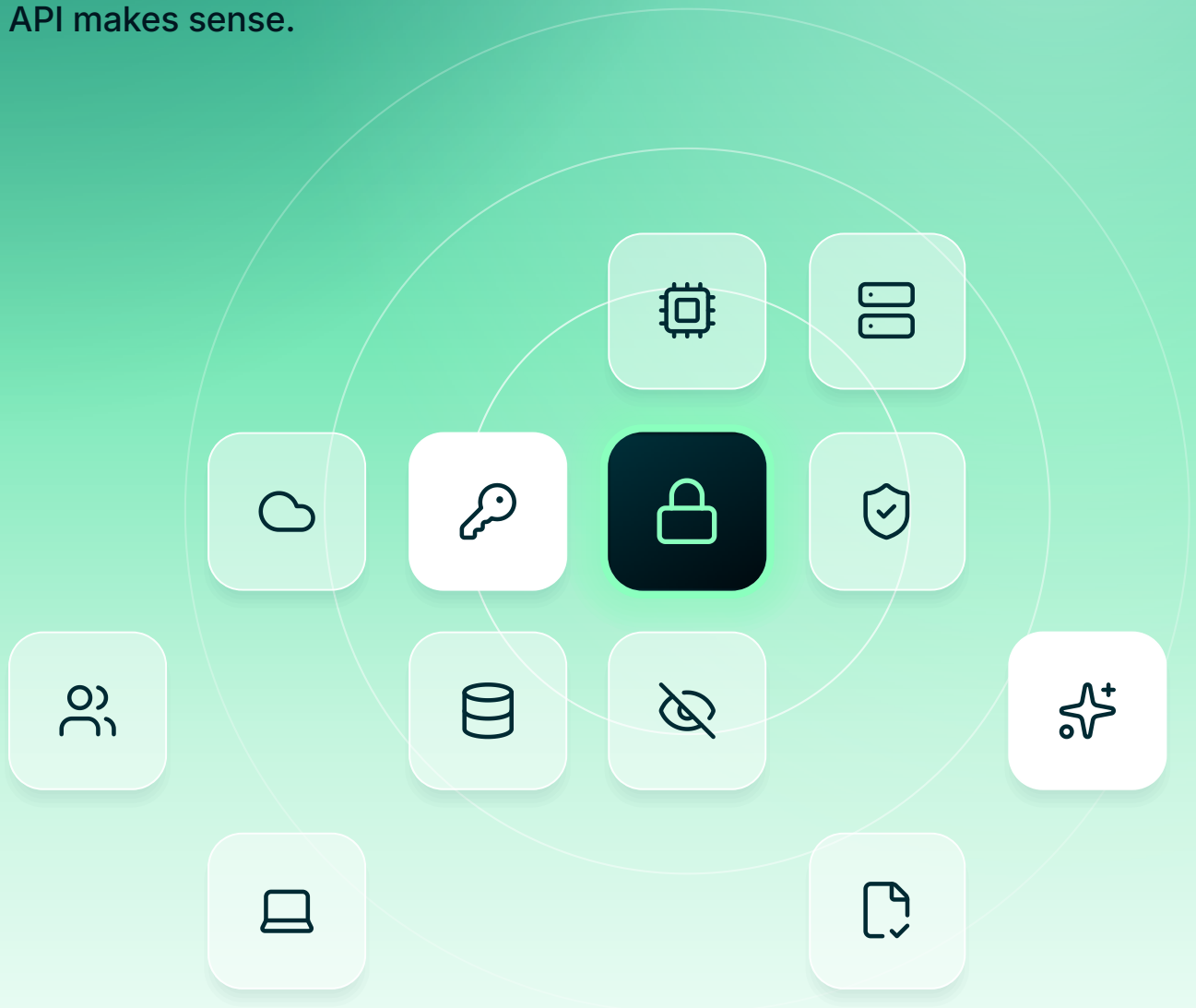
Privatemode AI

Where should your AI run?

An overview for financial decision makers and data protection officers on why adding a Confidential AI API makes sense.



Edgeless Systems | By Edgeless Systems, the makers of Privatemode

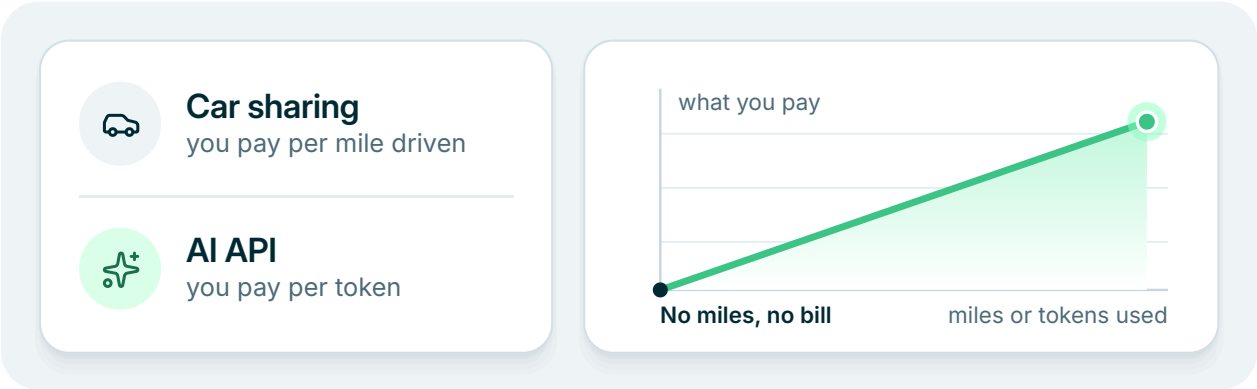
1

A confidential API belongs in every AI setup

There are many ways how you can run AI models. Depending on what you want to do, they all have advantages and disadvantages.

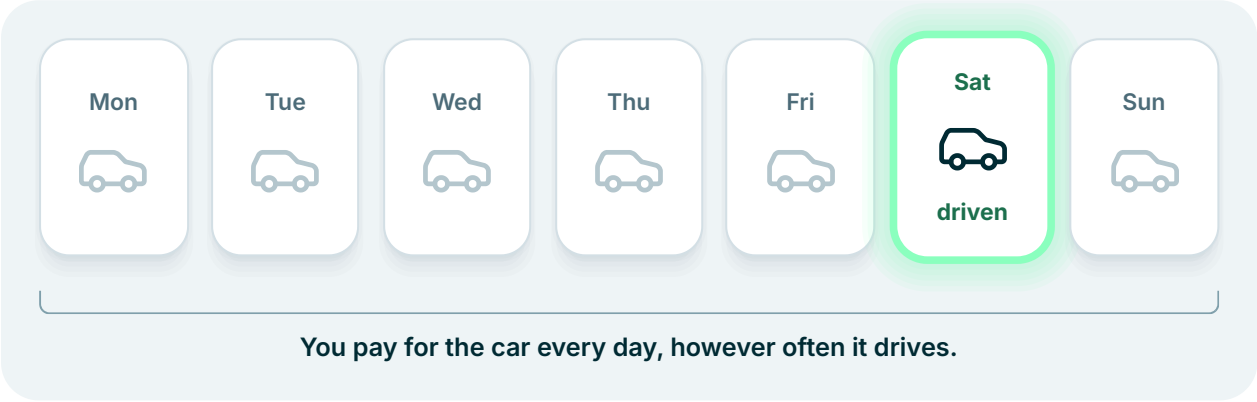
Most teams first get in touch with AI via a cloud provider that sells access to a model. Think of ChatGPT for instance; you don't own the model, but you can easily purchase a subscription for a fixed monthly fee. OpenAI hosts the model on your behalf. If your use case requires that you access the model itself programmatically, you can also use the API. Put simple, an API is just a service endpoint to which you send a request and receive a response.

In that case, you pay per "token". A token is the unit in which LLMs read and generate text, and the unit in which usage is billed, much like electricity is billed per kWh. For simplicity, you can think of a token as roughly a word; that is, you pay approximately per generated word. You pay per *use*. An analogy would be car-sharing, where you pay per mile driven.



Of course, there are also other ways how to consume AI models, and the more time teams spend with AI, the more they usually look into renting or even

owning the stack. In the car analogy, this would be *renting* the car (e.g. a monthly lease) or even *buying* the car. For an AI model, this means renting or buying servers, and then hosting e.g. open weight models on this hardware. You now no longer pay per tokens – instead, you need to calculate in terms of hardware pricing, electricity bills and setup/maintenance efforts. You need to think about workloads (think of questions like "Is my server capable of handling not only the base loads, but also peak loads?", or "Is my server idle most of the time?"). Again, in the car analogy: have you bought a car but only drive it once per week?



"But how is this different to any other software purchase I ever made?", you might wonder. And you are correct. In the past, when you thought about introducing e.g. an ERP system, you could easily think about self-hosting.

The difference with AI is this: depending on your use case, it gets orders of magnitudes more expensive and complex. The hardware you need to rent or buy isn't common hardware, e.g. a simple UNIX machine that has a few dozen GBs of RAM and couple hundred of GBs or TBs of storage. Instead, you need to purchase specialized hardware that is easily eating your former hardware bill for breakfast. Sam Altman (CEO of OpenAI) once put it simple: "the compute costs are eye-watering". In addition, operating AI models isn't trivial. You will need to have experts to maintain your models if you want to run them in production.

> the compute costs are eye-watering
>
> **Sam Altman, CEO of OpenAI**

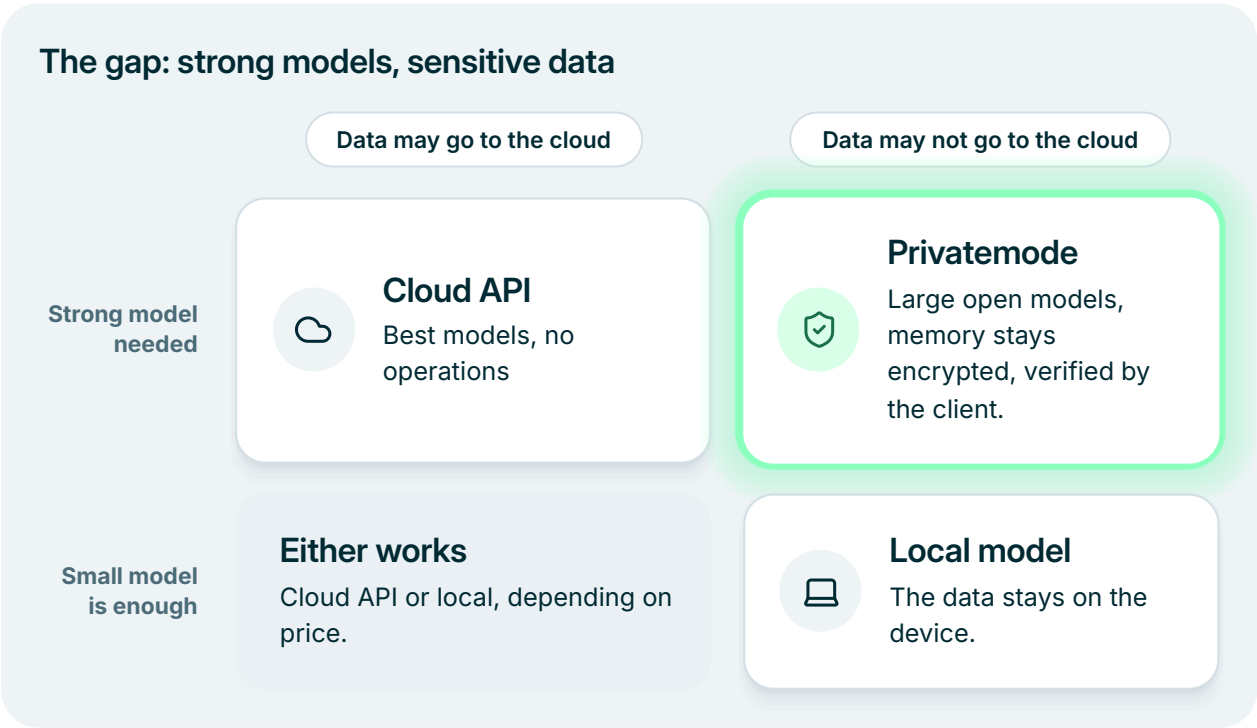
"But do I really need the large models for my use case?" is a next question you might wonder. That leads to second argument: it's hard to tell upfront.

AI models have been getting more capable on both ends:

- The frontier models that make the headlines are getting bigger; and as you can see in benchmarks like Artificial Analysis, research is not even remotely close to slowing down. AI models are getting better and better.
- Smaller models are getting better as well! This means that we are getting more capable AI models that could be running on a device like a MacBook or similar.

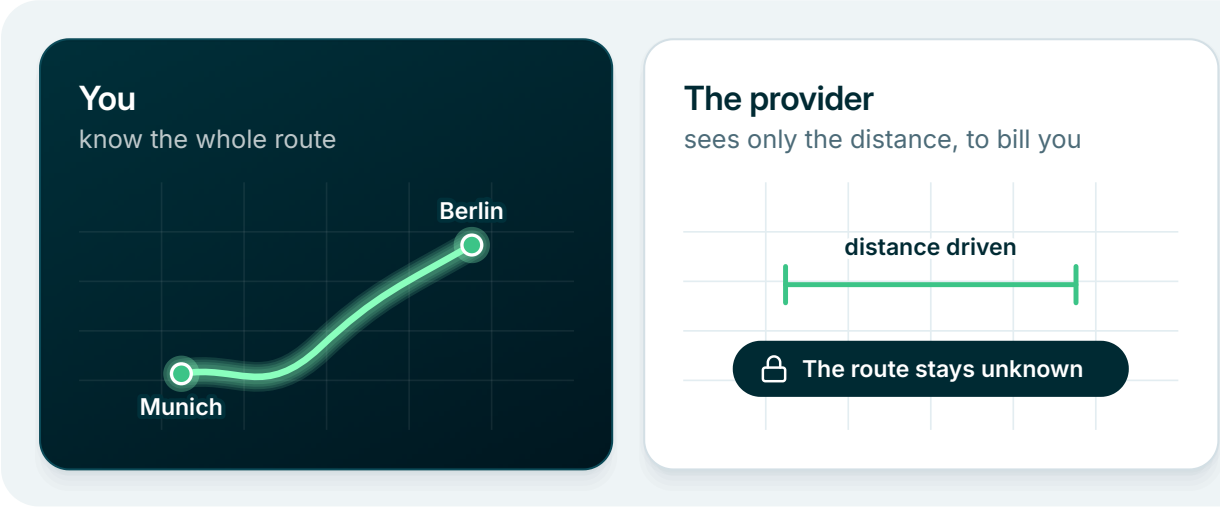
What kind of model you'll need however highly depends on the complexity of your task at hand. For "agentic AI", which means that the AI can really do delegated work on your behalf, you will need frontier models.

"Ok, but frontier models are in the cloud and I can't send the data to an API, because I am not allowed to or want to share it"; this is something we hear very often, specifically in regulated industries. The usual assumption: you need to self-host ("buy the car"), and this means you need to spend a ton of capital without knowing the outcomes – usually a risky scenario. Well, now is the time for us to shamelessly do our sales pitch for Privatemode AI. Because that is the field where confidential AI is highly relevant, and we argue that a confidential AI API belongs into each AI stack.



A confidential AI API is a service that hosts frontier AI models on your behalf, but it encrypts your requests and the responses verifiably end-to-end. End-to-end means that, by design, even the provider can't read the content of your request and the response itself; to the provider, the content is only

available in encrypted text. This is possible due to innovations in runtime-encryption, so-called Trusted Execution Environments (TEE). We spare you the technical details here. It's important to understand though that this encryption is actually verifiable, i.e. you can get proof that the infrastructure provider is not using different software to process your data or using other environments than the TEEs. To stick with the car analogy: this is like car-sharing and you pay per mile, but the car-sharing provider only knows how far you've traveled (otherwise they can't bill you), but they never know that you drove from Berlin to Munich. That is something only you know.



We claim: every company should put a confidential AI API into their stack.

Here are our three picks when confidential AI API can make your life that much easier:

When you are just starting:

You don't know yet which model you'll need, and you also don't know what kind of workloads you need to expect. A confidential AI API allows you to avoid capital-intensive and uncertain investments upfront.

When you use AI a lot and bought AI servers, but can't reliably serve peak loads:

Your use has a steady base and short peaks. In car-analogy, this is that you might have a big family and only one car; sometimes, the demand overlaps. In that case, it is great if you can rent a car on demand instead of buying a second one.

When your data is sensitive:

This is self-explanatory. You want access to the frontier league of AI models but can't use regular AI APIs.

And these scenarios aren't mutually exclusive. It could be that you are starting in one project and are uncertain about requirements while another use case already is running on high workloads and you must think about how you can enable the peak loads.

So confidential AI as an API truly should belong into each set of tools for a company that wants to transform their operations with AI.



Be aware: running large AI models is hard

At the time that we write this content piece, we hear more and more that people claim that it has gotten “so easy” to run an AI model on your own MacBook.

This is somewhat true. As mentioned above, significant improvements have been made in the capabilities of small models, both in what small models can do as well as how they can be served even on “tiny” machines. But this is a bit like saying “I can do my financial forecasting fully in Excel”. For some teams, this might be true. For other teams, it is completely unrealistic. We face that most teams imagine use cases for AI that are more on the agentic side, and for agentic use cases you usually need frontier models.

Even AI model providers that serve models to customers via API must look deeply into perfect utilization of hardware resources to make machines profitable. You need highly trained experts to keep the models running. You pay a fortune in hardware. A small miscalculation can be a financial nightmare.

There are definitely companies that should run frontier models in-house, on their own machines. Statistically speaking, you are most certainly not one of them. At least not in the first project. You are most likely far better served to use frontier models via API; in case of sensitive data, use a confidential AI API.

That being said: running small AI models is getting easier day by day. To give one example that pretty much every company faces: Optical Character Recognition (OCR) is a task to digitize e.g. scanned documents. This isn’t a task that requires frontier AI, and can very well be served also with smaller

models. If you don't want to host models at all, you can still use a cloud API for this though.

Understand your use case profiles

Different AI use cases have completely different requirements and usage patterns, as mentioned above.

If you've already progressed somewhat in your AI transformation, you likely see that you have a set of models. Maybe in one use case you are deploying a model by Mistral for OCR. In another use case you use Claude Code with Anthropic's Sonnet or Opus.

Basically, you already are adopting a hybrid stack by then. It is important to always understand the options you have; most companies that think about buying hardware for self-hosting are doing this unaware of the confidential AI API option.

Very simplified, an AI has input and output tokens. There is a third type, cached input tokens, but we are leaving them out for simplicity's sake.

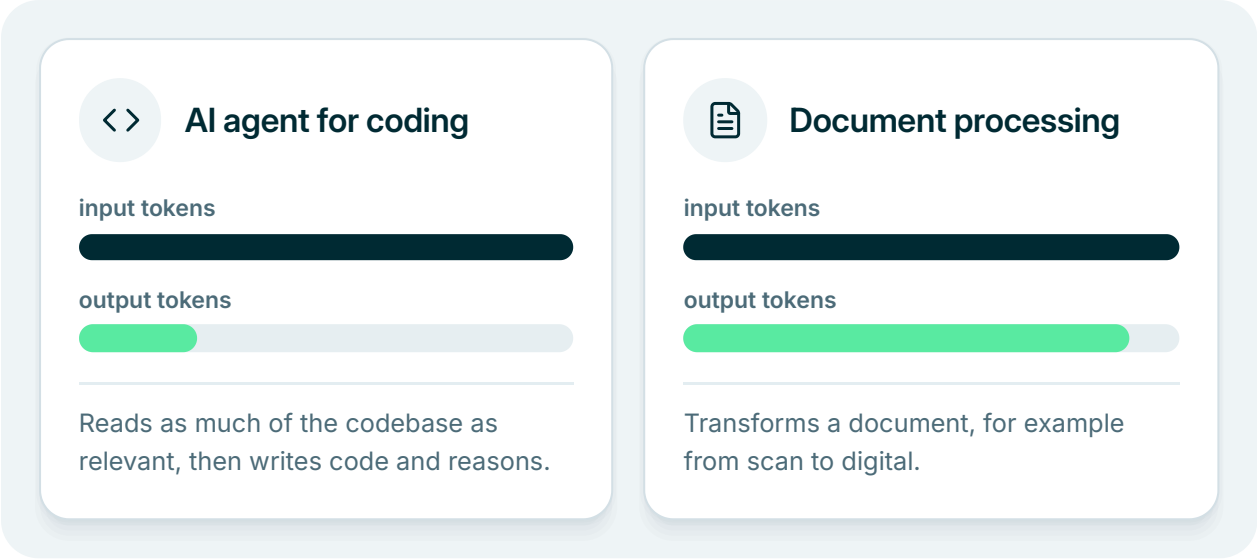


Most companies that think about buying hardware for self-hosting are doing this unaware of the confidential AI API option.

You will notice that different use cases have very different distributions when it comes to how they consume tokens. An AI agent for coding e.g. has usually very large volumes of input token consumption per task, since it reads as much of the codebase as relevant (in the ideal case) to solve the task. The task itself is e.g. to write a piece of code, which is a concatenation of output tokens. In between, it usually reasons, which are also output tokens.

A document processing use case on the other hand usually has similar input token consumption as it has output token consumption; it aims to transform a document (e.g. from scan to digital).

It is overall important to understand the token consumption profiles of your use cases. How many tokens are consumed per request on average, and which model is usually required? This will help you understand whether you have the amount of workloads that justify self-hosting or not.



To figure this out, we can only recommend starting with an API, always. It helps you answer the questions:

- How many tokens will you need in a month?
- Which model will be good enough?
- Will your team still use the AI use case in three months?

The unexpected challenges when self-hosting

We're not going to list every possible challenge here that comes with running frontier models yourself. However, we want to emphasize just a few:

You bought the wrong hardware:
Some machines are optimized for training models, others are optimized for inference (i.e. running an already trained model that doesn't need to train further). Also, you might notice that your AI is writing super slowly: chances are you bought a chip that isn't optimized for decoding, so you'll have to stick with an 8 token/second speed.

You can't serve your workload:
Maybe you have too little compute available (that is actually not too bad, given that you hopefully run this already profitable w.r.t. total cost of ownership), or too much (in that case your machine runs idle, and you are wasting expensive resources). If you have too little compute available, you can add an API in the mix.

You have issues staying up-to-date:
Models are being released in no time, and the requirements for operating can be quite hard. You might need to stick with the legacy AI if you don't have the right hardware.

You can't simply buy the hardware:
GPUs of this class are hardly sold one by one. You buy a complete server for several hundred thousand euros, and delivery can take months. An API is there the same day.

You need the right people to maintain:
We can't emphasize this enough. Calculate talented people that operate the models into the total cost of ownership!

For most companies, it is a very valid choice to host AI models to a certain degree themselves, but only for smaller models with very well-chosen use cases. Most frontier AI models shouldn't even be considered for 99% of companies in self-hosting.

Sensitive data does not force you to buy

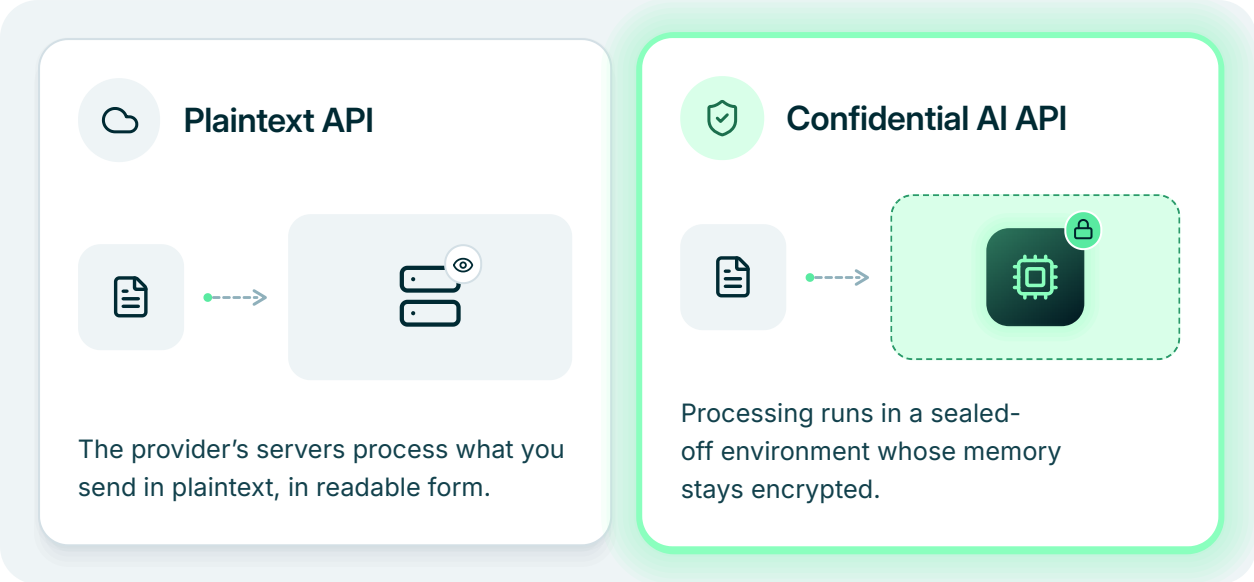
As mentioned above, most companies also see self-hosting as a necessity because they want to process sensitive data with powerful AI.

With a normal API, which we call a plaintext API, the provider's servers process what you send in plaintext. That is, in readable form.

A confidential AI API works differently. Processing runs in a sealed-off environment whose memory stays encrypted (the before-mentioned TEE). It is designed so that the provider cannot read your data, and the verification runs on your side before anything is sent.

More details about Confidential Computing

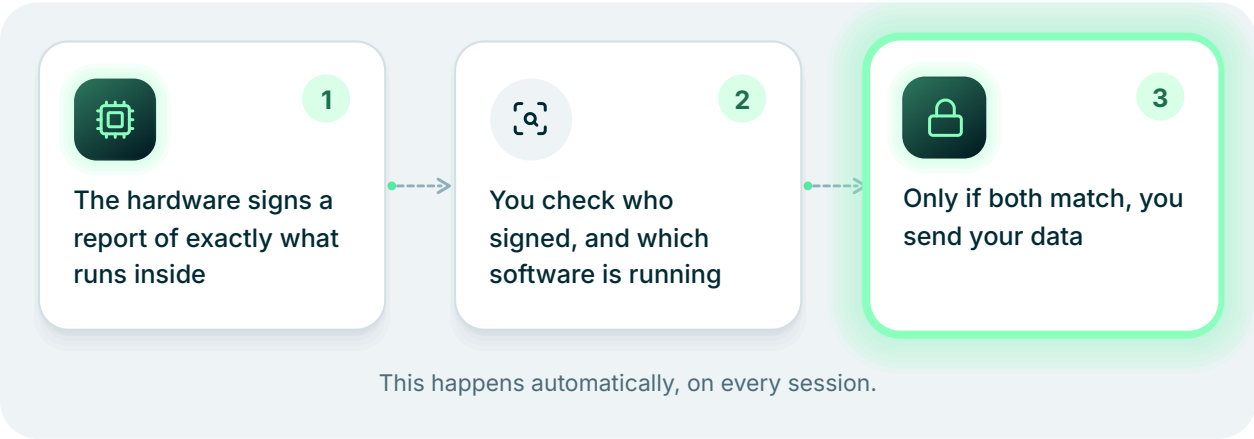
Confidential computing (the technology behind Confidential AI) is a feature of modern processors and GPUs. The hardware creates an isolated environment whose memory stays encrypted. This environment is shielded from everything



around it: the host operating system, the software that manages the servers, and the people who run them.

That is how it is designed so that the provider cannot read your data.

The hardware produces a signed report of the software running inside the protected environment. You can check this report to confirm two things: that the software the infrastructure provider lists is actually running, and that the environment is a genuine trusted execution environment (TEE). This is called remote attestation. Data is only sent if both checks pass. In Privatemode, this check runs automatically on your side for every session.



Put very simple, a confidential AI API is giving you best of both-worlds; the scalability and economics of a cloud, paired with the data protection of on-premises. A company providing a confidential AI API isn't "just" claiming that they won't use your data for training of other AI models or such; they actually can't by design.



What this means for data protection

Ask three things:

1. where is the data readable
2. who can reach it there
3. and who checks that?

Here is our overview

PATH	PLAINTEXT IS IN	WHO CAN REACH IT THERE	WHO CHECKS
Plaintext API	the provider's memory	the provider's administrators, the data center it rents from, the model's owner	the provider's auditors, after your data was processed
Confidential AI	a sealed-off environment on the chip with encrypted memory	nobody outside it; the provider is excluded by design (physical attacks aside)	software on your side, automatically before it sends data
Own hardware	your own memory	your administrators, anyone with physical access	your own audits and logs

Be aware that a confidential AI API doesn't mean you should not get a data processing agreement (DPA) with the vendor; that should always be agreed upon.

A simple test is:

“would it be problematic if the provider’s employees can read the requests I send?”

No

Any path will do. E.g. when the use case processes public code, website copy, test data, etc.

Yes

Use a confidential API, or your own hardware. Source code, contracts, customer or health data. A plaintext API is the wrong place for it, whatever the contract says, because the provider’s ability to read the data is itself the risk.

If you use a confidential API, make sure it is verified on your side. There are exceptions: some rules allow a plaintext API under conditions, such as § 203 of the German Criminal Code on professional secrecy, for a provider who is bound to secrecy. Ask your legal counsel.

Where to go next

Start with your own work, try your own numbers, and check our claims.



Start on an API

You can try out our confidential AI API with frontier AI models for free. Sign up with the promo code “buying-guide” (click on the button below), and your starting quota is 10 million tokens instead of 5 million.

Sign up with the code → portal.privatemode.ai/?promo=buying-guide

Try your own numbers

We provide calculators for several financial questions, so you can model according to your assumptions. Those are high-level calculators.

Open the financial models → privatemode.ai/financial-models

Check our claims

We share audit reports and security reviews on request, and the documentation of remote attestation is public.

Request the audit reports → privatemode.ai/request-documentation

Want to talk it through? Tell us what you plan to run, and we will tell you where an API fits.

Book a conversation → privatemode.ai/book-a-meeting



Johannes Hötter
VP Growth
jh@edgeless.systems



David Knichel, PhD
Solution Architect
dk@edgeless.systems



Moritz Eckert
VP Product & Technology
me@edgeless.systems