Flow (mental model)

1. User picks a value in each category (Model, Hosting/Residency, Data Strategy, Access/Governance, Integration Surface).
2. We compute 4 live metrics:

- **Cost** → € per 1K “effective tokens” (includes infra/API + pipeline overhead)
- **Security** → 0–100 (higher is safer)
- **Speed** → 0–100 (higher is faster)
- **Issues** → 0–100 risk (higher = more problems to expect: leaks, outages, staleness, false positives)

3. We map those into a **Score (0–100)** with weights. Use presets (Finance vs Security) or a custom weight slider.

Base + Deltas (the rule sheet)

All deltas are additive. Clamp Security/Speed/Issues to [0, 100]. Cost cannot go below €0.

Defaults (if you need them):

- **Cost** base: **€0.010 / 1K tokens**
- **Security** base: **50**
- **Speed** base: **50**
- **Issues** base: **50**

1) Model choice

Model	Cost Δ (€/1K)	Security Δ	Speed Δ	Issues Δ	Why
ChatGPT	+0.020	-5	+10	-5	Premium latency; US processing unless EU option; mature tooling reduces incidents
Claude	+0.012	+5	+8	-8	Strong guardrails; good latency; safer defaults
Gemini	+0.006	-2	+8	-5	Good multi-modal; US residency unless configured

Llama (self)	+0.004*	+10	-5	+8	Self-host infra cheap per-query, but ops risk/latency & drift
Gemma (self)	+0.003*	+8	-2	+6	Light model, similar trade-offs as Llama

*The “cost” here is **inference energy + amortized hardware**; no API fee but not free.

2) Hosting / Residency

Pick one from each sub-group.

a) Deployment target

Target	Cost Δ	Security Δ	Speed Δ	Issues Δ	Why
Public Cloud	+0.000	-10	+15	-5	Fast to scale, mature SRE, but data exits perimeter
Private Cloud	+0.004	+5	+5	+2	Dedicated tenancy; slower to scale
Local Server	+0.010	+15	-10	+10	Full control, but ops + outages risk and slower nets

b) Data residency

Residency	Cost Δ	Security Δ	Speed Δ	Issues Δ	Why
Germany	+0.002	+15	-2	-2	Strongest compliance
EU	+0.001	+10	0	-1	GDPR baseline
Global	+0.000	-10	+2	+2	Weakest residency guarantees

3) Data Strategy

Strategy	Cost Δ	Security Δ	Speed Δ	Issues Δ	Why

Internal Storage (static KB)	+0.000	+5	+10	+8	Cheap & fast, but stale = higher error risk
RAG (live retrieval)	+0.004	+3	-5	-8	Freshness lowers hallucinations; retrieval adds cost/latency
Fine-tuning	+0.010	-5	+8	-5	Great alignment; training data handling is risk; fast at runtime

4) Access / Governance

(You can pick multiple — apply all chosen deltas.)

Control	Cost Δ	Security Δ	Speed Δ	Issues Δ	Why
MFA	+0.002	+15	-2	-5	Fewer account takeovers
RBAC	+0.003	+12	-1	-6	Least privilege lowers incident scope
Moderation Filter	+0.002	+10	-3	-8	Catches PII/exfil but adds friction
Privacy-by-Design	+0.003	+15	-1	-6	No storage: strong privacy, less debugging
Free Use (no guardrails)	-0.004	-20	+2	+15	Cheap, fast onboarding; risky in prod

5) Integration Surface

Surface	Cost Δ	Security Δ	Speed Δ	Issues Δ	Why
Standalone Web App	+0.001	-2	+8	+2	Quick to adopt; broader attack surface
API Integration	+0.003	+3	+5	-3	Tighter control, observability
Mobile App	+0.004	-5	+5	+5	Endpoint sprawl risk

Microsoft 365	+0.003	+8	+3	-2	Mature controls within MS perimeter
Google Workspace	+0.002	-2	+5	-1	Good collab; US residency caveats
Internal Systems	+0.006	+12	-3	-4	Best data control; more integration effort

---

Putting it together (calculation)

1) Aggregate metrics

Cost(€/1K) = clamp_min(baseCost + Σ(costDeltas), 0)
Security(0-100) = clamp(baseSecurity + Σ(securityDeltas), 0, 100)
Speed(0-100) = clamp(baseSpeed + Σ(speedDeltas), 0, 100)
Issues(0-100) = clamp(baseIssues + Σ(issuesDeltas), 0, 100)

2) Normalize Cost to a 0–100 "CostScore" (higher is cheaper)
Pick bands that make sense for your org. Example:

CostScore = clamp(linear_map(Cost, from €0.002…€0.040, to 100…0), 0, 100)

3) Final Score (0–100)
Choose a profile or let users slide weights (they must sum to 1.0).

- **Finance profile** (optimize spend):
  `w = {CostScore: 0.40, Security: 0.30, Speed: 0.20, Issues: 0.10}`
- **Security profile** (optimize compliance):
  `w = {Security: 0.45, IssuesInv: 0.25, CostScore: 0.15, Speed: 0.15}`

Where `IssuesInv = 100 - Issues`.

Score = w_cost*CostScore + w_sec*Security + w_speed*Speed + w_issues*(100 - Issues)

Worked example (so you can sanity-check)

Selection

- Model: **Claude**
- Deployment: **Private Cloud**
- Residency: **EU**
- Data Strategy: **RAG**
- Governance: **MFA + RBAC + Moderation**
- Surface: **API Integration**

Step 1 — Sum deltas

- Model (Claude): **Cost +0.012**, Sec **+5**, Speed **+8**, Issues **-8**
- Deployment (Private Cloud): **+0.004**, **+5**, **+5**, **+2**
- Residency (EU): **+0.001**, **+10**, **0**, **-1**
- Data (RAG): **+0.004**, **+3**, **-5**, **-8**
- Governance (MFA + RBAC + Moderation):
    - MFA: **+0.002**, **+15**, **-2**, **-5**
    - RBAC: **+0.003**, **+12**, **-1**, **-6**
    - Moderation: **+0.002**, **+10**, **-3**, **-8**
- Surface (API): **+0.003**, **+3**, **+5**, **-3**

Totals

- **Cost Δ** = 0.012 + 0.004 + 0.001 + 0.004 + (0.002+0.003+0.002) + 0.003 = **0.031 €/1K**
- **Security Δ** = 5 + 5 + 10 + 3 + (15+12+10) + 3 = **63**
- **Speed Δ** = 8 + 5 + 0 - 5 + (-2-1-3) + 5 = **7**
- **Issues Δ** = -8 + 2 - 1 - 8 + (-5-6-8) - 3 = **-37**

Apply to bases (Cost base €0.010; others base 50)

- **Cost** = 0.010 + 0.031 = **€0.041 /1K**
- **Security** = clamp(50 + 63) = **100**
- **Speed** = clamp(50 + 7) = **57**
- **Issues** = clamp(50 - 37) = **13**

CostScore (map €0.002→100, €0.040→0):

- Cost €0.041 is just beyond the expensive end → **≈0**

Finance profile score

Score = 0.40*CostScore (≈0)
+ 0.30*Security (30.0)
+ 0.20*Speed (11.4)
+ 0.10*(100-Issues) = 0.10*87 = 8.7
= ~50.1

Security profile score

Score = 0.45*Security (45.0)
+ 0.25*(100-Issues) = 0.25*87 = 21.75
+ 0.15*CostScore (≈0)
+ 0.15*Speed (8.55)
= ~75.3