NYC TECH WEEK · BUILDING LLM-POWERED PRODUCTS

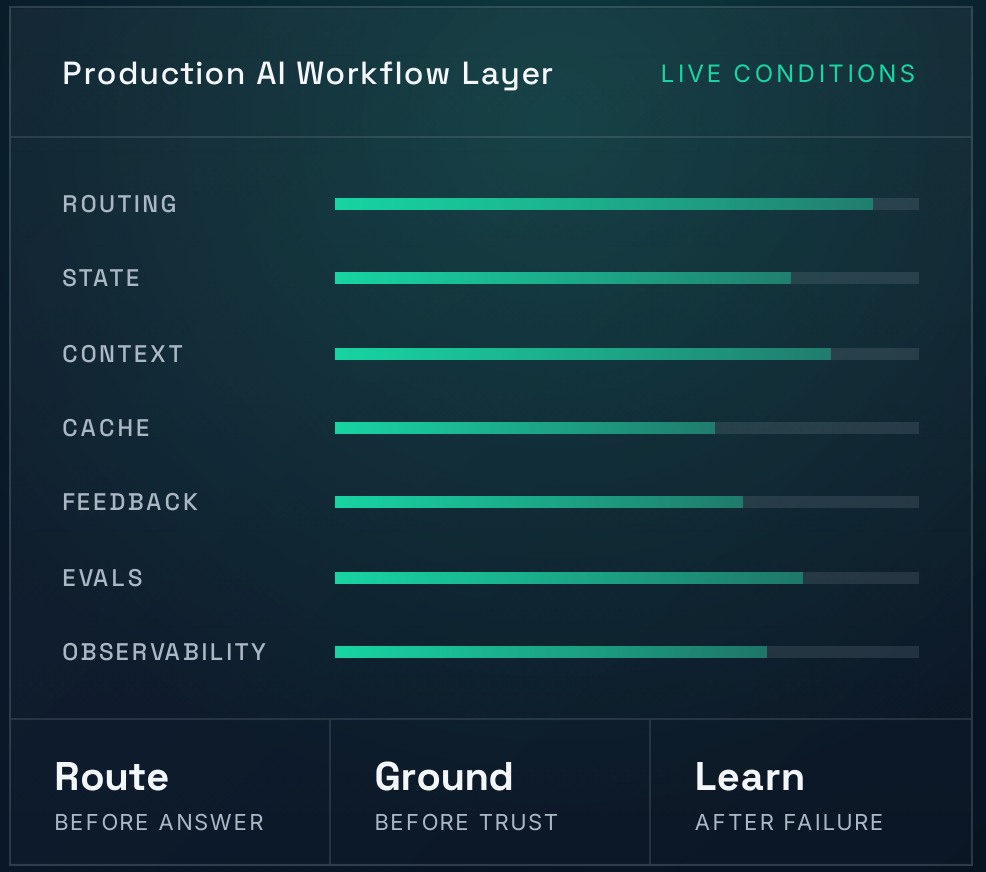
From Prototype to Production: What Actually Changes

Building production AI workflow systems — what the architecture actually required.

ROUTING | STATE | CONTEXT | CACHING | FEEDBACK | EVALS

Shilpi Nayak

Co-founder & CTO, GoodFin · 15+ yrs building large-scale AI at Google, Akamai, Intel



What We Built

A production AI workflow layer for a high-trust advisory product — not one assistant, but a system.

SUPPORTED WORKFLOWS

- Domain-specific Q&A
- Permissioned document questions
- Onboarding & profile collection
- High-trust transaction flows
- User-specific analysis
- Deep research & recommendations
- Scheduling, referrals & expert review

PRODUCTION STACK LAYERS

INTERFACES	Chat · Audio · Embedded UI
ORCHESTRATION	Router · State machine · Workflow nodes
MODELS	Provider abstraction · Model selection
KNOWLEDGE	Domain data · FAQ · Documents · Research
STATE	Profile · Permissions · Workflow · History
PERFORMANCE	Cache · Pre-compute · Refresh jobs
QUALITY	Tracing · Evals · Expert feedback · Gates

The domain was fintech. The architecture applies to any product where **AI supports expert-led decisions.**

One Interface, Many Workflows

The user sees one AI experience. The system routes across many different kinds of work.

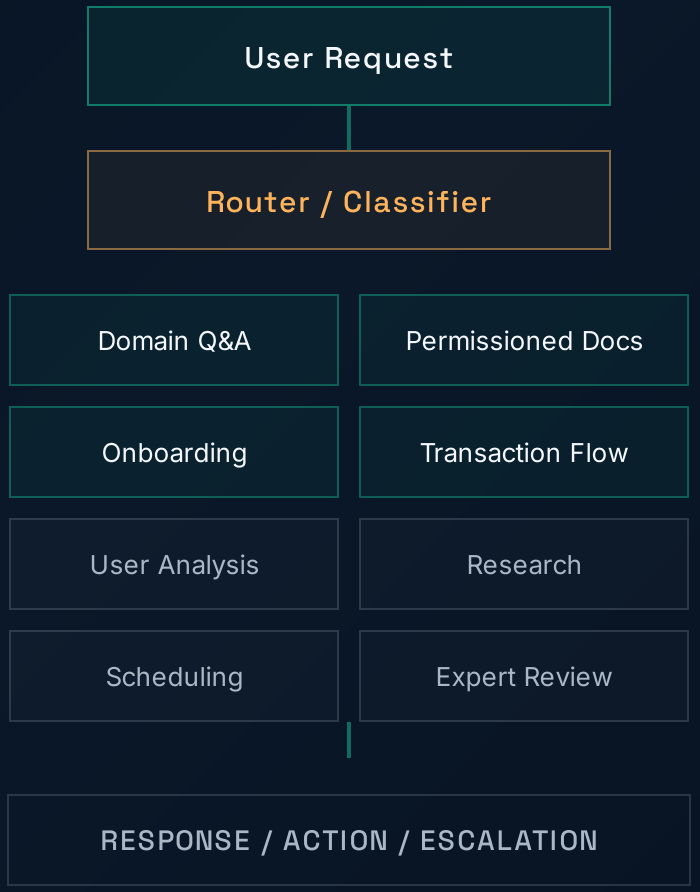
A document question is different from onboarding.

Onboarding is different from a transaction flow. A transaction flow is different from analysis.

A high-risk answer may need human review.

Routing determines quality, latency, cost, and risk — not just direction.

The first production decision is not **what to answer.**
It is **where to route.**



Routing Decided the Path, Model, and Cost

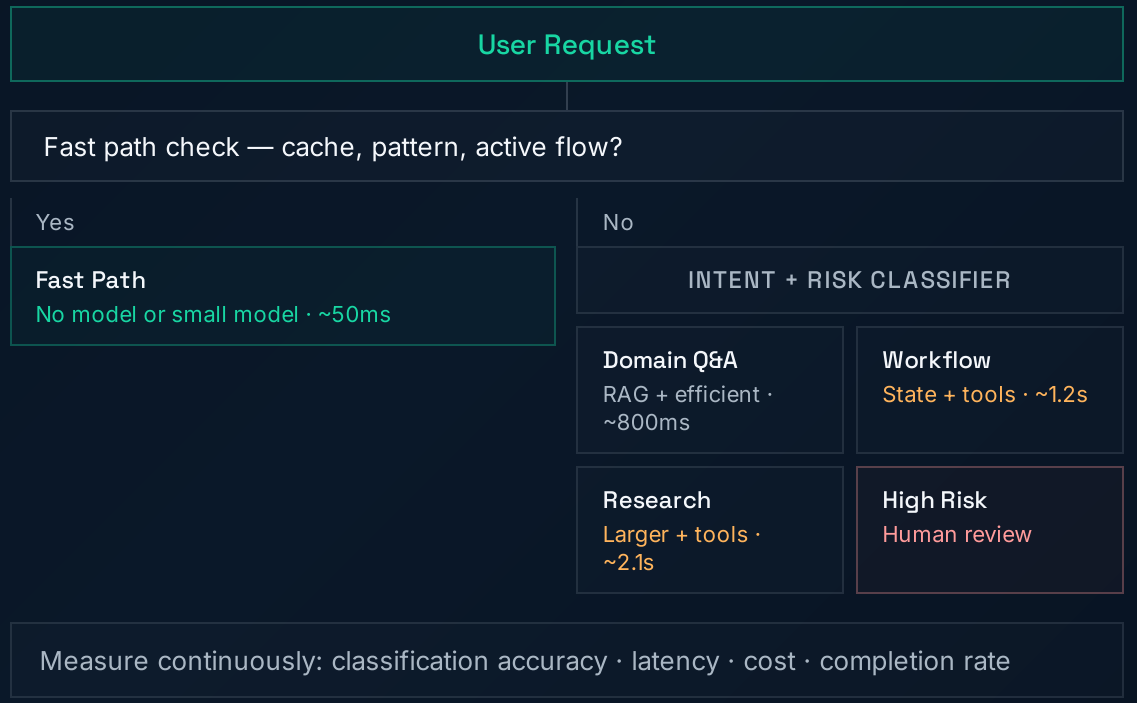
Model choice was not a one-time architecture decision. It became part of routing.

TASK → PATH → MODEL

TASK	PATH	MODEL
Greeting / cached response	Fast path	**No model**
Intent classification	Classifier	**Small, fast model**
Product / document Q&A	RAG	**Efficient model**
Transaction / workflow	State + tools	**Stronger reasoning**
Research / synthesis	Tools + analysis	**Larger model + tools**
High-risk / ambiguous	Escalate	**Human review**

Model choice became a routing decision.
The goal was not the biggest model everywhere — it was the right path for each task.

ROUTING DECISION FLOW



Caching Became Policy

Caching was not just a technical optimization. It became a product decision about when speed is safe.

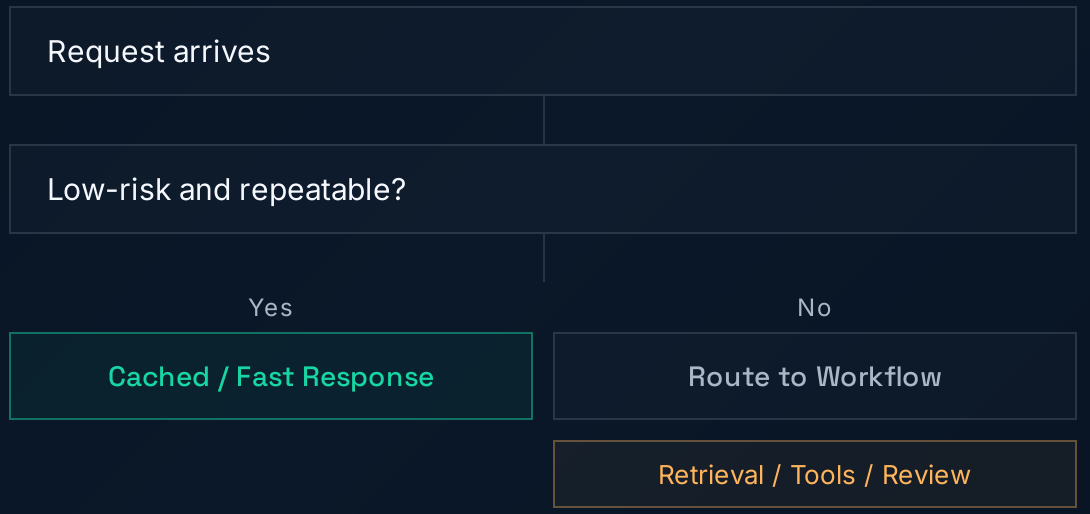
SAFE TO CACHE

- Simple repeated queries
- Low-risk patterns
- Pre-computed reports
- Scheduled refreshes

MUST BYPASS CACHE

- Confirmations and dollar amounts
- Active transaction flows
- Permissioned documents
- Anything stale that could mislead

CACHING DECISION



The question is not *how do we make everything fast?*
The question is **when is speed safe?**

State Made the Product Coherent

Chat is the interface. State is what makes it feel like a product.

WHAT THE SYSTEM HAD TO CARRY

User Profile
Preferences, history

Permissions
Access, roles, scope

Workflow State
Step, progress, flags

Prior Messages
Context, corrections

Long-term Memory
User preferences, prior context across sessions

Expert Feedback
Human corrections

WHAT STATE ENABLED

- Continue workflows across turns
- Answer side questions and return to task
- Recover from corrections without restarting

State tells the system where the user is right now.
Memory helps it remember preferences across sessions.
Both must be retrieved selectively — otherwise they become noisy

CONVERSATION STATE MODEL

User Context	PERSISTENT
Profile · Preferences · Permissions	
Workflow State	SESSION
Onboarding · Transaction · Scheduling · Step / Position	
Knowledge Context	RETRIEVED
Documents · Products · Research · Memory	
Feedback Context	LEARNED
Expert corrections · Eval cases	
Active Flow	CONTROL
Continue · Pause · Exit · Re-attach	

Context Was Engineered, Not Dumped

A lot of what people now call context engineering showed up here.

The easy question:

Can we retrieve?

The hard question:

Should this context be used here, for this user, in this workflow?

BEYOND VECTOR SEARCH

Vector search finds relevant text. A **knowledge graph** answers relationship questions — who competes, who invested, which entities are adjacent.

CONTEXT SOURCES

- Profile & permissions
- Domain data & FAQ
- Permissioned documents
- Prior conversation

CONTEXT LEVELS

LEVEL	APPROX. SIZE	USE CASE
Minimal	~800 tokens	Simple routed cases
Basic	~1,500 tokens	Light personalization
Standard	~2,500 tokens	Normal domain questions
Full	~4,000 tokens	Transaction flows / deep context

CONTEXT ASSEMBLY PIPELINE

1. **Retrieve** — fetch candidates from knowledge sources
2. **Permission** — filter by user access and scope
3. **Rank** — score by relevance to this request
4. **Add state** — inject workflow and memory context
5. **Respect budget** — trim to the right context level
6. **Model / tools** — only now does the model see context

✓ **Right context beats more context**

Real Users Break Workflows in Small Ways

Many production failures are not dramatic hallucinations. They are recovery failures.

WHAT USERS ACTUALLY DO

- — Say "good" instead of "looks good"
  The system expects a confirmation signal, not a word
- — Say "no" at the wrong moment
  Ambiguous negation mid-flow breaks the expected path
- — Change information mid-flow
  State must update without restarting the whole workflow
- — Ask a side question then return
  The system must hold position and re-attach cleanly
- — Abandon and return later
  Session state must survive gaps

The issue is not always model intelligence.
Often, the workflow is too brittle.

EXPECTED PATH

Confirm → Next Step → Complete

REAL PATH

”no” / ”change that” / side question

CAN THE SYSTEM RECOVER?

Yes → continue Trust maintained	No → stuck Trust breaks
Side question handled Re-attach to flow	State not updated Wrong assumptions

The user does not care that the system is sophisticated.
They care whether it understands them and helps them continue.

Feedback and Evals Closed the Loop

Human feedback is useful only if the loop closes. The correction must become product data — and feed the right layer: routing, retrieval, context, or response.

THE FEEDBACK LOOP

1. **User interaction**
   Normal production usage
2. **Failure or uncertainty**
   Wrong answer, low confidence, or user correction
3. **Expert correction**
   Factual · Workflow · Tone · Missing context — each type feeds a different layer
4. **Stored example**
   Correction becomes retrievable, searchable context
5. **Retrieval or workflow update**
   Future similar requests benefit immediately
6. **Eval case + regression check**
   One correction becomes a permanent regression test
7. **Improved system**

WHAT WE MEASURED

Routing accuracy **Correct path selected**	Grounding **Answer backed by source**
Answer quality **Correctness + relevance**	Latency **Per route type**
Workflow completion **Flow finished without break**	Regressions **Did a fix break something else?**
Repeat failures **Same mistake twice = system failure**	

Evals are not a quality check.
They are the release gate.

Tests: routing accuracy · grounding · workflow completion · latency · regressions · repeat failures

Observability: What You Need to See

You cannot improve what you cannot trace. Observability is not logging — it is understanding the full request path.

WHAT TO TRACE ON EVERY REQUEST

ROUTE TAKEN	Which path was selected and why
CONTEXT USED	What was retrieved, ranked, and injected
MODEL CALL	Prompt, model, tokens, latency
CACHE HIT/MISS	Was the fast path used, and should it have been?
WORKFLOW STATE	Position in flow, transitions, exits
HUMAN REVIEW	Was escalation triggered, and what was the outcome?
EVAL RESULT	Did this response pass the regression check?

A trace is not just a log.
It is the audit trail that makes the system debuggable and

EXAMPLE REQUEST TRACE

0ms	Request received User: "What is the minimum investment for Fund A?"	IN
12ms	Router: classified as domain Q&A Confidence 0.94 — direct RAG path selected	OK
18ms	Cache miss — proceeding to retrieval Query not in low-risk cache	MISS
340ms	Context assembled: 2,100 tokens 3 document chunks + user profile + permissions	OK
780ms	Model response generated Grounded — source: Fund A term sheet, page 4	OK
790ms	Eval check passed — no regression Matched 2 similar eval cases	PASS
795ms	Response delivered to user	OUT

Alert trigger: If latency > 1.5s on domain Q&A, flag for routing review.

The Production System Around the Model

These are the design decisions we had to make. Not the perfect architecture. The real one.

DESIGN DECISIONS

1. **Route by workflow type**
   Before answering, know what kind of work this is
2. **Match path to task**
   Fast when safe, careful when needed, escalated when appropriate
3. **Manage state explicitly**
   Workflows break without it; stale state misleads with too much of it
4. **Assemble context, do not dump it**
   Right context beats most context
5. **Make caching a policy decision**
   Speed is only safe when the content is safe to cache
6. **Build for recovery**
   Real users break workflows in small, predictable ways
7. **Close the feedback loop**
   Human corrections must become product data, not Slack messages
8. **Evals are the release gate**
   Measure routing, grounding, latency, completion, regressions

PRODUCTION AI WORKFLOW STACK

INTERFACE	Chat · Audio · Embedded UI
ROUTING	Classifier · Workflow selector · Escalation
STATE	Profile · Permissions · Workflow · History
CONTEXT	Retrieve · Filter · Rank · Budget
MODEL	Provider abstraction · Model selection
CACHE	Policy · Pre-compute · Refresh jobs
HUMAN	Review · Correction · Escalation
FEEDBACK	Expert corrections → context + evals
EVALS	Routing · Grounding · Quality · Regression
OBSERVABILITY	Traces · Latency · Alerts · Audit

Applies to any advisory workflow: **fintech, insurance, legal, healthcare navigation, enterprise operations.**

What Actually Changes

The shift from prototype to production is accountability.

THE QUESTION CHANGES

PROTOTYPE ASKS	PRODUCTION ASKS
Can the model answer this?	Did it route correctly?
Does it sound right?	Was the context right?
Does the demo impress?	What happens when it's wrong?
	Did the system improve?

The model answers.

The system improves.

Access to strong models is no longer the only differentiator.
The teams that win are those whose systems learn, measure, recover, and earn trust over time.

PRODUCTION AI WORKFLOW SYSTEM

- Routing
- \+ State management
- \+ Context assembly
- \+ Cache policy
- \+ Guardrails & permissions
- \+ Human feedback loop
- \+ Evals & regression checks
- \+ Observability & traces
- = **Reliable product behavior**