kyndryl | aws

Why AI Proofs of Concept Stall Before Production and what it takes to scale in the agentic era

A field guide to the three places AI projects die: cost, security, and focus. And the playbook for getting from demo to production and staying there.



Contents

PAGE

03 It’s not the technology. It’s the decisions around it

06 Cost-cutting has a floor. Growth doesn’t

09 The token and cloud cost cliff

13 No guardrails, no security, no policy

16 Building infrastructure instead of product

19 Budget the talent, not just the GPUs

22 Build an AI hub or lab, or borrow one

25 Move fast and make small mistakes

27 A 90-day action plan for CIOs

29 From AI lab to production, with Kyndryl and AWS





It’s not the technology. It’s the decisions around it

The challenge is no longer building an impressive AI proof of concept (POC). The challenge is deploying AI securely and reliably in production.

If you lead technology at any company, you have likely seen this pattern. A team builds an impressive generative AI or agent solution in a few weeks. Everyone is impressed, including the leadership. A budget gets approved. And then, months later, the project quietly dies or makes it into production but gets switched off the first time someone reads the bill.

You're not imagining this pattern and you're certainly not alone. The data shows just how widespread the problem has become. The 2025 Kyndryl Readiness Report finds 57% of senior leaders agree that their innovation efforts often stall after the proof-of-concept phase.

The three places AI projects die

In my experience, AI projects that struggle to reach production often encounter one or more of three issues. The rest of this guide is built around them.

01

Cost blindness
A POC tuned for a handful of users hits production traffic and the token and cloud resources bill multiplies until finance shuts it down.

02

No guardrails
The agent ships with no security, identity, or policy layer, so it can't be trusted with anything that matters.

03

Wrong focus
The team spends too much of its budget on infrastructure and runs out of money before the product is ready.



AI is not just another technology layer

Here is the part some boards still resist: companies should now consider a clear AI strategy aligned to their business priorities. Whether you sell insurance, cement, logistics, or healthcare, AI is no longer a side experiment owned by an innovation team. It is becoming part of how the company operates, makes decisions, serves customers, and competes.

This is the mindset shift that matters most. AI is not just another technology layer. It will increasingly reshape how work is done: what gets automated, which decisions can be delegated to AI, how teams are structured, and where people remain essential.

Imagine a mid-size insurer testing an AI assistant for claims. It summarizes each claim, and the team likes it. But the process itself is still the same, so the business impact is limited. The real value comes when the company redesigns the workflow around AI. An agent reads the claim, checks the policy and documents, sends complex cases to a person, and processes the rest automatically.

Claims eventually may be handled in hours instead of days, productivity increases, and the same team can manage much more work.

That is the difference between adding AI as a feature and using AI to redesign how the business operates.

Recommendations

- → Judge every AI initiative by whether it survives production and moves a number the business cares about, not by the demo.
- → Treat AI as a change to how work gets done: redesign the workflow around AI instead of adding it to the old process.
- → Check every proposed project against the three failure modes (cost, guardrails, and focus) before you fund it.

If you take one thing from this opening: stop judging AI by whether the demo is impressive. Start judging it by whether it survives production and moves a number the business cares about. The next nine sections are about how to make that happen.



Cost-cutting has a floor. Growth doesn’t

The most common strategic mistake I see CIOs make with AI is focusing entirely on saving money. Savings matter, but there is a limit to how much you can cut. The organizations creating the most value use AI to drive growth, then measure everything.

When AI lets you serve more customers, close deals faster, launch products sooner, or open a capability you simply couldn't before, the upside compounds. So, When deciding where to invest in AI, prioritize opportunities that create new value and drive growth not just those that reduce costs. Save where it's easy but build where it grows.

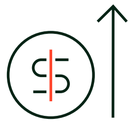
Before you build anything, find the use case

One of the most important early decisions is choosing the right first use case. This is where strategy actually happens: not in the model selection, but in deciding where AI brings the most value to your specific company. A great use case has four traits:

WHAT A FUNDABLE AI USE CASE LOOKS LIKE



Value
Does it improve something the business cares about, such as revenue, speed, capacity, or customer retention?



ROI clarity
Can you define a credible timeframe for **positive return after accounting for the real production costs?**



Data and readiness
Do you have the right data and access available today?



Success metrics
Can you define, before you start, exactly how you'll know it worked?

If a proposed project can't clearly answer all four questions, it isn't ready to be treated as a business initiative. It may still be worth exploring, but it should remain an experiment until the business case is clear.

Measure everything.
Set KPIs before you write code

It sounds obvious, but teams often skip it. They build an impressive demo, get positive feedback, and move forward without clearly defining what success should look like. Six months later, the project may be live, but no one can confidently say whether it improved the business.

Start with the business result you want to change. Define a small number of KPIs that leadership already understands and cares about.

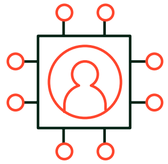
You should also establish a baseline before the AI solution goes live. If a process takes three days today, record that. If each employee handles 40 cases a week, record that. If a transaction costs US$20 today, record that. Without a starting point, it is very difficult to prove that AI made anything better.

And do not wait until the end of the project to look at the numbers. Build the measurement into the system from Day 1. The team should be able to see what is improving, what is getting worse, and whether the business value still supports the cost of running the solution.

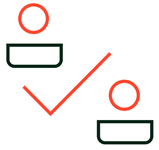
A STARTER KPI SET FOR ANY AI PROJECT



Business value
Revenue or pipeline influenced, throughout per employee, customer satisfaction, retention



Adoption
Active users, tasks completed by AI vs. handed to a human, repeat usage



Quality and trust
Accuracy / acceptance rate, hallucination, human-override rate, escalations



Unit economics
Cost per task, tokens per request, cost per active user, monthly run-rate vs. budget

Recommendations

- Pick one high-value use case with clear, measurable outcomes and take it to production before adding more.
- Write the ROI case with production costs included from day one, not the POC's bill.
- Define clear KPIs before you build and make sure the system tracks and reports them automatically.
- Use AI to drive growth and increase capacity, not just to reduce costs and headcount.



The token and cloud cost cliff

This is a frequently overlooked risk. A team builds a POC and tests it with one or a handful of users. The monthly cost looks trivial, so nobody worries about it. The project gets promoted to production. Then, several months later, when the cloud and model invoices finally catch up, leadership sees the real number and immediately shuts it down

A simple production example

Take a simple AI assistant on Amazon Bedrock that uses Claude Sonnet 5.5, priced at US$2 per million input tokens and US$10 per million output tokens at the time of writing. In development and testing, a few people use it. Each call sends the model about 2,000 tokens (instructions, documents, and chat history) and gets about 500 back, which costs about one cent. At 10,000 calls a month, the model bill is about US$90, and with cloud resources the whole thing costs about US$225 a month. At that level, the cost is easy to approve and easy to overlook.

Then it goes live. More people use it, so the number of calls grows six times. Conversations get longer and the assistant pulls in more documents, so each call carries about 2.5 times more tokens. Six times more calls, each 2.5 times bigger, means 15 times more tokens. In this illustrative scenario, cloud infrastructure is assumed to increase by about three times

ILLUSTRATIVE MONTHLY COST IN US$, DEV/TEST VS. PRODUCTION

Line Item	Dev/test	Production	Change
Model calls/month	10,000	60,000	x6
Tokens per call	~2,500	~6,250	×2.5
Tokens/month	25 million	375 million	×15
Model/token cost	$90	$1,350	×15
Cloud resources (compute, vector store, getaway, logging)	$135	$405	×3
Total/month	$225	$1,755	×7.8
Total/year	$2,700	~$21,000	×7.8

Illustrative example. Model cost uses Claude Sonnet 5.5 on Amazon Bedrock at US$2 per million input tokens and US$10 per million output tokens (on-demand, at the time of writing), with about 80% of each call's tokens as input and 20% as output. Cloud costs are illustrative and depend on your architecture. Prices change and vary by region; check current Amazon Bedrock pricing.

The monthly bill goes from about US$225 to about US$1,755, almost eight times more, and tokens grow from 40% of the bill to 77%. A project budgeted at under US$3,000 a year now costs about US$21,000. Across a portfolio of AI features, these costs can become material, which is why production economics should be modelled before scaling. The POC didn't lie; it just never represented production.

And this is still a small rollout. Every new user and every longer conversation adds tokens, so as usage keeps growing, the token bill keeps growing with it, much faster than the infrastructure bill.

Why tokens explode the way they do

Cloud costs grow the way most people expect: more users, more servers. Token costs are different. You pay for every word the model reads and every word it writes, and the model reads far more than the user types. A one-line question can carry pages of hidden text with it. Tokens grow because of:

- More users, longer conversations. Every new message resends the earlier conversation, so a chat gets more expensive with every turn.
- Bigger context. Instructions, retrieved documents, and chat history go with every request, and are often many times longer than the question itself.
- Retries. When an answer fails a check or is blocked by a guardrail, the model runs again, and you pay again.
- Agents. An agent plans, calls tools, checks its work, and talks to other agents, with a model call at every step. In Anthropic's reported research-system implementation, agents used about 4 times the tokens of a chat interaction and multi-agent systems about 15 times.

The principle is simple: token cost = number of calls × tokens per call × price per token. Production multiplies the first two at the same time, which can cause token spend to grow faster than expected as usage scales. This is the tokenomics of production AI: understanding how usage turns into cost per task before you scale.

TokenOps: Treat tokens like a budgeted resource

We use "TokenOps" here as shorthand for applying cost-management discipline to AI token consumption, much as FinOps does for cloud spend. The core idea is simple: tokens are a metered resource with a real price, so plan, measure, and control them the way you would compute or storage, before you scale, not after the invoice. "We'll figure out the cost later" is one of the biggest mistakes in an AI project. By the time the production bill arrives, the system is already built.

- Model routing. Send easy requests to small, cheap models; reserve the large model for the hard ones
- Caching. Cache repeated prompts, retrievals, and common answers so you don't pay to generate the same thing twice
- Context discipline. Send only the most relevant context and retrieve only the information the model needs
- Unit economics. Track cost per task and tokens per request on a live dashboard, make it a number the team sees daily
- Project production cost up front. Estimate production tokens at expected scale during the POC and put that number in the business case.

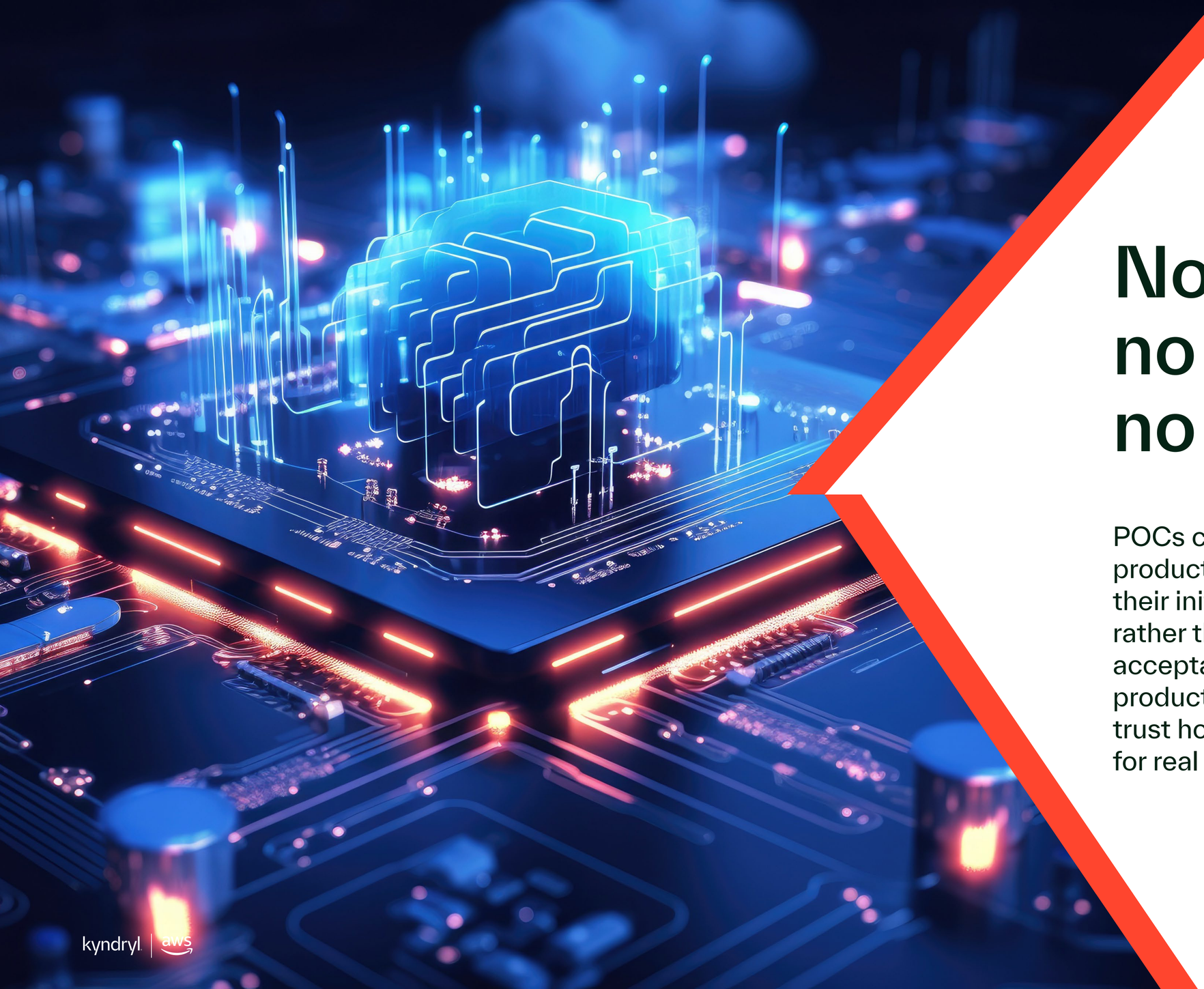
On AWS, smart routing can help control model costs

The biggest lever on token cost is simple: don't send every request to your most expensive model. Amazon Bedrock offers models of different sizes and prices in one place, and most everyday tasks, such as classifying a request, extracting fields, or summarizing a short text, don't need the most powerful one. At the time of writing, Claude Haiku 5.5 costs one-twentieth as much per token as Claude Sonnet 5.5.

In practice, put a gateway in front of your models whose main job is to reduce cost. It sends each request to the smallest model that can do the job well, answers repeated questions from a cache instead of paying for them again, and enforces a token budget for each team or application. I describe one way to build this with Amazon API Gateway, AWS Lambda, Amazon Bedrock, and a semantic cache in An AI Cost Optimization Gateway on AWS.

Recommendations

- Model production token and cloud costs at expected scale during the POC, and put that number in the business case.
- Route each request to the smallest model that does the job well, and cache repeated prompts and answers.
- Track cost per task and tokens per request on a live dashboard from day one.
- Set budgets and alerts for token spend, just as you do for cloud resources.



No guardrails, no security, no policy

POCs can be built with limited production-grade security because their initial purpose is experimentation rather than deployment. That may be acceptable in a sandbox, but not in production. If you cannot control and trust how the AI behaves, it is not ready for real users or sensitive data.

When a POC is built to impress, security is the first thing cut and the last thing added. The POC has no controls on what the model can say, no identity model for what it can access, and no policy for what it's allowed to do. Then someone wants to put it in front of customers or connect it to real systems, and the whole thing stalls, because nobody can answer the simplest risk questions.

Kyndryl's 2026 People Readiness Report (Page 25) found that governance maturity is not yet keeping pace with autonomy. An agent that can take actions, send emails, move data, call APIs, trigger workflows, is far more dangerous than a chatbot that only talks. The more autonomy you give it, the more it needs a security layer around it.

What "securing an agent" means

You're protecting against three things: the model behaving badly (toxic, off-policy, or hallucinated output), the model being manipulated (prompt injection, jailbreaks, data exfiltration), and the model doing more than it should (over-broad access and unchecked actions).

That requires six layers:

1 **Guardrails**
Help filter inputs and outputs and mitigate unsafe content, off-topic use, prompt-injection attempts, and leakage of sensitive data. Keep the model acting the way policy says it should.

2 **Data boundaries**
Control exactly what data the agent can read and where its outputs can go. Keep regulated and sensitive data on the right side of the line.

3 **Vector database security**
The knowledge base behind a RAG agent is a new store of sensitive data. Encrypt it, control who and what can query it, filter retrieval by the user's permissions, and check documents before ingestion so poisoned content or hidden prompt injection never reaches the model.

4 **Agent identity and authentication**
Every agent has its own identity, separate from the people it serves. It authenticates every tool call, acts on behalf of a user only within that user's permissions, never holds long-lived credentials in its code, and gets only the access the task needs.

5 **Policy as code**
Encode the rules, what's allowed, what needs a human, what's forbidden, as code that's versioned, reviewed, and enforced automatically.

6 **Agent actions and tools**
Define which tools, APIs, and other agents each agent may call, with which parameters, and which actions need human approval. Log every action so you can trace what the agent did and why.

The fast path: use AWS native services

You do not need to build this from scratch. AWS already provides native security controls for AI.

Amazon Bedrock Guardrails helps control what goes into and comes out of the model, including unsafe content and sensitive data. AWS IAM controls what each agent can access and what actions it can perform.

For agents, Amazon Bedrock AgentCore adds controls built for autonomy. AgentCore Identity gives each agent its own identity, keeps credentials out of agent code, and lets an agent act on behalf of a user only within that user's permissions. Policy in AgentCore, generally available since March 2026, checks every tool call the agent makes through AgentCore Gateway against rules written as code, outside the agent's own logic.

For the vector database, Amazon Bedrock Knowledge Bases encrypts stored data with AWS KMS keys and supports metadata filtering, so an agent retrieves only the documents the current user is allowed to see.

Using native controls can reduce the need to build equivalent capabilities from scratch and can support consistent management and auditability.
Treat guardrails, agent identity, policy, and vector database controls as a required part of every AI architecture, not something you add later.

Test it like real software, before production

You wouldn't launch a banking application without first putting it through rigorous security testing.
An agent is no different, and arguably needs more, because its behavior is dynamic and less predictable.
Two practices belong in every project before go-live:

Agent simulation testing. Test the agent with many realistic scenarios before users do. Make sure it can handle unexpected questions, edge cases, and attacks.

Penetration testing. Run a pen test the same way you would for any production application, now including AI-specific attacks like prompt injection and jailbreaking. Do it before production, not after an incident. On AWS, AWS Security Agent runs on-demand penetration tests. It reviews your design documents and code, identifies and tests security vulnerabilities, and can suggest fixes or create pull requests to address them.

Recommendations

- Make security review a mandatory gate for every AI project: no guardrails, no production
- Use cloud-native guardrails and identity controls by default. They are faster to deploy, easier to manage, and easier to audit
- Give every agent its own identity, least-privilege access, and a human approval step for high-impact actions
- Secure the vector database like any production data store: encrypt it, filter retrieval by user permissions, and check content before ingestion
- Run agent simulation and penetration testing before go-live, for example with AWS Security Agent



Building infrastructure instead of product

This one is easy to miss because it looks like good engineering. Teams move fast and invest heavily in monitoring, tracing, observability and infrastructure, but the agent itself is still only average. By the time they realize it, much of the budget is already gone.

I see talented AI engineers do this constantly. They're moving fast, they care about quality, so they pour their energy into observability, tracing, logging, monitoring, and the overall platform. All of it is genuinely good practice. But it isn't the product. Months later they have an elegant infrastructure, and an average agent that doesn't actually do its job well, and they've spent most of the AI budget getting there.

Today, building an agent is the easy part. Moving it to production, reliably, safely, affordably, is the whole job.

Buy the platform. Build the product.

Do not spend your engineering budget rebuilding capabilities that already exist. Use cloud platforms, managed services, and proven agentic AI frameworks for the common building blocks: security, guardrails, identity, orchestration, monitoring, and integrations.

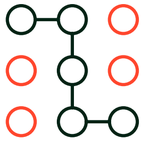
For example, on AWS, an enterprise can combine managed services such as Amazon Bedrock with an agentic AI framework such as Kyndryl's to provide much of the foundation needed to build and operate agents at production scale.

That allows the engineering team to focus on the parts that create business value: the use case, company data, business logic, agent quality, and user experience.

WHERE YOUR TEAM'S TIME SHOULD GO



Model hosting and inference
The business workflow you are improving



Agent orchestration runtime
Domain logic, business rules, edge cases



Guardrails, identity, policy enforcement
Your data, context and retrieval quality



Observability, tracing, logging
The user experience and trust around the agent

Most companies want AI to work with their own business data. That is where retrieval-augmented generation (RAG) becomes important: it allows an AI application to retrieve relevant information from company documents and data before generating an answer.

On AWS, services such as Amazon Bedrock Knowledge Bases can provide much of this capability. Instead of spending months building custom data pipelines, search components, and vector databases, teams can focus on the business use case and the user experience.

The principle is simple: use managed infrastructure for the common building blocks and spend your engineering budget on what makes your AI solution valuable to the business.

Recommendations

- Use managed services for the common building blocks: model hosting, orchestration, guardrails, identity, and observability.
- Spend engineering time on the workflow, domain logic, data quality, and user experience.
- Start with managed retrieval, such as Amazon Bedrock Knowledge Bases, before building custom data pipelines.
- Review where the budget is going; if most of it funds plumbing, rebalance toward the product.



Budget the talent, not just the GPUs

Large companies may spend extensively on AI infrastructure such as GPUs, but not enough on the people needed to make it successful. That imbalance can hurt a project more than the technology itself.

I see this often in large technology companies. Kyndryl's 2026 People Readiness Report found 52% of business leaders said it has become more difficult over the past 12 months to find employees with the right skills for their company's AI strategy.

The usual assumption is: "Our current team can figure it out."

Sometimes they can. But moving AI from a POC into production requires experience with things like cost, security, reliability, monitoring, data quality, and user adoption.

If the team does not have that experience, the company may only discover the gap after months of work and a large part of the budget has already been spent.

If your team will do it alone, at least get a guide

If you want to build your in-house team, that can work but do not expect them to figure everything out on their own. Bring in a strategic partner that has real experience taking AI systems from POC to production and operating them on a production scale.

If you are building on AWS, there is another simple way to narrow the field: check the partner's AWS Specializations and Competencies, especially the AWS AI Competency and its Agentic AI categories. AWS validates these partners through technical reviews and customer success examples, so the designation is a useful signal that the partner has demonstrated real experience.

And one more thing: ask to work with the actual team that earned that experience. A company may have the competency, but you still want the architects and engineers who built those production systems involved in your project.



Experienced AI talent requires investment

Experienced AI engineers and architects are expensive, but this is one of the areas where paying for experience can save you much more later. Someone who has already taken AI agents into production has seen the problems that usually appear after the POC: rising costs, security gaps, poor data quality, reliability issues, weak monitoring, and systems that need to be rebuilt because the first design was not ready for real users.

At the same time, you cannot solve everything by hiring a few expensive experts. Kyndryl's 2026 People Readiness Report indicates that many organizations are finding it harder to source the skills needed for their AI strategy. Organizations should also avoid making their AI programs permanently dependent on a small group of external specialists.

The better approach is to build a mix:

Bring in a small core of experienced AI engineers and architects who can set the direction, make the important architecture decisions, and help the first production projects succeed.

Train your existing engineers while they work with them. Your current teams already understand your systems, customers, data, and business processes. Add strong AI skills to that knowledge, and you build a much stronger long-term capability.

Recommendations

- Budget for experienced people alongside infrastructure. They are a production requirement, not a luxury.
- Pair a small core of experienced AI engineers and architects with your existing teams, and train in the flow of real projects.
- Choose partners with production experience and validated credentials, such as the AWS AI Competency.
- Ask to work with the people who built those production systems, not just the company that earned the badge.



Build an AI hub or lab, or borrow one

Every company needs a place to experiment safely and a way to be guided through the first hard project. You don’t have to own all of it. You do have to ship one thing to production, because the first win makes everything after it possible.

AI capability does not build itself.

Companies need a structured way to move from ideas to real production.

This is where an AI Hub or AI Lab can help. The goal is to bring business and technology teams together, identify the right use cases, prioritize the ones that can create real value, and help teams move from an idea to a working solution.

Customers can use dedicated workshops to explore their business challenges, understand where AI
can help, compare possible use cases, and decide which ones are worth investing in. From there, the team can quickly test the idea, validate the technology and business value, and prepare the solution for production.

The lab environment also can give customers access to experienced architects, engineers, security specialists, and AI experts, along with proven tools, patterns, and lessons learned from other projects.

At Kyndryl's AI Innovation Labs, our experts collaborate closely with customers to turn AI ideas into practical business solutions. By identifying high-value use cases, designing intelligent workflows, and developing working prototypes, we help organizations bridge the gap between experimentation and production, enabling them to scale AI-driven solutions across their operations and deliver measurable business outcomes.

Kyndryl AI Innovation Labs are open in Dallas, Liverpool, and Luxembourg. A two-day, in-person session is designed to help customers progress from problem definition towards a working prototype and a roadmap to production. In the UK, Kyndryl's AI hub in Liverpool, the Kyndryl AI Innovation Lab Liverpool in the Royal Liver Building, combines the lab with Kyndryl Consult advisory and co-creation services and a growing team of AI and software engineering specialists. Chapter 10 explains how to work with the labs.



The trust sequence: one win can enable the next

Here is the most important move in the whole playbook, and it is simple: get one AI project into production.

Pick one use case with clear business value and measurable results. But before you build anything, map the workflow as it exists today. Understand where time is being lost, where decisions slow down, where people repeat manual work, and where errors create friction.

Then decide which activities should be handled by AI or agents and which should remain human-led. Identify where human review, approval, or escalation is still required, and design the future workflow around both trust and adoption. The goal is not simply to automate a task. It is to create a better way of working that people will use.

Only then should you build the solution and take it all the way from idea to production. If you need help, bring in experienced people who have done it before. Make sure you cover the things that are often missed: production costs, security and guardrails, KPIs, real users, and ongoing operations.

Once that first project is live and working, it can begin to build trust in the process.

Leadership can see that AI is delivering real value, not just a good demo. The technology team has learned what it takes to move from POC to production. Finance has seen the real cost and the real return. Security has seen how the solution behaves in a real environment. And the business now has proof that the approach works.

That first project becomes the model for what comes next. Subsequent projects can benefit from the experience, reusable patterns and confidence created by the first production deployment.

THE TRUST SEQUENCE

First win
One high-value use case goes all the way to production, with real users and the KPIs you defined up front.

Result: proof, not a demo

→

Trust earned
Leadership sees business value, finance sees the real cost and return, and security sees how it behaves in production.

Result: permission to scale

→

Compounding
You reuse the patterns, guardrails, and cost model, so the next project is faster to approve, build, and ship.

Result: a repeatable path to production

Recommendations

- Set up an AI hub or lab, your own or a partner's, where business and technology teams prioritize use cases together.
- Map the current workflow and decide which steps AI should handle and which stay human-led before you build.
- Take one use case all the way to production, then reuse its patterns, lessons, and trust for the next one.



Move fast and make small mistakes

The AI space moves every month with new models, new agent frameworks and new services that removes work teams previously had to build themselves.

Something that was difficult a few months ago may now be available as a standard feature. A case that was too expensive earlier in the year may suddenly make sense because models became cheaper, faster, or more efficient.

That is why every quarter, companies should revisit ideas they rejected in the past. A use case that required too many tokens six months ago may now work with a smaller model at a much lower cost. New features such as caching can also reduce how many tokens you use. In other words, the cost of running AI can improve quickly, but only if you keep reviewing what is possible.

This is why waiting can be risky. If you wait for AI to fully mature before you start, you may find yourself behind companies that have already spent time learning, testing, and building real experience. They already know what works, what does not, how much it costs, and how to move projects into production. To stay competitive, companies need to start using AI as part of the way the business works. Not endless POCs. Not isolated demos. Real use cases, real users, and real production systems.

Mistakes are part of the price. Keep them small

Here's the permission slip every CIO needs to hear and to give: You are going to make mistakes, and that's okay. There is no version of moving fast in a fast-moving field without missteps. The discipline is keeping the missteps small and learning as you go.

Recommendations

- Build a habit of re-evaluating models and costs every quarter. What was impossible may now be affordable.
- Treat small, contained failures as the cost of learning, and review them openly.
- Don't wait for the field to settle. It won't. Engage now and learn in motion.



A 90-day action plan for CIOs

How to get from “we should do AI” to one project in production

DAYS 0 – 30
Choose and Frame

- Pick one use case with clear business value and measurable results, focused on growth or increasing capacity, not only on cutting costs.
- Write the business case with production token and cloud costs estimated, not POC costs.
- Define KPIs up front, business value, adoption, quality, and unit economics (cost per task).
- Decide whether to build in-house or work with a partner. If you build internally, bring in an experienced partner or advisor who has already taken AI systems into production
- Stand up (or borrow) an AI lab environment to work in safely, for example a two-day session at a Kyndryl AI Innovation Lab.

DAYS 30 – 60
Build with Discipline

- Build the agent focused on the workflow and domain, buy the platform, guardrails, and observability.
- Track token costs from day one: use the right model for each task, cache repeated answers, keep prompts focused, set cost limits, and monitor the cost of each task.
- Add the security layer now: cloud-native guardrails, a dedicated identity for each agent with least-privilege access, policy as code, vector database access controls, and data boundaries.
- Compare your original production cost estimate with the real costs you are seeing.

DAYS 60 – 90
Harden and Ship

- Run agent simulation testing and a penetration test before go-live.
- Put a human-approval step on any high-impact action.
- Launch to real users; track the KPIs you defined on day one.
- Capture lessons, cost, quality, what broke, and feed them into project number two.
- Use the win to build trust with leadership and finance, then scale to the next use case.



From AI lab to production, with Kyndryl and AWS

Everything in this guide comes down to one move: get the first AI project into production, then do it again. That is the work Kyndryl does with customers every day, together with AWS.

Kyndryl helps enterprises design, build, secure, and run AI on AWS, from the first use-case workshop to an agent that is live, governed, in production. We bring decades of experience running mission-critical systems to exactly the problems this guide describes: production costs, security and guardrails, focus on business value, and the skills to sustain it.

Kyndryl and AWS: a partnership built for agentic AI

Kyndryl is an AWS Premier Tier Services Partner. In June 2026, Kyndryl and AWS expanded their multi-year Strategic Collaboration Agreement to help customers adopt, deploy, and scale agentic AI while modernizing and operating mission-critical workloads on AWS. Under that agreement, the two companies are co-developing industry-specific agentic AI modernization blueprints and offerings, and AWS is investing in Kyndryl's talent development, joint solution engineering, and AI specialization.

WHAT THE PARTNERSHIP MEANS FOR YOU

- **Validated AI expertise.** Kyndryl holds the AWS AI Competency, including Agentic AI, alongside AWS Competencies such as Digital Sovereignty. It is the same signal Chapter 06 recommends you look for in a partner.
- **Depth on AWS.** A global AWS business and technical team with more than 11,000 AWS certifications, including AWS Ambassadors who build and speak on generative and agentic AI.
- **A production foundation, not a blank page.** The Kyndryl Agentic AI Framework, combined with AWS services such as Amazon Bedrock, Amazon Bedrock Guardrails, and Amazon Bedrock Knowledge Bases, provides orchestration, governance, and human oversight, so your engineers spend their time on the product, as Chapter 05 recommends.
- **Advice that starts with value.** Kyndryl Consult helps you map workflows, prioritize use cases, and write the business case with production costs and KPIs included from day one.

Spotlight: the Kyndryl AI hub in Liverpool

The Kyndryl AI Innovation Lab Liverpool, in the city's iconic Royal Liver Building, is Kyndryl's AI hub for customers in the UK and beyond. Led by Kyndryl Consult, it brings together advisory and co-creation services, a Kyndryl Vital studio for designer-led collaboration on real business challenges, and a team of AI and software engineering specialists focused on AI and data science.

HOW AN ENGAGEMENT RUNS

Find the use case
Workshops with Kyndryl Consult to map the workflow, prioritize use cases, and write the business case with production costs and KPIs.

Prove it in the lab
a focused engagement at a Kyndryl AI Innovation Lab designed to develop a working prototype and a roadmap to production.

Ship it and run it
Build on AWS with guardrails, least-privilege identity, and TokenOps cost controls; operate it, then scale to the next use case.



kyndryl | aws

The bottom line

In our experience, AI projects that struggle to reach production often do so because of decisions made around the technology: teams focus on impressive demos instead of real business value, underestimate production costs, move too fast without enough security, spend too much on infrastructure, and do not invest enough in the people who can make the project work.

Every one of these is a decision you can control.

The companies getting real value from AI are not always the ones using the newest or most powerful models. They are the ones choosing the right problems to solve, knowing what success looks like, understanding the real cost of running AI in production, building security in from the start, and keeping the team focused on business results.

Start with one project. Get it into production. Learn from it. Then do it again.

Contributions

Ido Vapner

CTO & AI Leader · Generative & Agentic AI · Kyndryl

Ido Vapner is a CTO and AI leader and AWS Ambassador with more than 18 years of experience in cloud, data, AI, and enterprise transformation. He works with CIOs, CTOs, CAIOs, and senior executives across CEE and emerging markets, helping organizations define and execute strategies for generative AI, agentic AI, cloud modernization, data platforms, security, and large-scale transformation.

As part of Kyndryl's strategic partnership with AWS, Ido helps drive cloud and AI initiatives, strengthen joint go-to-market activities, and accelerate customer adoption of AWS. He is also an AWS Ambassador specializing in generative and agentic AI, and a speaker and panelist on enterprise AI, cloud, data, and emerging technologies.

Chandrachood Raveendran

AI Leader · Generative & Agentic AI · Kyndryl

George Palangattil Koshy

Senior Cloud & AI Architect and AWS Ambassador · Kyndryl

Learn More

- → Kyndryl and Amazon Web Services (AWS): the alliance, competencies, offerings, and customer stories
- → Kyndryl and AWS expand strategic collaboration to advance agentic AI adoption (June 2026)
- → AI-ready enterprise modernization with AWS (August 2026)
- → Kyndryl AI Innovation Labs: Dallas, Liverpool, and Luxembourg
- → Kyndryl AI Innovation Lab Liverpool opens in the Royal Liver Building
- → Kyndryl Agentic AI Framework
- → Kyndryl on the AWS Partner Network

Ready to move your first AI project from demo to production?
Book a consultation with Kyndryl.

© Copyright Kyndryl, Inc. 2026

Kyndryl is a trademark or registered trademark of Kyndryl, Inc. in the United States and/or other countries. Other product and service names may be trademarks of Kyndryl, Inc. or other companies.

This document is current as of the initial date of publication and may be changed by Kyndryl at any time without notice. Not all offerings are available in every country in which Kyndryl operates. Kyndryl products and services are warranted according to the terms and conditions of the agreements under which they are provided. The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions. Kyndryl has no obligation to develop or release any of the functionality or products described in this statement. Any information about Kyndryl's possible future offerings is subject to change by Kyndryl at any time without notice and does not represent a commitment, promise or obligation for Kyndryl to deliver or make available any offering.