A CXO Briefing · Choosing an AI Model

Choosing an AI Model for Your Enterprise

How to pick the right model, and know when to switch

Executive Summary

There is no single best AI model. There is a best model for a given job, a given data path, and a given tolerance for lock-in.

Most model decisions are made by brand reflex: the team reaches for the most famous name and stops there. That works until the bill scales, the data has to stay inside your walls, or the vendor changes terms and you discover how expensive it is to leave. The model you run is a business decision wearing a technical costume.

The landscape splits into two families. Proprietary models, reached only through a vendor's API, usually set the frontier on raw capability and are the fastest to adopt. Open-weight models, which you can run on your own cloud tenant or your own hardware, have closed most of the quality gap, often win on price at scale, and leave you free to move. Neither family is the right answer on its own. Most serious deployments use more than one.

This briefing lays out the four ways to deploy a model, the five questions that actually decide the choice, what genuinely separates proprietary from open-weight, the licence and origin fine print to check before you build, and a simple guide for matching the model class to the job.

01 The Deployment Ladder

Four ways to put a model to work

Read this as a spectrum of ownership. As you move down, your control over the model and your data rises, and so does the effort to run it. The goal is not to reach the bottom. It is to match the rung to the job.

1
OPTION 01

Proprietary model, via the vendor's API

Models like Claude, GPT, or Gemini, reached only through the maker's API. Usually the highest raw capability and the fastest to adopt: no infrastructure to stand up. You cannot run these yourself, and your prompts travel to the vendor to be processed.

CAPABILITY CEILING — HIGHEST
CONTROL & PORTABILITY — LOW

2
OPTION 02

Open-weight model, on managed cloud

Open-weight models such as Llama, Mistral, or Qwen, run inside your own cloud account through a managed service like Microsoft Azure AI Foundry or AWS Bedrock. Strong capability, your data stays inside your tenant, and you keep the freedom to move the model later.

CAPABILITY CEILING — HIGH
CONTROL & PORTABILITY — HIGH

3
OPTION 03

Open-weight model, self-hosted

The same open-weight models, run on infrastructure you control, so prompts never leave your environment at all. Full sovereignty and, at steady high volume, often the lowest unit cost. The trade-off is real: you carry the hardware, the uptime, and the skills to run it.

CAPABILITY CEILING — HIGH
CONTROL & PORTABILITY — TOTAL

4
OPTION 04

A smaller model, fine-tuned to one job

A compact model shaped to a single, repeating task, extraction, classification, tagging, routing. Narrow by design, but fast and very cheap at scale, and easy to run yourself. Often the smartest choice for high-volume work that does not need a frontier brain.

CAPABILITY (IN ITS LANE) — FOCUSED
CONTROL & COST-FIT — EXCELLENT

02 The Decision

The five questions that actually decide it

Capability benchmarks make headlines, but they rarely decide the right choice on their own. These five questions do most of the work, and they are the ones your technology lead will want answered.

Before you commit to a model, answer these

- **How hard is the task, really?** Match the model to the difficulty. Paying frontier prices to extract a date from an email is waste; trusting a tiny model with high-stakes reasoning is risk. Most workloads sit in the middle.
- **What does it cost at your real volume?** Price is per token, so the number that matters is price multiplied by your actual usage. A model that is far cheaper and almost as good usually wins once you are running millions of calls a month.
- **Where does your data travel?** To the vendor's servers, to your own cloud tenant, or never past your walls. This single answer often rules out whole options before capability is even discussed. (It is the subject of our companion briefing on data safety.)
- **How hard is it to leave?** Open weights are portable: you can move them between hosts or in-house. Proprietary models carry a switching cost in re-engineering and re-prompting. Low price today does not offset expensive lock-in tomorrow.
- **Who made it, and does that matter for your context?** Different models carry different licences, usage limits, and jurisdictions. None of that reflects on quality, but it can reflect your clients', sector's, or board's procurement rules, so check it early rather than late.

03 Proprietary vs Open-weight

What actually separates the two families

This is the choice most people frame as "which is better," when the useful question is "which fits this job." The two families differ less on quality than on ownership, cost shape, and how free you are to change your mind.

	Proprietary models Claude, GPT, Gemini and similar	Open-weight models Llama, Mistral, Qwen, DeepSeek and similar
Top-end capability	Generally sets the frontier	Closing the gap quickly; often leads on price-to-performance
Can you run it yourself	No, vendor API only	Yes, on your cloud tenant or your own hardware
What you pay for	The model and its hosting, bundled together	Compute only; the model weights themselves are free
Cost shape at scale	Per-token vendor pricing, set by the vendor	Can fall sharply at steady high volume if self-hosted
Switching cost	Higher: re-engineering and re-prompting to move	Lower: the same weights run across many hosts
Licence and terms	The vendor's terms of service	An open licence, which you should read (see next section)

Capability rankings and prices change almost monthly, so treat any specific comparison as a snapshot and re-check before you commit. One phrase worth knowing: "open weight" is not the same as "open source." The trained model is released for you to run and adapt, but the training data and full recipe usually are not. For most enterprise purposes, the freedom to run, host, and move the model is what matters.

04 The Fine Print

Licence and origin: facts to confirm up front

When you build on an open-weight model, you inherit its licence and, for some organisations, questions about where and by whom it was made. None of this is a judgement on a model's quality. It is procurement housekeeping that is cheap to do early and costly to discover late.

Three things to check before you build

A capable model with the wrong terms for your context is the wrong model. These are quick to verify and easy to overlook.

Licence type

Permissive licences such as Apache 2.0 and MIT let you use, modify, self-host, and redistribute freely. Others attach conditions you need to read.

Usage limits

Some "open" licences add restrictions, for example caps on user numbers or limits on certain sectors. Confirm yours fits your scale and use.

Origin & residency

Some clients, sectors, or boards have rules on a model's origin or where data is processed. Check these against your own procurement policy.

The point: the question is not which flag a model flies or which lab built it. It is whether its licence, limits, and origin are compatible with your obligations, confirmed before you build on it rather than after.

05 Resilience

Don't bet the business on a single model

Models get deprecated, prices get revised, and endpoints have outages. Tying a production system to exactly one model from one vendor turns each of those ordinary events into your problem.

The pattern: a primary, a fallback, and a way to switch

Mature deployments route between models rather than depending on one. A primary model handles the work at the quality you need. A cheaper or independently hosted model stands behind it, so if the primary is unreachable or a call fails, the system keeps running instead of going dark. Some teams also keep a high-quality reference model to spot-check output quality over time.

The benefit is leverage as much as uptime. When you can switch models with a configuration change, no single vendor's pricing or roadmap can hold your product hostage. Building this routing layer is a modest piece of engineering, and it is what separates a demo from a system you can run a business on.

06 The Takeaway

Which model class for which job

The practical summary. Match the model to the work, not to the brand. Most firms will run more than one of these at the same time.

High-volume, simple, repeating tasks Extraction, classification, tagging, routing, where the job is narrow and the scale is large	SMALL / FINE-TUNED
Everyday drafting and internal work Summaries, first drafts, internal Q&A, where good-enough at low cost beats the absolute frontier	MID-TIER, MANAGED
Hardest reasoning, high-stakes output Ambiguous, complex, or client-facing work where quality differences carry real consequences	FRONTIER MODEL
Sensitive data that cannot leave Anything bound by confidentiality, regulation, or residency rules on where it may be processed	OPEN-WEIGHT, IN YOUR ENVIRONMENT

E EnableWealth
AGENTIC AI & AUTOMATIONS

enablewealth.ai

This briefing is general guidance for business leaders, not legal or procurement advice, and it does not recommend any specific model or vendor. AI model capabilities, pricing, and licence terms change frequently; verify current terms with each provider and your own counsel before relying on them.

Drawn from the public documentation and licences of leading proprietary and open-weight model providers, and from current industry benchmarks and pricing surveys. Model names are illustrative of each category, not endorsements. Current as of June 2026.