Cloud Modernization / Developer Productivity / Platform Engineering

# How We Built an AI Matching Engine in 11 Weeks That Cut Influencer Shortlisting from a Month to Minutes

Our client is a Nordic-headquartered influencer marketing platform that plans and runs large-scale social campaigns for major consumer brands across beauty, retail, and on-demand delivery, spanning both Instagram and TikTok. Every campaign starts with a brief and ends in a shortlist of influencers to approach, and that translation was being done entirely by hand. A single brief could take up to a month to become a final shortlist, a hard ceiling on how much new business the company could take on.

They came to gravity9 to build an AI-based recommendation engine, running natively on MongoDB Atlas, that would hold every recommendation to a measurable standard by scoring it against who the company had actually hired in the past, and put the result in front of campaign managers as a live application rather than an offline scoring script.

## Utilized Technology Stack:

**Database & Vector Search:** MongoDB Atlas; Atlas Vector Search, Voyage-4-large embeddings

**Backend Framework:** Python (FastAPI)

**Frontend Framework:** React

**Agent Orchestration:** LangGraph

**Learned Ranking Mode:** LightGBM (LambdaRank)

**Reranking Model:** Voyage Rerank-2.5

**Large Language Models:** Anthropic Haiku 4.5, Sonnet 5 (brief parsing, intent classification, explanation generation)

**Real-Time Delivery:** Server-sent event streaming

## Review of the Challenges

The client needed a recommendation engine. Every influencer in its pool had worked with the company before, so every candidate already looked plausible. The job was to find the specific people who would genuinely get hired, in the right order, out of thousands of candidates across two platforms. That is a ranking problem, and it is harder than filtering: there is no shortcut for narrowing the field.

Four problems stood in the way. Quality had to be measurable rather than a matter of opinion, and the measurement itself had to be trustworthy; the client had around nine thousand completed influencer bookings across more than a thousand past campaigns, a real answer key showing exactly who had been hired before. No single scoring formula could serve every campaign, since an established brand with dozens of past collaborations and a brand new to the platform are not the same problem. A ranked list on its own was not enough, because campaign managers needed to interrogate a recommendation and defend it to their own clients. And human judgment had to become a usable signal: when a manager rejected a name, the stated reason mattered, but a comment like “not on brand” is not a rule that can safely be applied everywhere.

Before any of this existed, shortlisting took up to a month, and the bottleneck scaled with headcount rather than technology. Years of institutional history sat in the database, unused. Whoever knew a brand’s history best made the call, which meant no audit trail and no defence if a client pushed back, and the knowledge left the company whenever that person did. New brands, the ones that needed careful matching most, had the least history to draw on and were served worst by a relationship-driven process.

## Our Approach

We built every decision on evidence, not instinct. Before any signal earned a place in the ranking, we tested it against real hiring outcomes, and we discarded work when the data told us to.

The clearest example: brand relationship history looked like the strongest signal available, so we built a full knowledge graph of brands, brand families, influencers, and campaigns. On its own, it separated real hires from the rest better than almost any other signal. Combined with everything else the model already knew, it added almost nothing. We kept it out of the ranking algorithm entirely and put it to work elsewhere instead, explaining to a campaign manager exactly how a candidate connects to a brand.

We also kept the ranking itself deterministic. A conversational layer interprets what a user wants and narrates the result, but a single, fixed component produces the actual order. No language model can reshuffle a shortlist, which is what makes a recommendation reproducible, auditable, and measurable in the first place.

Because no two campaigns are alike, we classified every one by how much shared history existed with that brand and gave each category its own signal weighting: cold-start campaigns lean on content and audience fit, established relationships lean on track record. Every evaluation number was pulled from a single training and test split, written once and read by both sides, so results could never drift.

Finally, we treated a client rejection as a lesson, not a life sentence. The named profile never came back, but the stated reason was generalized into a broader description of what to avoid, deliberately excluding the influencer's content niche so that one rejection could never blacklist an entire category of good candidates.

## Our Solution

The result is a five layer system: a data layer, an embedding layer, a ranking pipeline, an explainability layer, and a conversational application layer that turns a chat message into a live, streaming result.

Campaign briefs and influencer profiles are split into short, topic-coherent chunks rather than embedded as one long document, cutting the number of embeddings by roughly seven times while making each one more useful to match against.

The ranking pipeline runs in stages, each of which can be switched on, off, or made optional, and each built so a failure falls back to the previous ordering rather than crashing or corrupting a result. A language model reads the free text brief and extracts what matters for matching. The campaign is then classified as new, developing, or established, which decides how the ranking signals get weighted. Vector search runs a fast semantic first pass across the full candidate pool, and ten structural signals, from brand history to audience overlap to engagement rate, are combined with it through Reciprocal Rank Fusion, a method for merging different scoring scales without recalibrating each one by hand. The strongest candidates then go through a more accurate reranking model that compares the brief against each one directly, before a learned ranking model, trained on the client's own hire and no-hire history, makes the final call. That learned model turned out to be the single biggest lever in the whole pipeline. Client rejections quietly demote similar candidates elsewhere on the list, without ever removing anyone outright.

Every recommendation dispatches with an explanation: a plain-language reason, a breakdown of exactly which signals drove the score, and a confidence indicator. None of it can invent a fact the pipeline did not already compute. A conversational layer sits on top of all of this, so a campaign manager can run a search, ask why a specific person was picked, check whether a brief is complete enough to match well, or reject a suggestion, all through one chat interface, with results streaming in live rather than arriving in one blocking response.

## Subsequent Outcomes

On 200 campaigns the model never saw during training, the engine now recovers 67.88% of the influencers actually hired, up from a 64.28% baseline. Its Mean Reciprocal Rank is 0.877, meaning the first genuine hire typically lands at, or very near, the top of the shortlist.

The client reviewed that number and chose to stop there, deliberately. An early target had been set higher, against a much smaller candidate pool. Closing the remaining gap would have meant fitting the model harder to past hiring patterns, which would have made it worse at exactly the job that matters most: serving brands with no history yet. A well-measured, explainable 67.88% across the full candidate pool beat a higher number bought with overfitting.

The two biggest performance gains were both measured, not assumed. The learned ranking model added 8.35 percentage points of recall on its own. Widening the reranking window to cover more candidates added a further 7.9 points.

The client ended the engagement with considerably more than it contracted for. A REST API, a full web application, live streaming results, a visual explanation behind every recommendation, and a feedback loop that learns from rejected candidates were all delivered beyond the original scope, on top of a recommendation engine that now explains every one of its decisions in plain language, directly answering the "black box" concern the client raised at the outset.

## Client Feedback

This is impressive, we are really happy with performance;  such an improvement.

"Built on evidence, not instinct a measurable, explainable recommendation engine that puts genuine hires near the top."

**Visit our Insights page for more articles about emerging technology trends, the health Industry, interviews, and more!**