FREE ENTERPRISE AI TOOLKIT

Enterprise LLM Selection Checklist

Score any Small Language Model or Frontier LLM before you commit

A 25-point scoring framework for enterprise AI model selection, with worked examples, weighted scorecard and decision guide.

How to Use This Checklist

Choosing between a Small Language Model (SLM) and a Frontier Large Language Model (LLM) is rarely a question of which model is "smarter". It is a question of **fit**: which option delivers the accuracy you need, at a cost you can sustain, under the security rules you must follow, inside the systems you already run.

This checklist turns that judgement into a repeatable process. You score each model option against 25 criteria grouped into five weighted factors, add up a score out of 100, and use the decision guide to choose between an SLM, a frontier LLM or a hybrid of both.

Who it is for

- **IT and engineering leaders** deciding what to build, buy or host.
- **AI/ML and data teams** running model evaluations and pilots.
- **Product owners** who need a defensible recommendation for stakeholders.
- **Procurement and vendor managers** comparing providers and contracts.

- **CXOs and business heads** who want a clear, comparable summary before approving spend.

When to use it

- **Before a pilot**, to decide which models are worth testing at all.
- **During vendor selection**, to compare API providers and self-hosted options on the same scale.
- **Before switching models**, when costs rise, a provider changes terms or a newer model is released.
- **At annual review**, to check whether your current choice still fits your volume and requirements.

The four steps

1. **Define the use case.** Fill in the Use Case Definition box below. One checklist = one use case. If you are evaluating three use cases, complete three checklists.
2. **Score each model on all 25 criteria.** Use the 1–5 scale below. Score how well each option meets the criterion for this specific use case, not in general.

3. **Apply the factor weightings.** Average each factor, then convert it to weighted points using the formula on the Scorecard page.

4. **Read the result with the decision guide.** Check the margin between the two options and review the red flags that can override the score.

Scoring scale

Score	Meaning	Example
5	Excellent fit; meets the criterion fully with no extra effort	Model already meets your latency target with room to spare
4	Good fit; minor gaps that are easy to close	Meets latency target at normal load, needs caching at peak
3	Acceptable; workable with moderate effort or cost	Meets target only after a hardware upgrade
2	Weak fit; significant effort, cost or risk to make it work	Misses target by 2–3x; would need architecture changes

1	Poor fit or not possible	Cannot meet the target under any realistic setup

Use Case Definition box

Complete this before scoring. Most model selection mistakes come from scoring against a vague use case.

Field	Your answer
Use case name	
Business owner / department	
What the model must do (one sentence)	
Expected volume (requests per month)	
Data sensitivity (Low / Medium / High)	

Latency target	
Option A (SLM being evaluated)	
Option B (Frontier LLM being evaluated)	
How success will be measured	

EXAMPLE | The running example used in this checklist

To show how each criterion works in practice, this checklist follows one use case from start to finish:

- **Use case:** Automatically classify incoming customer support emails into 12 categories (card issues, loan queries, KYC updates and so on) at a mid-size retail bank.
- **Volume:** about 400,000 emails per month, with Monday-morning peaks.
- **Data sensitivity:** High. Emails contain names, account details and transaction information.
- **Latency target:** under 1 second per email so tickets are routed instantly.

- **Option A:** a fine-tuned open-weight small model hosted in the bank's private cloud.

  Option B: a frontier LLM accessed through a cloud API.

Factor 1: Task Complexity

Weight: 25% • Criteria 1–5

Task complexity carries the highest weight because it decides whether a model can do the job at all. Cost savings, privacy benefits and easy integration mean nothing if the model cannot produce reliable answers for the task.

The five criteria below help you work out whether your task is **narrow and repeatable** (where SLMs shine) or **broad and reasoning-heavy** (where frontier LLMs earn their higher cost).

TIP | Why this factor matters

> The most common selection mistake is choosing by model size. A frontier LLM on a simple classification task wastes money; an SLM on a multi-step analysis task produces unreliable answers. Getting this factor right prevents both.

1. The task is narrow and well-defined

A narrow task has a clear input, a clear expected output and a limited range of correct answers. The more precisely you can describe the job, the more likely a small, specialised model can do it well.

What to check

- Can you describe the task in a single sentence without "and" or "depending on"?
- Is the output one of a fixed set of options (labels, fields, yes/no)?
- Would two human experts give the same answer most of the time?

How to score

If the task is narrow, score the SLM 4–5. Frontier LLMs usually handle narrow tasks well too, so they often score 4; the difference shows up later

in cost and latency. If the task is broad and open-ended, score the SLM 1–2 and the LLM 4–5.

> **EXAMPLE | Narrow vs broad**
>
> **Narrow:** "Classify this support email into one of 12 categories." (Running example: SLM 5, LLM 4.)
>
> **Broad:** "Read this customer complaint, decide what went wrong, and suggest a goodwill gesture that fits our policy and the customer's history." This needs judgement across several sources, so it favours a frontier LLM.

2. Inputs follow a predictable format

Models trained or fine-tuned on a consistent input format become very accurate on it. When inputs vary widely in layout, language, length or quality, the model has to generalise, and larger models generally generalise better.

What to check

- Do inputs come from the same template, system or form?
- Are inputs in one or two languages, or many?

- How much noise is there (scanned pages, typos, forwarded threads, attachments)?

How to score

Predictable, templated inputs: SLM 4–5. Highly varied, messy or multilingual inputs: SLM 2–3, LLM 4–5.

> **EXAMPLE | Predictable vs varied inputs**
>
> **Predictable:** purchase orders exported from the same ERP template every day.
>
> **Varied:** commercial contracts from 200 different counterparties, each with its own structure and wording.
>
> **Running example:** support emails are free text but short, mostly in English and Hindi, and follow common patterns. SLM 4, LLM 5.

3. Single-step reasoning is enough

Single-step tasks need one decision: identify, extract, label or summarise. Multi-step tasks need the model to chain decisions together: compare, calculate, infer, then recommend. Each extra step multiplies the chance of error, and frontier LLMs are much stronger at keeping long reasoning chains on track.

What to check

- How many distinct decisions does the model make before producing the answer?
- Does the task involve calculations, comparisons across documents or "if this, then that" logic?
- Does the answer need an explanation of why, not just what?

How to score

One step: SLM 4–5. Two or three steps: SLM 3, LLM 4–5. Long chains of reasoning or planning: SLM 1–2, LLM 5.

> **EXAMPLE | One step vs many**
>
> **Single step:** "Is this document an invoice or a receipt?"
>
> **Multi-step:** "Does the indemnity clause in this contract conflict with our standard policy, what is the financial exposure, and what should we negotiate?"
>
> **Running example:** picking one category is a single decision. SLM 5, LLM 5.

4. Outputs can be checked against fixed rules

When outputs can be validated automatically, you can catch and correct a smaller model's mistakes cheaply. When outputs are open-ended text that only a human can judge, errors are expensive to detect, which raises the value of a stronger model.

What to check

- Can each output be validated against a list, schema, format or business rule?
- Can low-confidence outputs be flagged automatically for human review?
- What is the cost of a wrong answer that slips through?

How to score

Rule-checkable outputs: SLM 4–5. Open-ended outputs requiring expert review: SLM 2–3, LLM 4.

> **EXAMPLE | Checkable vs open-ended outputs**
>
> **Checkable:** an extracted GST number either matches the official 15-character format or it does not; an assigned label is either on the approved list or it is not.

> **Open-ended:** a two-page market summary for the leadership team can only be judged by a reader.
>
> **Running example:** the category must be one of 12 approved labels, and anything below a confidence threshold goes to a human queue. SLM 5, LLM 4.

5. Broad general knowledge is not required

Frontier LLMs carry a huge amount of general world knowledge. SLMs carry far less. If your task can be answered entirely from your own data, especially with retrieval-augmented generation (RAG) over an approved knowledge base, the SLM's smaller knowledge base is not a problem.

What to check

- Can every correct answer be found in your own documents, systems or labelled data?
- Does the task need knowledge of current events, other industries or general facts?
- Will you ground the model with a knowledge base, or rely on what it already knows?

How to score

Domain-limited and grounded in your data: SLM 4–5. Needs broad world knowledge: SLM 1–2, LLM 5.

EXAMPLE | Domain knowledge vs world knowledge

Domain-limited: an HR assistant that answers employee questions using only 40 internal policy documents.

Broad: a research assistant that compares competitors' strategies across markets and industries.

Running example: categories are defined entirely by the bank's own taxonomy. SLM 5, LLM 4.

TIP | Quick read on Factor 1

If the SLM scores 4 or 5 on at least four of these five criteria, your task is a strong candidate for a small model. If it scores 2 or below on two or more, a frontier LLM, or a hybrid with an LLM handling the hard cases, is usually the safer choice.

Factor 1 scoring table

#	Criterion	SLM (1–5)	LLM (1–5)	Notes
1	The task is narrow and well-defined			
2	Inputs follow a predictable format			
3	Single-step reasoning is enough			
4	Outputs can be checked against fixed rules			
5	Broad general knowledge is not required			
	Factor average (sum ÷ 5)			

Factor 2: Cost & Infrastructure

Weight: 20% • Criteria 6–10

Frontier LLM pricing looks cheap per request, but enterprise workloads run thousands or millions of requests a month. SLMs can look cheap too,

until you add GPUs, fine-tuning, monitoring and the people needed to run them.

This factor makes you calculate the **total cost of ownership (TCO)** for each option, not just the price tag.

> **TIP | Why this factor matters**
>
> Cost is where pilots that worked fail in production. A model that costs very little for 1,000 test requests can become a major budget line at 1 million requests a month.

6. Monthly inference/API cost estimated at expected volume

For API-based models, cost scales with tokens: every word in and every word out is billed. For self-hosted models, cost is mostly fixed (the server runs whether busy or idle) until you need more capacity.

What to check

- Average input tokens and output tokens per request.
- Monthly request volume, including peaks and expected growth.
- Current price per million tokens from the provider's official pricing page (prices change often, so always check the current rate).

- For self-hosted: monthly cost of the GPU or CPU instances needed to serve peak load.

How to score

Score each option on how comfortably its monthly cost fits your use case: lowest realistic cost at your volume = 5, unaffordable at your volume = 1.

> **EXAMPLE | Estimating the running example**
>
> **Formula:** monthly tokens = requests × (input tokens + output tokens).
>
> **Running example:** 400,000 emails × about 1,200 tokens each = about 480 million tokens per month through the API (Option B). Multiply by the provider's current rate and compare it with the fixed monthly cost of the private-cloud GPU instance for Option A.
>
> At this volume and with short outputs, the self-hosted SLM is typically far cheaper. SLM 5, LLM 2.

7. Hosting and hardware requirements identified

Small models (roughly in the 1–10 billion parameter range) can often run on a single GPU, and some quantised versions run on CPUs or edge devices. Frontier models are usually consumed through a provider's API because

self-hosting them needs large multi-GPU clusters, where that is offered at all.

What to check

- GPU type and memory needed to serve the model at your peak load.
- Whether a quantised (compressed) version meets your accuracy bar.
- For APIs: network connectivity, private networking options and the provider's rate limits.
- Lead time to procure hardware or cloud capacity.

How to score

Requirements fully identified and available: 5. Requirements unknown, or hardware not available within your timeline: 1–2.

> **EXAMPLE | Hardware planning**
>
> **Option A:** the bank's private cloud already has GPU instances approved for analytics workloads; one instance plus one standby is enough. SLM 4.
>
> **Option B:** no hardware needed, but the provider's default rate limit is below Monday peak volume and needs a higher tier. LLM 4.

8. Fine-tuning and retraining costs accounted for

SLMs often reach high accuracy only after fine-tuning on your labelled data. That means data preparation, labelling, training runs and periodic retraining as your categories or data change. Frontier LLMs often get good results with prompt engineering and a few examples, so upfront training cost is lower.

What to check

- How many labelled examples are needed, and who will label them?
- Cost of training runs (compute and engineering time).
- How often the model must be retrained (new categories, new products, data drift).

How to score

Low, predictable training effort: 4–5. Large labelling effort or frequent retraining with no budget set aside: 1–2.

> **EXAMPLE | Training effort**

> **Option A:** needs about 5,000 historic emails labelled into the 12 categories; the support team can do this in two weeks, and quarterly retraining is planned. SLM 3.
>
> **Option B:** works with a well-designed prompt and 2–3 examples per category; no training run needed. LLM 5.

9. Monitoring and support costs included

Every production model needs ongoing monitoring: accuracy tracking, drift detection, logging, alerting and someone to respond when something breaks. These costs are often left out of the business case.

What to check

- Tools for logging, tracing and evaluating model outputs.
- Human review of a sample of outputs (quality assurance).
- On-call or support coverage for incidents.
- Regular evaluation runs against your test set.

How to score

Monitoring planned and budgeted: 4–5. Not yet considered: 1–2. Self-hosted options usually need more in-house support; API options shift some of it to the provider.

> **EXAMPLE | Monitoring plan**
>
> The bank samples 2% of classified emails each week for human review and tracks accuracy on a dashboard. Option A also needs GPU health and uptime monitoring by the platform team. SLM 3, LLM 4.

10. 12-month TCO fits the approved budget

Add criteria 6–9 together for a 12-month view, including expected volume growth. This is the number finance will ask for, and the one that decides whether the project survives past the pilot.

What to check

- All cost lines included: inference, hosting, training, monitoring, support and integration.
- Volume growth included (for example, +30% over 12 months).
- A contingency buffer for price changes or unexpected load.

How to score

Comfortably within budget with buffer: 5. Within budget only at today's volume: 3. Over budget: 1.

Example: 12-month TCO comparison for the running example (relative)

Cost line	Option A: SLM (private cloud)	Option B: Frontier LLM (API)
Inference / API usage	Low (fixed instance cost)	High (scales with every token)
Hosting & hardware	Medium	None
Fine-tuning & retraining	Medium	Low
Monitoring & support	Medium	Low–Medium
Integration	Medium	Low
12-month total at 400K emails/month	**Lower**	**Higher, and grows with volume**

In the running example, the SLM has higher setup effort but much lower running cost, so it wins on 12-month TCO. SLM 5, LLM 2.

Factor 2 scoring table

#	Criterion	SLM (1–5)	LLM (1–5)	Notes
6	Monthly inference/API cost estimated at expected volume			
7	Hosting and hardware requirements identified			
8	Fine-tuning and retraining costs accounted for			
9	Monitoring and support costs included			
10	12-month TCO fits the approved budget			
	Factor average (sum ÷ 5)			

Factor 3: Data Privacy & Security

Weight: 20% • Criteria 11–15

Enterprise AI systems process customer records, financial data, employee information and confidential documents. Where that data travels, who can see it and how long it is kept are often non-negotiable requirements, not preferences.

This factor checks whether each model option can meet your data protection obligations in practice.

> **TIP | Why this factor matters**
>
> A model that fails a compliance review cannot go live, however well it performs. Scoring this factor early prevents months of wasted pilot work.

11. Data classification completed

Before you can judge whether a model is safe to use, you need to know what data it will see. Classify the data the model will receive as input and the data it might produce as output.

What to check

- Personal data (names, phone numbers, addresses, ID numbers).
- Financial data (account numbers, transactions, card details).
- Health, employee or other special-category data.
- Confidential business information (pricing, strategy, source code, contracts).

How to score

This criterion scores your readiness rather than the model. If classification is complete, score both options 5. If not, score both 1–2 and finish it before going further.

> **EXAMPLE | Classifying the running example**
>
> Support emails contain customer names, partial account numbers and transaction details. Classification: **personal + financial data, High sensitivity**. Both options: 5 (classification is complete).

12. Data residency and compliance needs met

Many regulations control where personal data can be processed and stored. Check each model option against the rules that apply to your organisation, industry and customers.

What to check

- **GDPR** if you process data of people in the EU.
- **India's Digital Personal Data Protection (DPDP) Act, 2023** for personal data of people in India.
- **HIPAA** for protected health information in the US.
- Sector rules, such as RBI requirements for payment data in Indian banking.
- Whether the API provider offers processing in your required region, and in writing.

How to score

Fully compliant with your required region and regulations: 5. Compliant only with extra contracts or configuration: 3. Not compliant: 1.

> **EXAMPLE | Residency check**
>
> The bank's compliance team requires customer data to be processed in India. Option A runs in the bank's own India-based private cloud: SLM 5. Option B's provider offers an India region only on a higher enterprise tier, with legal review still pending: LLM 3.

13. Deployment model chosen

The deployment model decides where your data goes. Each option trades control against convenience.

What to check

- Which of the four deployment models below each option supports.
- Which deployment model your security team will approve for this data classification.

How to score

Option supports the deployment model your security team requires: 5. Supports it only with major changes: 2–3. Cannot support it: 1.

Deployment models at a glance

Deployment model	Where data goes	Best for
Public cloud API	To the provider's cloud, under their terms	Low-sensitivity data, fast start, frontier capability
Private LLM (dedicated or VPC deployment)	Into an isolated environment in your cloud account or a dedicated tenant	Sensitive data where you still want managed services
Local LLM (on-premises)	Stays inside your own data centre	Highly regulated data, strict residency rules
On-device AI	Stays on the user's laptop, phone or edge device	Offline use, field devices, the strictest privacy needs

Running example: the security team requires a private deployment for High-sensitivity data. Option A is a private LLM by design: SLM 5. Option B would need the provider's dedicated deployment offering: LLM 3.

14. Encryption, access controls and audit logging in place

Whatever model you choose, the surrounding system must protect data in transit and at rest, control who can use the model and record what happened.

What to check

- Encryption in transit (TLS) and at rest.
- Role-based access control and single sign-on for users and services.
- Audit logs of prompts and responses, with sensitive fields masked.
- Protection against prompt injection and data leakage through outputs.

How to score

All controls supported and configured: 5. Supported but not yet configured: 3. Gaps that cannot be closed: 1.

> **EXAMPLE | Security controls**
>
> Option A inherits the bank's existing private-cloud controls (encryption, IAM, central logging): SLM 5. Option B supports encryption and SSO, but prompt logging with field masking must be built in the bank's gateway: LLM 4.

15. Provider data-retention and training-use terms reviewed

For any third-party model or API, read the contract terms on what happens to your data after each request. Consumer and enterprise terms often differ.

What to check

- Does the provider store prompts and responses? For how long?
- Can your data be used to train or improve the provider's models?
- Is a zero-data-retention or reduced-retention option available?
- Who at the provider can access your data, and under what conditions?
- What happens to your data if you end the contract?

How to score

Terms reviewed and acceptable: 5. Acceptable only with a negotiated addendum: 3. Unacceptable or not reviewed: 1. For a fully self-hosted model you control, score 5.

> **EXAMPLE | Contract review**

Option A is self-hosted with open-weight model files, so no data leaves the bank: SLM 5. Option B's enterprise terms exclude training on customer data, but retention settings need a signed addendum: LLM 3.

WATCH OUT | A smaller model is not automatically a secure model

Hosting an SLM locally keeps data inside your walls, but it does not remove the need for encryption, access control, monitoring, prompt-injection defences and security testing. Score criteria 14 and 15 on what is actually in place, not on where the model runs.

Factor 3 scoring table

#	Criterion	SLM (1–5)	LLM (1–5)	Notes
11	Data classification completed			
12	Data residency and compliance needs met			
13	Deployment model chosen			

14	Encryption, access controls and audit logging in place			
15	Provider data-retention and training-use terms reviewed			
	Factor average (sum ÷ 5)			

Factor 4: Performance & Scalability

Weight: 20% • Criteria 16–20

A model that impresses in a demo can disappoint under real enterprise load. This factor checks performance on **your data, at your volume, under your peak conditions**.

Score this factor only after testing. If you have not tested yet, score both options 3 as a placeholder and come back to it after the pilot.

> **TIP | Why this factor matters**

> Public benchmark scores measure general ability on standard tests. They cannot tell you how a model handles your documents, your customers' language or your Monday-morning traffic spike.

16. Accuracy tested on real business data

Build a "golden test set": a few hundred real examples with the correct answer agreed by your experts. Run every candidate model against the same set and compare the results.

What to check

- A test set of 200–500 real, representative examples (including hard and rare cases).
- The right metric for the task: accuracy, precision/recall, F1 or a human quality rating.
- Results broken down by category, so weak spots are visible.

How to score

Meets or beats your accuracy target: 5. Within a few points of target: 3. Well below target: 1.

EXAMPLE | Accuracy test

On 500 labelled support emails, the fine-tuned SLM reached 94% accuracy and the frontier LLM (prompted, not fine-tuned) reached 91%. The target was 90%. SLM 5, LLM 4. (Illustrative figures; your results will differ.)

17. Latency meets the user-experience target

Latency is how long the user or system waits for a response. Measure the typical time (p50) and the slow-case time (p95), because users notice the slow cases.

What to check

- Your target for this use case (real-time chat, near-real-time routing or overnight batch).
- Measured p50 and p95 response times at realistic load.
- Network latency for API calls from your region.

How to score

Comfortably within target at p95: 5. Within target at p50 but not p95: 3. Misses target: 1.

EXAMPLE | Latency test

Target: under 1 second per email. The SLM, running close to the ticketing system, averaged about 150 ms. The API averaged about 900 ms with slower outliers at peak. SLM 5, LLM 3.

18. Throughput handles peak request volume

Throughput is how many requests the system can process per second or minute. Design for peaks, not averages.

What to check

- Peak requests per minute (often 2–5x the average).
- API rate limits and the tier needed to raise them.
- For self-hosted: concurrent requests per GPU and autoscaling options.
- Behaviour when overloaded: queueing, slowdown or failure.

How to score

Handles peak with headroom: 5. Handles peak only with queueing: 3. Fails at peak: 1.

EXAMPLE | Throughput test

Monday 9–11 am volume is about 3x the daily average. The SLM instance handled the peak with a short queue: SLM 4. The API hit rate limits on the default tier: LLM 3 until the tier is upgraded.

19. Output consistency verified across repeated runs

Enterprise processes need predictable outputs. The same input should produce the same (or an equivalent) answer, in the expected format, every time.

What to check

- Run a sample of inputs several times and measure how often the answers agree.
- Check format compliance (valid JSON, approved labels, required fields).
- Use deterministic settings (such as low temperature) where the provider supports them.

How to score

Very consistent with near-perfect format compliance: 5. Occasional variation: 3. Frequent variation or format errors: 1.

> **EXAMPLE | Consistency test**
>
> The team ran 50 emails 5 times each. The SLM returned the same label 99% of the time; the LLM 96%, with occasional extra explanation text that broke the parser until the prompt was tightened. SLM 5, LLM 4.

20. Can scale with 2–3x growth in workload

Plan for success. If the use case works, volume will grow and other teams will want to use it. Check what breaks first as load increases.

What to check

- Cost curve: does cost rise in line with volume (API) or in steps (more servers)?
- Capacity: can you get more GPUs or higher rate limits quickly?
- Architecture: can the solution be reused for similar use cases?

How to score

Scales smoothly to 3x: 5. Scales with planned investment: 3. Hard limits: 1.

EXAMPLE | Scaling check

At 3x volume (1.2 million emails a month), the SLM needs one more GPU instance, a small step cost: SLM 4. The API scales easily in capacity but cost triples: LLM 4.

TIP | Test under realistic conditions

Run performance tests with production-like data, at production-like volume, from the same network location your system will use. Ten clean demo prompts will not show you the problems that appear at 10,000 requests an hour.

Factor 4 scoring table

#	Criterion	SLM (1–5)	LLM (1–5)	Notes
16	Accuracy tested on real business data			
17	Latency meets the user-experience target			

18	Throughput handles peak request volume			
19	Output consistency verified across repeated runs			
20	Can scale with 2–3x growth in workload			
	Factor average (sum ÷ 5)			

Factor 5: Integration & Maintenance

Weight: 15% • Criteria 21–25

A model only creates value once it works inside your existing systems and keeps working month after month. This factor covers the day-two work: connecting, updating, switching and supporting the model.

> **TIP | Why this factor matters**

> Integration and maintenance effort is easy to underestimate during selection and hard to reduce later. A slightly weaker model that fits your stack can beat a stronger one that needs a custom platform around it.

21. Compatible with existing apps, APIs and databases

Check how each option connects to the systems it must read from and write to, such as your CRM, ticketing tool, data warehouse or identity provider.

What to check

- Available SDKs, standard API formats and existing connectors.
- Authentication compatible with your identity system.
- Support for the input and output formats your systems use.

How to score

Plugs into existing systems with standard connectors: 5. Needs custom middleware: 3. Major rework needed: 1.

> **EXAMPLE | Integration check**

> The ticketing system already calls internal REST services. The SLM is served behind an OpenAI-compatible endpoint inside the private cloud, so both options use the same integration code. SLM 4, LLM 5.

22. Model update and versioning process defined

Models change. API providers release new versions and retire old ones; open-weight models get improved releases. Every change can shift accuracy, so you need a controlled way to update.

What to check

- Can you pin a specific model version?
- How much notice does the provider give before retiring a version?
- Will you re-run your golden test set before every upgrade?
- Who approves a model change going to production?

How to score

Full version control and a defined regression-test process: 5. Partial control: 3. No control over when the model changes: 1.

EXAMPLE | Versioning

With the self-hosted SLM, the bank controls exactly when the model changes: SLM 5. The API allows version pinning, but versions are retired on the provider's schedule, which forces periodic re-testing: LLM 3.

23. Vendor lock-in and provider-dependency risk assessed

Dependency on one provider exposes you to price rises, policy changes, outages and product retirement. Reduce the risk by designing so the model can be swapped.

What to check

- Do you rely on provider-specific features that others do not offer?
- Is there an abstraction layer (an AI gateway or internal API) between your apps and the model?
- How long would it take to switch to an alternative model?

How to score

Easy to switch (days to weeks): 5. Possible with significant rework: 3. Effectively locked in: 1.

EXAMPLE | Lock-in check

The bank routes all model calls through an internal AI gateway, so switching either option means changing configuration and re-testing, not rewriting apps. Open-weight SLM 5; proprietary LLM 4.

24. Fallback plan defined if the model or API fails

Every model will sometimes be unavailable or unsure of its answer. Decide in advance what happens then, so the business process keeps running.

What to check

- What happens during an outage: backup model, queue, or manual process?
- What happens to low-confidence outputs: human review or a stronger model?
- Has the fallback been tested?

How to score

Fallback defined and tested: 5. Defined but untested: 3. No fallback: 1.

EXAMPLE | Fallback design

If the SLM's confidence is below 0.80, or the instance is down, the email goes to a human triage queue. The same rule applies to the API option during provider outages. SLM 5, LLM 4.

25. Internal team has the skills to run and maintain it

Self-hosted models need MLOps skills: serving, scaling, fine-tuning and monitoring. API-based models need strong prompt engineering, evaluation and vendor management. Be honest about your team's current skills.

What to check

- Who will own the model in production?
- Does the team have experience with the required tooling?
- Is there budget and time for training or hiring?

How to score

Team has the skills today: 5. Skills can be built within the project timeline: 3. Major skills gap with no plan: 1.

EXAMPLE | Skills check

The bank's platform team runs GPU workloads for analytics but has not fine-tuned a language model before; two engineers will be upskilled during the pilot. SLM 3. The API option needs mainly prompt and evaluation skills the team already has: LLM 4.

Factor 5 scoring table

#	Criterion	SLM (1–5)	LLM (1–5)	Notes
21	Compatible with existing apps, APIs and databases			
22	Model update and versioning process defined			
23	Vendor lock-in and provider-dependency risk assessed			
24	Fallback plan defined if the model or API fails			

25	Internal team has the skills to run and maintain it			
	Factor average (sum ÷ 5)			

Scorecard & Decision Guide

Once all 25 criteria are scored, convert them into one weighted score per option. The weighting makes sure the factors that matter most for model selection have the most influence on the result.

How to calculate the score

1. **Average each factor.** Add the five criterion scores for the factor and divide by 5. This gives a factor average between 1 and 5.
2. **Convert to weighted points.** Weighted points = (factor average ÷ 5) × factor weight. A perfect 5.0 on Task Complexity earns the full 25 points.
3. **Add the weighted points** for all five factors. The maximum is 100.

Your weighted scorecard

Factor	Weight	SLM: avg → points	LLM: avg → points
Task Complexity	25%		
Cost & Infrastructure	20%		
Privacy & Security	20%		
Performance & Scalability	20%		
Integration & Maintenance	15%		
Total (out of 100)	**100%**		

Worked example: support email classification

Here are the running example's scores from all five factors in the weighted scorecard:

Factor	Weight	SLM: avg → points	LLM: avg → points
Task Complexity	25%	4.8 → 24.0	4.4 → 22.0
Cost & Infrastructure	20%	4.0 → 16.0	3.4 → 13.6
Privacy & Security	20%	5.0 → 20.0	3.6 → 14.4
Performance & Scalability	20%	4.6 → 18.4	3.6 → 14.4
Integration & Maintenance	15%	4.4 → 13.2	4.0 → 12.0
Total (out of 100)	**100%**	**91.6**	**76.4**

RESULT | SLM leads by 15.2 points

The SLM scores **91.6** and the frontier LLM scores **76.4**. The margin is more than 10 points, so the decision guide recommends the **SLM** for this use case.

The result makes sense: the task is narrow, volume is high, the data is highly sensitive and outputs can be checked automatically. The LLM's advantages (no training effort, easier integration) do not outweigh its higher running cost and the extra compliance work.

Decision guide

Result	Recommendation	What to do next
SLM leads by 10+ points	Go with a Small Language Model	Plan fine-tuning data, hosting and monitoring. Keep an LLM fallback for edge cases if Factor 1 showed any weak spots.
LLM leads by 10+ points	Go with a Frontier LLM	Negotiate enterprise terms (retention, region, rate limits), add an abstraction layer and set up cost monitoring from day one.
Within 10 points	Consider a hybrid architecture	Use the SLM for routine, high-volume requests and route complex or low-confidence requests to the LLM. See the hybrid section below.

Red flags that override the score

A weighted score is a guide, not a verdict. Some conditions should decide the outcome on their own, whatever the total says:

- **Highly sensitive data that cannot leave your infrastructure** → choose a local or private model, even if a cloud LLM scores higher.

- **Multi-step reasoning across varied sources** (two or more SLM scores of 1–2 in Factor 1) → choose a frontier LLM, or a hybrid where the LLM handles reasoning.
- **Very high volume, repetitive requests** → choose an SLM, or a hybrid with the SLM as the first layer, to keep costs under control.
- **Any compliance criterion (12 or 15) scored 1** → that option cannot go live until the issue is resolved.
- **No one to own a self-hosted model** (criterion 25 scored 1 with no plan) → choose a managed option until skills are in place.

When the answer is both: hybrid architecture

Many enterprises get the best result by combining the two. In a hybrid architecture, a small model handles the routine majority of requests cheaply and quickly, and a frontier LLM is used only where its extra capability is needed.

Example: a hybrid customer support flow

1. An **SLM** reads each incoming request and classifies its intent.

2. **Routine questions** are answered by the SLM using an approved knowledge base.

3. **Complex complaints, unusual requests or low-confidence cases** are forwarded to a **frontier LLM**.

4. Every response is **checked against business rules** before it reaches the customer.

If 80–90% of requests are routine, most traffic never reaches the expensive model, while customers with complex issues still get high-quality answers. To make this work, define clear routing rules, confidence thresholds, fallback paths and monitoring for both models.

Adjusting the weights for your organisation

The default weights suit most enterprise use cases. If your priorities differ, adjust them before scoring (not after) and keep the total at 100%.

Profile	Task	Cost	Privacy	Performance	Integration
Default	25%	20%	20%	20%	15%

Regulated industries (banking, healthcare, government)	25%	20%	25%	20%	10%
Cost-sensitive, high-volume operations	20%	30%	20%	20%	10%
Innovation / R&D use cases	35%	15%	15%	20%	15%

TIP | Make the result defensible

Record the reason for every score in the Notes column. When leadership or auditors ask why you chose a model, the completed checklist is your evidence, and it makes re-evaluation easy when prices, models or regulations change.

NEXT STEP

Lead AI Model Decisions With Confidence

This checklist helps you make one decision well. The **Certified Generative AI Professional** certification from GSDC helps you make every AI decision well, from understanding how language models work to bringing generative AI into real business processes responsibly.

Build practical skills in:

- **Generative AI foundations:** how LLMs and SLMs work, and where each fits.
- **Model evaluation and selection:** comparing capabilities, costs and risks for business use cases.
- **Enterprise implementation:** identifying use cases and integrating AI into workflows.
- **Responsible AI adoption:** governance, privacy and security considerations.

GSDC
Global Skill Development Council

CERTIFIED GENERATIVE AI PROFESSIONAL

LEARN DIRECTLY FROM GLOBAL PRACTITIONERS, GENERATIVE AI EXPERTS, AND INDUSTRY LEADERS WHO ARE SHAPING THE FUTURE OF GENERATIVE AI.



ABOUT GSDC CERTIFICATION

EBOOK

Extensive and exclusive Ebook created by world's experts to help you with understanding core concepts.

CREATED BY EXPERTS

GSDC certifications are created and authored by world's leading experts in the field.

EBOOK

Extensive and exclusive Ebook created by world's experts to help you with understanding core concepts.

LEARNING MATERIALS

Get access to learning materials such as videos, ebooks, templates, and practice exams, which will help you clear the certification exam.

LEARNING OBJECTIVE

- Understand how to fine-tune and apply large language models for specific tasks and business requirements.
- Evaluate ethical considerations in generative AI, including bias, fairness, data privacy, security, transparency, and responsible AI practices.

Enroll now with the code **LEARN20** To avail **20%** discount

Enroll Now

www.gsdcouncil.org