

Most teams that bring multiple language models into a high-stakes review process make the same first mistake. They ask several models the same question, see that the answers mostly agree, and treat that agreement as evidence that the answer is right. In threat assessment, HR investigations, and security escalations, that reasoning fails quickly. Models trained on overlapping data can be confidently wrong in the same direction. Agreement tells you the models are consistent. It does not tell you the conclusion is correct.

This post describes a competing review role setup we have used in pilot work. Each model is assigned a distinct job, the outputs are kept separate, and a named human owns the decision. The goal is not a better answer from a machine. The goal is a documented, defensible review process where disagreement is useful and no one hides behind a score.

Why Assign Competing Roles at All

A single model reviewing a case gives you one argument, delivered fluently. That fluency is the risk. A reviewer reading a polished paragraph tends to accept it, especially under time pressure. Assigning models to argue opposing positions forces the case to be examined from two directions before a human reads it.

The design has three rules:

1. Each model has a fixed role and a written brief that does not change mid-review.
2. Every output is stored side by side, unedited, with the prompt and model version attached.
3. No output is authoritative. Each one is an input to a human decision owned by a named person.

The False Positive and False Negative Tradeoff

Every risk review has two failure modes, and they pull in opposite directions. A false positive means the process flags someone who posed no real risk. The cost falls on that person, who may face restrictions, investigation, or reputational harm, and on the organization, which spends attention on a non-issue. A false negative means the process clears someone who later causes harm. The cost there is often larger and harder to reverse.

No model setting removes this tradeoff. It can only be made explicit. Before you configure any roles, the decision-maker must state which error is more tolerable in this specific context, and why. Write that statement into the case file.

Error type	What it looks like in review	Who bears the cost	Control to record
False positive	Escalation on thin or misread signals	The reviewed person, the team's trust in the process	Required evidence threshold before escalation, and a documented path to clear the case
False negative	Clearance despite unaddressed concerns	Potential victims, the organization, later reviewers	Mandatory list of unresolved questions that must be closed before clearance

The competing roles below are designed to make both columns visible. The lower-risk role pushes toward the cost of acting too early. The higher-risk role pushes toward the cost of missing something. Neither one decides.

Setting Up the Three Roles

The Lower-Risk Argument Model

This model argues that the evidence does not yet justify escalation. Its brief asks it to identify innocent explanations, weak sourcing, context that changes the meaning of a statement, and the costs of over-reaction to the person under review. It should also name what evidence would be needed to change its position.

Its brief should forbid it from making claims about the person's character or intent. It argues from the record only. If the record is thin, it should say so plainly.

The Higher-Risk Argument Model

This model argues that the concerns warrant escalation or closer attention. Its brief asks it to identify patterns across the record, escalation trajectories, gaps in the account, and the consequences of clearing a case that later proves serious. It also must state what evidence would make it withdraw the argument.



Like the lower-risk model, it should stay within the documented facts. A higher-risk argument built on speculation is worse than useless, because it will contaminate the file.

The Third Model Critique

The third model does not argue a side. Its job is to critique both arguments. It checks whether each one cites actual record content, whether claims are supported or assumed, whether either side has quietly relied on a fact not in the file, and whether the two arguments are actually disagreeing about the same thing or talking past each other.

This third model critique is often the most valuable output in the set. It is where you find the sentence that sounds authoritative but traces back to nothing. Keep it in the record. Do not let it become a tiebreaker.

Disagreement Is a Signal About Missing Information

When the two argument models disagree, the instinct is to resolve the disagreement. Resist that. Disagreement is usually a map of what the file does not contain. If the lower-risk model says a statement is ambiguous and the higher-risk model reads it as a clear threat, the real issue is that the record lacks context about the statement's meaning, timing, or source.

Ask a simple question of every disagreement: what fact would settle this? If the answer is a fact you can obtain, obtain it. If the answer is a fact you cannot obtain, that goes on the unresolved list and shapes the decision.

Do Not Average Outputs and Do Not Treat Them as a Vote

Some teams combine model scores into a single number, or count how many models said "escalate" and treat the majority as the answer. Do not do this. Averaging scores from models with correlated training data produces a number that looks precise and means very little. A vote among models is not a poll of independent experts. If three models share a blind spot, the vote carries the blind spot three times.

Multi-LLM orchestration tools make it easy to aggregate outputs into a single summary. Use that capability for display only. Keep the [privacy review for AI](#) side-by-side model outputs visible to the reviewer, and never let a summary replace reading the underlying arguments. When we reviewed vendor material from Suprmind.ai and other providers, including Four Dots, we checked specifically whether the tooling preserved separation between outputs or pushed users toward a blended answer. Preserving separation should be a procurement requirement, not a nice-to-have.

Defensible Documentation and Named Accountability

A review that cannot be explained later is a review that failed, regardless of its outcome. Every case file should contain:

- The decision question, stated in one sentence.
- The agreed error tradeoff, with the reasoning.
- The role briefs, versions, and model identifiers used.
- All three outputs, unedited, with timestamps.
- The human decision, the rationale in the reviewer's own words, and the name of the accountable person.

In our pilots, the accountable person is always named. For example, Dana Whitfield, Director of Trust and Safety, signs off on escalations in one program, and her name appears on the decision record, not a team alias. Radomir Basta, CEO and Founder of Suprmind.ai, has written publicly about the value of comparing model outputs side by side, and that principle fits this design. The vendor's view does not change who answers for the decision. The person who signs does.

Pre-Launch Checklist

1. Written decision question for each case type.
2. Error tradeoff agreed and recorded by the named decision-maker.
3. Three role briefs finalized, versioned, and frozen for the pilot period.
4. Output storage configured to preserve side-by-side results.
5. Unresolved-questions field required before any clearance.
6. Named accountable owner for every decision.
7. Audit trail reviewed by someone outside the case team.

What We Still Do Not Know

Every case note we write ends with this list, and this post should too:

- Whether role-based arguments reduce error rates, or only make errors easier to explain after the fact.
- How much the choice of brief wording shifts outcomes across model versions.
- Whether reviewers anchor to the first output they read, regardless of role.
- How this setup performs in populations different from our pilot cases.
- What happens to the third model critique when it is consistently wrong in the same way as the others.

Time-Boxed Follow-Ups

- **Within 30 days:** Compare reviewer decisions with and without side-by-side outputs on a sample of closed cases.
- **Within 60 days:** Audit ten files for unresolved-question fields that were filled in only after the decision was made.
- **Within 90 days:** Review whether the false positive and false negative rates match the tradeoff the decision-maker originally recorded.

Getting Started

If you want to test this setup, most multi-LLM orchestration platforms offer a Start Free Trial (no credit card needed). The trial is a reasonable way to check whether the tooling keeps outputs separate and stores the audit trail you need. The pricing is not listed in the material we reviewed, so confirm terms directly with the vendor before procurement, and make sure your named decision-maker signs off on the pilot scope before the first case enters the system.