ALPHA
intelligence
LABS

Build, Assemble, or Buy:

A Blueprint for the AI-Native Investment Office

Ken Akoundi, Drake Paulson

Everything an investment office would have to define, build, staff, secure, adopt, and maintain to run its own AI-native investment intelligence platform, and what that tells you about the decision.

> *"Should you find yourself in a chronically leaking boat, energy devoted to changing vessels is likely to be more productive than energy devoted to patching leaks."*
>
> Warren Buffett, Berkshire Hathaway Shareholder Letter, 1985

© Alpha Intelligence Labs Inc. 2026

For discussion purposes only · alphalabs.ai · ken@alphalabs.ai

Contents

Why a blueprint

The ten layers

Before and after the build

The team it takes

What it costs

The decision

About the authors

Why a blueprint

AI models and techniques have now made it possible to create truly AI-native investment workflow and intelligence platforms. These platforms streamline document and data inputs, connect systems, automate actions, retain decisions, and generate meaningful insights, often with little or no manual intervention. The modern investment office needs a purpose-built investment intelligence platform to automate its workflows and to capture and leverage its own investment and organizational intelligence. The question for the chief investment officer (CIO) and their technology leaders is no longer whether the office needs one. It is whether to subscribe to an existing product or to build one in-house. Others try a simpler route: why not achieve the same result with an enterprise license for Copilot, ChatGPT, Claude, or Gemini? Investment offices have smart people, real budgets, and workflows no outsider knows better than they do, and AI tools have made application development far easier. So where is the challenge in building an in-house solution?

The honest answer is not that it cannot be done. It is that the scope is almost always underestimated. An AI-native investment intelligence platform is not a chatbot plus a few prompts. It is ten interlocking systems, each of which must be built, secured, audited, and maintained indefinitely. Around those systems sits a full software lifecycle (defining the problem, gathering requirements, driving adoption, and supporting the system after launch) that draws as heavily on the business team's time as on the technology team's.

This paper lays out those systems in enough detail to serve as a blueprint. If you decide to build or assemble, it is your project plan. If you decide to buy, it is your vendor due-diligence checklist. Either way, it should make the decision clearer.

What this blueprint builds

The system this blueprint describes is an AI-native, enterprise-level investment management workflow and intelligence platform, or in short, an investment intelligence platform. It is the system an investment office relies on to collect its information, run its core processes, and remember why it made the decisions it made.

It exists to solve the core problems that nearly every investment office shares:

- Document overload. Files arrive from GP portals, custodians, administrators, inboxes, and shared drives, and saving, naming, and filing them becomes someone's full-time job.
- Manual workflows. Most repeatable work, such as transaction processing and portfolio reporting, happens outside core systems and still depends on significant manual effort.
- Institutional amnesia. The context behind a manager relationship or an IC decision lives in individual inboxes, notebooks, and memories. When people leave, that context leaves with them.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

- The research backlog. Teams spend their time gathering and reconciling information rather than analyzing it, so the analysis that matters most is the analysis that gets deferred, or never done at all.
- Accountability. Few offices can trace outcomes back to the expectations and decisions behind them, so the lessons of each investment are rarely captured or applied to the next.

Solving these problems requires a single connected system: every source flowing into one place, every document understood, every manager, fund, and decision linked over time, and agents that run the office's standardized processes end to end. That is the scope this blueprint describes, and it is the scope most build and assembly estimates leave out.

First, why an enterprise AI license is not the build

Most offices now have an enterprise license for Copilot, ChatGPT, Claude, or Gemini. Those tools are excellent and worth using. But when it comes to comprehensive investment intelligence, they provide a model, not a system. Four gaps define what an office would still have to build around them:

- The data gap. Most of what defines an investment office (DDQs, LPAs, side letters, capital call notices, IC memos, manager correspondence) is private. No general model has seen it, and none will unless you build the pipes to bring it in.
- The access gap. Those documents live behind GP portals, SharePoint permissions, email inboxes, and legacy file shares. Connecting them safely is an engineering and governance project, not a setting.
- The understanding gap. A general model reads text but does not know that a commitment field in a capital call ties to a drawdown schedule, or that a changed DDQ answer is a signal. That domain logic has to be built.
- The analysis gap. A general model answers the question in front of it. It cannot connect what the office expected at the time of a decision with what actually happened years later, because that history was never captured in one place.

Nor will the large AI vendors close these gaps for you. The institutional investment market is roughly one million users, against hundreds of millions for general productivity tools. No hyperscaler roadmap is built for a rounding error. Everything that follows is what it takes to close those gaps.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

Build, assemble, or buy

When an investment office or its parent organization's IT team says "build" today, they rarely mean writing every component from scratch. They usually mean assembling: combining off-the-shelf software (a document collection service, SharePoint, OneNote, Outlook, a CRM, and an enterprise AI license) with their own code on top and in between. Where technology is run by a parent organization, its IT team often proposes exactly this, using tools the organization already licenses. It is a reasonable instinct, and for a narrow, clearly defined problem it can work.

The challenge is that assembly treats the symptoms, not the underlying problem. Each component solves one problem well: collecting documents, storing files, capturing notes, or answering questions. None of them was designed to share an entity model, a permission model, or an audit trail with the others. The office ends up with a collection of capable tools that cannot see one another, and the root problems of disconnected information and institutional amnesia remain unsolved. The integration between them becomes the real project, and it is never finished. Every vendor release, model update, and new data source can break the connections the team has built.

An assembled solution still requires most of the ten layers that follow. It simply distributes them across more vendors, more contracts, and more points of failure. Whichever path an office chooses, the blueprint applies: a builder writes each layer, an assembler stitches the layers together, and a buyer should expect a vendor to demonstrate each one working in production.

Can two developers get us 80% of the way?

With today's AI coding tools, a small team can build an impressive prototype in weeks: a chat interface over a folder of documents, a few connectors, and answers that look right. It is tempting to conclude that two developers can deliver most of the platform. They can deliver most of the demo. The demo is the easy part of the work. The rest appears only when real users, real data, and real risk arrive:

- Real data. Hundreds of sources that change without notice, scanned and duplicate files, and the same manager under several names.
- Defensible accuracy. "Usually right" is acceptable in a demo. A capital call figure that is wrong a few percent of the time is a fiduciary problem.
- Permissions and security. Separating functions, keeping the AI from summarizing what a user cannot open, and passing the parent organization's security review.
- Memory over time. Searching documents is easy. Linking managers, funds, decisions, and outcomes across years is the hardest layer, and a quick build usually skips it.
- Continuity. With two people, every enhancement waits in their queue, every model update needs their re-testing, and the departure of either one puts the system at risk.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

For a single, narrow workflow, two developers can deliver real value. The mistake is treating the prototype as most of the platform. Two developers can get an office 80% of the demo. The other 20% is where the platform actually lives.

The blueprint at a glance

Layer	What it must do	Why it is hard
1. Data connectors	Bring every source into one place, automatically	GP portals, custodians, and internal systems change without notice; every source is different
2. Document intelligence	Classify, extract, and validate every document type	Dozens of formats; errors are fiduciary errors
3. Institutional memory	Connect managers, funds, holdings, people, and decisions over time	The same manager appears under many names, and context must stay linked for years
4. Model layer	Use the right AI model for each task and keep it current	Models change every few months
5. Agents and workflows	Automate real investment-office work end to end	Each workflow is its own product, built from many coordinated agents
6. User experience	Let staff ask, review, and act with confidence	Trust requires citations and transparency
7. Permissions and access	Show each person only what their role allows	AI can leak what users cannot open
8. Security and compliance	Protect data and prove it to auditors	Controls must be documented, tested, and evidenced continuously
9. AI governance	Make every AI-assisted decision defensible	Boards and regulators now expect it
10. Operations and continuity	Keep it all running after launch	The work never ends, and builders leave

The ten layers describe what has to be built. They sit inside a full software lifecycle, and the stages outside the technical build are the ones most often underestimated:

Lifecycle stage	What it requires	Why it is hard
1. Define the problem	Identify the root problems and the outcomes the system must deliver	Teams tend to solve the symptoms they see rather than the causes beneath them

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

Lifecycle stage	What it requires	Why it is hard
2. Gather requirements	Translate investment, operations, and compliance workflows into specifications	It takes significant time from the business teams who know the work
3. Build	Deliver the ten layers	The scope described in this paper
4. Deploy and drive adoption	Migrate data, adjust processes, and train every user	Software that is not used consistently never reaches its potential
5. Support and maintain	Keep the system, its users, and its roadmap current	It requires both technical and business support, indefinitely

The ten layers

1. Data connectors

Nothing works until the office's information flows into one place without manual effort.

- GP portal collection. Automated log-in, multi-factor handling, navigation, and download across every manager portal, each with its own layout. Portals change without warning, so collection breaks and must be monitored and repaired.
- Email ingestion. Secure connection to shared and personal inboxes, with detection of which messages and attachments matter.
- Drives and file shares. SharePoint, Google Drive, and network shares, respecting the permissions already set on them.
- Calendars and CRM. Meetings, attendees, and relationship history.
- Notes and research systems. Research management platforms, note-taking tools, and the Word and Google Docs files where the team records its own views.
- Meeting transcripts. Transcripts and summaries from Zoom, Granola, and similar services, linked to the managers and funds discussed.
- Portfolio data. PMS, IBOR, and ABOR reports, administrator and custodian feeds across public and private holdings, benchmarks, and public filings.
- Other applications and data feeds. Accounting, risk, and performance systems, along with the spreadsheets and one-off data sets every office accumulates.
- Scheduling and recovery. Run collection on a schedule, retry failures, alert when a source goes quiet, and keep a log of every run.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

2. Document intelligence

Raw files are not knowledge. Each one has to be recognized, read, and checked.

- Classification. Tell a capital call from a distribution notice, a quarterly report, a K-1, an LPA, a side letter, or a DDQ, automatically.
- Extraction. Pull the fields that matter (amounts, dates, fund, vehicle, NAV, fees) into structured data.
- Validation. Check extracted values against what is expected, flag anomalies, and route exceptions to a person.
- Hard cases. Scanned PDFs, 100-page documents, tables, and comparing a document against its prior version to catch what changed.

3. Institutional memory

This is the core of the value and the hardest layer to build. It turns documents into a connected, searchable history of the office.

- Entity model. Managers, funds, vehicles, holdings, contacts, meetings, and decisions, and how they relate.
- Entity resolution. Recognize that “ABC Partners,” “ABC Capital Partners LP,” and an email domain are the same manager.
- Internal commentary. The office’s own notes, meeting discussions, and investment views, linked to the managers and funds they concern. Not just the calendar entry for a manager meeting, but what was actually discussed. This is often the most valuable context an office has, and the most scattered.
- History and timeline. What a manager said two years ago, how a position evolved, why the IC made a decision.
- Retrieval. Find the right passage across thousands of documents and return it with a citation to the source.

4. Model layer

No single AI model is best at everything, and the leader changes every few months.

- Multi-model access. Contracts and integrations with several model providers, each vetted for security.
- Routing. Send each task (reasoning, extraction, search, summarization) to the model that does it best, balancing accuracy, speed, and cost.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

- Evaluation. A benchmark suite built on real investment-office tasks, so a new model can be tested before it touches production.
- Regression testing. Every model update can silently change behavior. Each one needs to be re-tested across every workflow.
- Fallback. A second provider ready when the first one has an outage.
- Open-weight models. The biggest unknown. Models such as Llama, Mistral, and Qwen are improving fast and can lower cost or keep data entirely in-house. But each one must be vetted for security, origin, and licensing terms, then hosted, monitored, and updated by the office itself. Ignoring them risks overpaying; adopting them adds a layer of work.

5. Agents and workflows

This is what staff actually see as value. Each workflow below is a product in its own right, with its own prompts, tools, rules, and testing:

- Document collection and filing across all managers and sources.
- Manager inbox triage: detect, classify, score, and route inbound manager opportunities.
- Due diligence support: DDQ review, consistency checks against prior answers, document comparison.
- Manager selection: structured comparison of a shortlist and a defensible decision record.
- Portfolio and manager monitoring: track holdings, exposures, and manager performance against expectations, and flag drift, style changes, and outliers.
- Risk and liquidity monitoring: watch concentration, liquidity, and policy limits, and forecast capital calls and distributions to plan commitment pacing.
- Capital call and distribution processing with reconciliation against expected schedules.
- Fee and expense validation: check management fees, carry, and expenses against LPA and side letter terms.
- Meeting briefs that pull together everything known before a manager meeting.
- Board and committee materials assembled from source data, with every figure traceable to its origin.
- Portfolio and allocation analysis, including strategic asset allocation support.
- Decision and outcome analysis: correlations and trends that compare recorded expectations with actual results.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

Behind each workflow sits more than one agent. A single task, such as processing a capital call, is broken into steps handled by specialized subagents: one classifies the document, another extracts the figures, a third reconciles them against the expected schedule, and a fourth checks the result before anyone sees it. An orchestration layer decides which agents run, in what order, with what context and memory, and what happens when one fails. Building that layer, and keeping dozens of agents working together as models change, is a system in its own right.

Every agent also needs guardrails: human review where judgment is required, confidence thresholds, a check that each answer is complete, and a log of every step it took.

6. User experience

Staff will only rely on a system they trust, and trust comes from seeing where every answer came from.

- Functional workspaces. Investment office activities generally fall into three areas: portfolio-related work (holdings, exposures, commitments, and cash flows), manager-related work (relationships, documents, meetings, and decisions), and research-related work (sourcing and evaluating new opportunities). Staff should work in views organized around those three areas, not around where data is stored. The memory layer is the foundation underpinning all three. If each area draws on its own data instead of a shared memory, the office is back to disconnected tools.
- Grounded answers. General models fill gaps with plausible guesses. Answers drawn from the office's own documents, each cited to its source, sharply reduce that risk but do not remove it. The system must show its evidence, say plainly when the evidence is not there rather than guessing, and make every answer easy to check.
- A conversational interface where staff ask questions in plain language and get answers with sources, tables, and suggested next steps.
- Review screens where people approve, correct, or reject what an agent produced.
- Tasks and notifications so nothing an agent flags gets lost.
- Sharing of answers and analyses that respects each viewer's permissions.

7. Permissions and access

In an AI system, permissions are harder than in ordinary software, because the AI can summarize documents a user was never allowed to open.

- Access by function. Separate rights for the investment team, operations, compliance, and leadership, set during onboarding.
- Granular rights. View, contribute, or own, set separately for documents, funds and holdings, managers, connections, and agents.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

- Permission-aware AI. Search, answers, and the knowledge layer must draw only on what the asking user is allowed to see.
- Governance. A standard model so access does not drift, audit of who changed what, and clean onboarding and offboarding.

8. Security and compliance

An AI system concentrates an office's most sensitive information in one place, so it must be secured at least as well as the systems it draws from.

- Encryption in transit and at rest, and encrypted backups across regions with tested recovery.
- Identity with single sign-on, multi-factor authentication, role-based access, and regular access reviews.
- Infrastructure security: network controls, intrusion detection, vulnerability scanning, and annual external penetration testing.
- Model-provider risk: contractual confirmation that providers do not train on or retain the office's data.
- AI-specific threats. Documents and emails can carry hidden instructions that try to direct an agent to leak data or take actions (known as prompt injection). Every agent that reads outside content needs defenses and testing against it.
- Incident response with a documented, tested plan.
- Audit-ready controls. Policies, evidence collection, and independent testing that satisfy the office's auditors and its institution's security program. A vendor serving outside clients provides a SOC 2 Type II report as that evidence. An internal system is not subject to SOC 2, but it still must meet an equivalent standard through the office's own controls and audits.

9. AI governance

Some of these obligations apply whether an office builds, assembles, or buys. Every office using AI needs a policy, an audit trail, and a view on regulation. The difference is who builds and maintains the tooling that supports them.

- A board-approved AI policy with risk tiers for different kinds of AI use.
- Explainability: show the rationale and confidence behind each AI output.
- An audit trail of every AI-assisted decision and who approved it.
- Regulatory tracking as AI rules evolve across jurisdictions.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

10. Operations and continuity

Launch is the beginning of the work, not the end. A system the office depends on daily must be kept running, current, and understood long after the people who built it have moved on.

- Monitoring and on-call for collection failures, model issues, and outages.
- Ongoing maintenance after every model release, portal change, and new document format.
- Documentation and knowledge transfer, so the system survives the departure of the people who built it.
- User support and training for every new hire and every new workflow.

Before and after the build

The ten layers are the technical core, but they are rarely where internal projects stall. Projects stall in the stages before and after the build, and those stages draw on the one resource an investment office cannot easily add: its business team's time.

Define the problem

Most internal projects begin with a solution rather than a problem. A team sees a symptom (documents filed by hand, a slow DDQ review, a missed change in a manager's answers) and builds a tool to address it. The root cause is usually broader: disconnected information and institutional amnesia. Defining the problem correctly is the most important step, and it is the one internal projects most often rush or skip. A vendor that has worked across many investment offices brings a clear point of view on where those root causes lie.

Gather requirements

Every workflow in this blueprint must be specified by the people who do the work. Investment staff, operations, and compliance must each describe their processes, edge cases, and exceptions, then review and test what is built. Most investment offices are already short-staffed, so that time comes directly from evaluating managers, running operations, and making investment decisions. It is a real cost, and it recurs with every new workflow and every enhancement.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

Deploy and drive adoption

Most software never reaches its potential because it never makes it through the adoption curve. Processes around the system must change, data has to be migrated, and every user has to be trained and supported until consistent use becomes habit. Adoption is required whether an office builds, assembles, or buys. The difference is that a vendor brings a tested rollout playbook from many deployments, while an internal team writes its playbook for the first time.

Support and maintain

Support is both technical and business. Beyond the operational work in Layer 10, someone must own the roadmap, prioritize enhancements, and retain the knowledge of how and why the system works. When the analysts and engineers who built it move on, that knowledge leaves with them.

The team it takes

A credible build of the full scope needs a dedicated team with these skills, along with committed time from the business:

Role	Responsibility
AI / ML engineers (2–3)	Model integration, routing, evaluation, agent design
Forward-deployed engineers (1–2)	Work alongside users to optimize prompts and agent instructions, tune each workflow, and turn real usage into improvements
Backend and data engineers (2–3)	Connectors, pipelines, document processing, memory layer
Frontend engineer (1)	Interface, review screens, sharing
DevOps and security engineer (1)	Infrastructure, monitoring, controls, audit evidence
QA engineer (1)	Regression testing across workflows and model updates
Product manager with investment expertise (1)	Translates office workflows into specifications
Designer (part-time)	Usability and trust in AI outputs
Business sponsor (part-time)	Owns priorities, outcomes, and adoption across the office
Business analysts (1–2)	Gather requirements, coordinate user acceptance testing, and train users

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

An assembly approach can reduce the engineering headcount, but it does not remove the need for these skills. It shifts some of them to integration and vendor management. The business roles remain in every scenario, and they draw on the same people the office relies on to run the portfolio.

Hiring takes time

Experienced AI engineers are among the most sought-after hires in the market. Recruiting a full team often takes six to nine months, and each new hire then needs time to learn the office's data, workflows, and terminology before contributing fully. The same is true of the business analysts and product roles. The clock on a build starts well before the first line of code.

What it costs

The cost of building is easy to underestimate because much of it never appears in the initial estimate. A full accounting includes:

- The team. The engineering, product, and business roles described above, along with recruiting fees and the cost of replacing people who leave.
- Infrastructure and models. Cloud infrastructure and AI model usage, both of which grow with data volume and adoption.
- Security, compliance, and audit. Controls, testing, evidence, and third-party software.
- Business team time. Specifying, testing, and adopting every workflow, drawn from investment and operations staff.
- Integration and vendor management. For an assembled approach, the licenses, contracts, and ongoing integration work across every component.

Each office's figures will differ, and an assembly approach changes the mix. What does not change is the scope. Once an office accounts for every layer and every stage of the lifecycle, the true cost of building becomes clear, and it is rarely the number in the original estimate. And the largest cost may not be money at all. It is the time the office spends without a working capability while its peers build institutional memory.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

The decision

No option wins on every dimension, and every path involves trade-offs. Building offers the most control. Buying offers the fastest and least risky path to the full scope. Assembling sits in between. The right choice depends on which trade-offs best serve the office's core goal.

	Build	Assemble	Buy purpose-built
Customization	Unlimited; built exactly to the office's needs	High within each tool, limited between them	Configurable, but within the vendor's design
Roadmap control	Fully owned by the office	Split across several vendors' roadmaps	Set by the vendor; the office influences but does not decide
Vendor dependence	None, but dependent on a few key people	Spread across several vendors	Concentrated in one vendor; mitigated by export and exit terms
Cost	Full team, infrastructure, and business time before value is delivered	Multiple licenses plus an integration team, indefinitely	A subscription and an implementation project
Scope	Limited to what the team can specify and build	Limited to what the components can share	Designed for the full problem from the start
Institutional memory	Must be designed and built from scratch	Every tool must still be connected to create a single source of memory, and that connection becomes the real project	A single memory layer by design
Talent	Months to recruit, then must retain specialized engineers	Integration and vendor management skills	The vendor's core team
Business team time	Heavy, for every workflow	Heavy, across several tools	Focused on configuration and adoption
Time to value	Measured in years	Fast for a first use case, slow for the full scope	Weeks to first value
Delivery risk	Sits with the office and its board	Sits with the office, spread across vendors	Sits with the vendor
Model currency	Locked to past choices unless re-engineered	Depends on each vendor's roadmap	Upgraded as better models ship

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

	Build	Assemble	Buy purpose-built
Security attestation	Built and evidenced through the office's own controls	A review for each vendor, plus the integrations	Already in place; request the report
Data ownership	Fully in-house	Spread across each vendor's terms	Set by contract; confirm ownership and export rights
Continuity	Depends on a few people staying	Depends on the people who built the integrations	A dedicated team, by design
Focus	Office runs a software project	Office runs an integration project	Office runs the portfolio

Internal builds and assemblies most often struggle for the same reasons: scope that grows as each problem reveals the next, enhancement and maintenance demands that never decline, turnover that takes both business and technical knowledge with it, and process changes that were never fully planned.

> Verdict: Building is possible, and it can work when the scope is tightly defined, the problem is unique to the office, and the solution needs little change over time. That is rarely true of a platform of this scope. An office that builds or assembles one has decided to become a software company: one that carries the delivery risk itself, invests years before seeing results, and depends on a handful of people staying. Its edge is allocating capital, not running software.

Data ownership and portability

Data ownership and portability is an important factor in the build versus buy decision. Building keeps every question of ownership, and every document, inside the office. That is a genuine advantage, but an internal build usually still sends those documents to an outside model provider, and it must still build the permissions that decide who inside the office sees what. Assembling spreads it across each vendor's terms. Buying puts it in a single contract, which is easier to review, but only if the office knows what to ask. Before signing, an office should confirm:

- Ownership. The office owns its data, its documents, and the outputs generated from them.
- No training. Its data is never used to train or improve models for anyone else, by the vendor or by its model providers.
- Retention. Model providers do not keep prompts or data beyond what is needed to process each request.
- Portability. Data, documents, and the structured records built from them (entities, relationships, history) can be exported in usable formats at any time.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

- Exit. On termination, data is returned and then deleted on a defined schedule, with written confirmation.
- A clear line between data and IP. The office should own everything that describes its portfolio: its data, documents, its own prompts, the outputs generated for it, and the structured records built from them. A vendor will reasonably own what it brings to every client: the software, agents and orchestration, connectors, prompt library and optimized prompts, workflow templates, and evaluation tools, licensed for as long as the subscription runs. The test is simple. Anything that describes the office's portfolio leaves with the office. Anything that makes the platform work for every office stays with the vendor.

Institutional memory is only an asset if the office can take it with it.

How to use this blueprint

- If you build or assemble, treat the ten layers and the full lifecycle as your project plan and budget for all of them, including the ones that never appear in a demo.
- If you buy, use the ten layers as your due-diligence checklist. Ask any vendor to show each one working in production: the connectors, the permission-aware answers, the audit report, the step-by-step agent logs, how they handle the next model release, and how you would export your data if you left.
- Either way, the technology team's role grows. Buying does not sideline the technology team, whether it sits inside the office or at the parent organization. It moves the team up the stack: owning security review, identity, data strategy, governance, vendor oversight, and integration with the office's other systems, and building the applications unique to the office on top of the platform. The team spends its time on what sets the office apart, not on rebuilding what every office needs.

The offices that move first will build a head start in institutional memory that peers cannot buy, replicate, or accelerate. Every year without a working capability is a year of memory not captured. The question is not whether to have this capability. It is whether to spend years building it.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only

About the authors

Ken Akoundi

Ken Akoundi is the founder and CEO of Alpha Intelligence Labs, which builds an AI-native investment intelligence platform for pensions, endowments, foundations, and family offices. His career has spanned J.P. Morgan, Deutsche Bank, and RiskMetrics Group, which he co-founded. He served as Chief Risk Officer of a fund of funds and, through Cordatius LLC, led more than 30 consulting engagements for institutional investment offices. An engineer by training, he holds a PhD in civil engineering from NYU Tandon School of Engineering and lectures regularly at universities. His earlier research on the board package production process at investment offices is also available on SSRN.

Drake Paulson

Drake Paulson is the founder of Paulson Advisory, which advises fintech companies serving the private markets and private equity firms investing in software. He has spent more than 25 years building and scaling software businesses across the investment ecosystem. He co-founded TopQ Software, a private equity performance analytics and due diligence platform acquired by eVestment, and then built and led eVestment Private Markets through its acquisition by Nasdaq. He later served as Chief Operating Officer of Bipsync, an investment workspace and research management platform, where he developed the company's AI governance framework and adoption strategy, and led client success and partnerships at Anduin. Earlier roles include Vitech Systems Group, DST International, Atlas Venture, JPMorgan Chase, and State Street. He holds a BA in Economics from the University of Connecticut.

About Alpha Intelligence Labs

Alpha Intelligence Labs builds Alphi, an AI-native investment intelligence platform for institutional investment offices. Alphi builds a memory layer from each office's own documents, using an ontology designed for the investment office, and connects documents, data, managers, funds, and decisions over time. Agents automate core workflows such as document collection and manager inbox triage, with answers cited to their sources. The company holds a SOC 2 Type II report and does not train models on client data. More at alphalabs.ai.

Disclosure: Ken Akoundi is the founder and CEO of Alpha Intelligence Labs, which provides an investment intelligence platform of the kind described in this paper. The views expressed are the authors' own.

© Alpha Intelligence Labs Inc. 2026 | For discussion purposes only



ALPHA intelligence LABS

© Alpha Intelligence Labs Inc. 2026. All rights reserved.
For discussion purposes only · alphalabs.ai