Building GenAI Platform at DoorDash

Principles, bets, pivots led us to success

Platform bets under a moving target

• Siddharth Kodwani & Swaroop Chitlur • QCon AI Boston • June 2, 2026

APRIL 2023

“

Are we really going to spend this much on OpenAI?

Signing the OpenAI contract was the first moment the platform question became real: if usage takes off, how do we make it productive, accountable, and safe?

THE THROUGH-LINE

The primitive kept changing

We thought we were building a platform for model calls. The platform's job changed as the product surface changed.

2023

Model calls and provider access

2024–25

Workflows, evals, traces, cost, optimization

2025–26

Agents, tools, identity, delegated work

OPERATING MODEL

Our decision compass

Not just Foundations Tenets on a wiki — these became our checklist for platform decisions.

CUSTOMER

Be customer obsessed

Start with the teams and use cases

PRODUCT

Build products, not systems

Onboarding, support, docs, roadmap

PATH

Make the right thing easy

Guardrails and paved paths

PROOF

Measure how we are doing

Adoption, cost, reliability, value

CHAPTER 1

The customer changed

- Demand arrived before the platform
- The customer expanded from MLEs to product engineers and non-engineers
- Decision: product impact first, APIs-first

CHAPTER 1 · APIS-FIRST

SWEs shouldn’t need ML platform expertise

“Use a notebook.” “What is a notebook?” — the moment the customer shift became concrete.

- Audience shifted: MLEs → all engineers
- Interface: APIs, SDKs, docs, examples, paved paths
- When the user archetype changes, the interface has to change too

CHAPTER 1 · PRODUCT IMPACT FIRST

Business impact, not generic chatbots

- Named product teams first; shared tools second
- Platform shape came from production use cases
- Customer Obsessed: start with the problem, not the solution

CHAPTER 1 · PRODUCTION USE CASES

Use cases clustered into three buckets

Automation

- Support & trust/safety
- Fraud review
- Receipt processing
- Quality review

Recommendations

- Catalog building
- Query intent
- Search & ads relevance
- Ranking

Personalization

- Richer shopping & logistics
- Context from Cx, Mx, Dx

CHAPTER 1 · PLATFORM PURPOSE

Help teams optimize between three forces

Accuracy

The right model is not always the biggest model

Latency

Product tolerance and latency budget

Cost

The cost curve as usage grows

CHAPTER 1 · ADOPTION

Demand escaped beyond ML folks

5,000+
internal users onboarded (excludes coding agents & chat tools)

~45
users onboarding per day

40%
non-engineers

CHAPTER 2

Adoption created accountability

- Adoption made cost, routing, and rate limits unavoidable
- Decision: LLM Gateway as the shared policy surface
- Technical proof: the Gateway

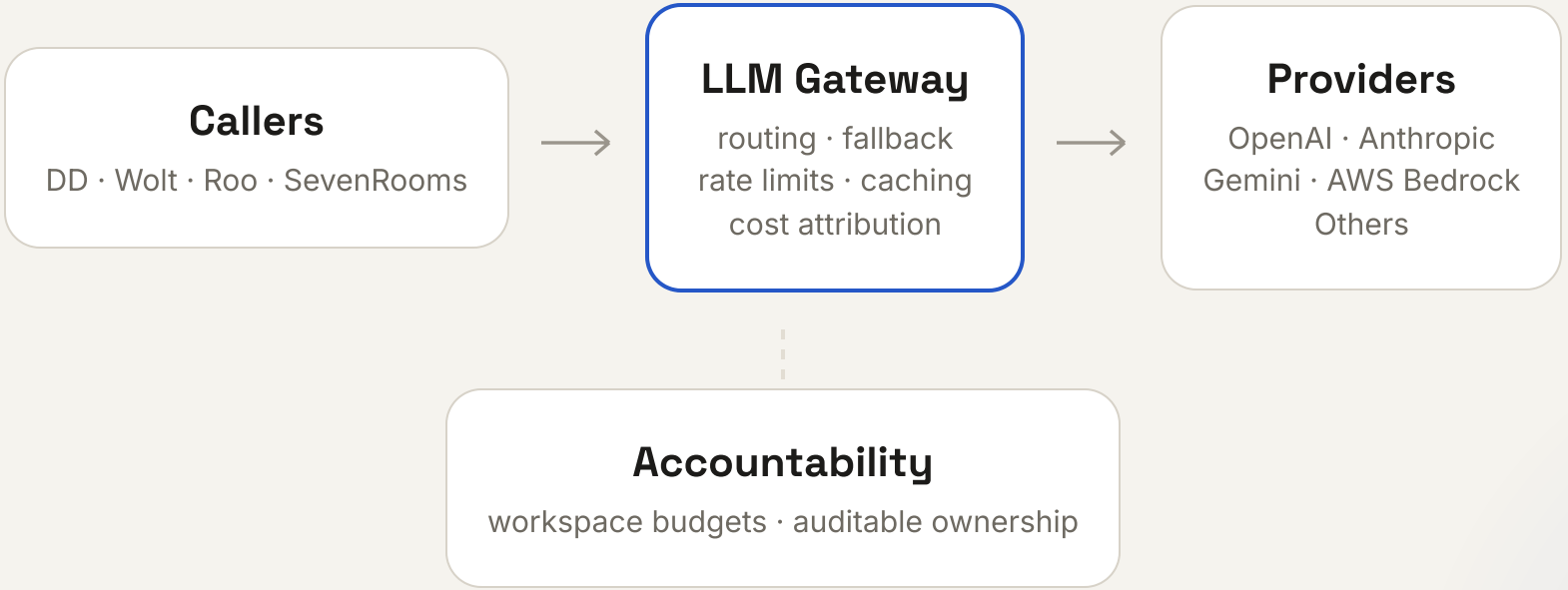
CHAPTER 2 · LLM GATEWAY

One shared policy surface

- One API, one SDK
- Routing and fallback
- Rate limits and prompt caching
- Cost attribution and budgets
- Workspace + API key as the ownership unit

CHAPTER 2 · ARCHITECTURE

The LLM Gateway sits between callers and providers



CHAPTER 2 · PLUMBING IS THE PRODUCT

A good platform surface makes change cheaper

- Providers expose different quota and routing models
- Product teams should not absorb that complexity
- Result: trade-offs became cheap to revisit

600M+
API calls / week (peak)

72K+
API calls / min (peak)

1,200+
API calls / sec (peak)

CHAPTER 3

Vendor-first taught us what to own

- Vendor-first was the right starting move
- It failed differently across surfaces
- Lesson: own the company-specific surfaces

CHAPTER 3 · EVALS

Evals failed on workflow fit

- Vendor onboarded in Q4 2025
- Teams needed custom tracing and annotation workflows
- Teams routed around the product — a signal they needed evals

CHAPTER 3 · OWN THE SURFACE

Buy to learn, then own what must fit

- Own the workflow surfaces that must fit your company
- If the vendor works, scale it
- If not, take the lesson and build

CHAPTER 3 · MODELS AT SCALE

Vendor-first worked, until scale changed the trade-off

- Deprecation treadmill
- Quota constraints outside our control
- Cost curves and control
- Portability went from a nice principle to a scaling requirement

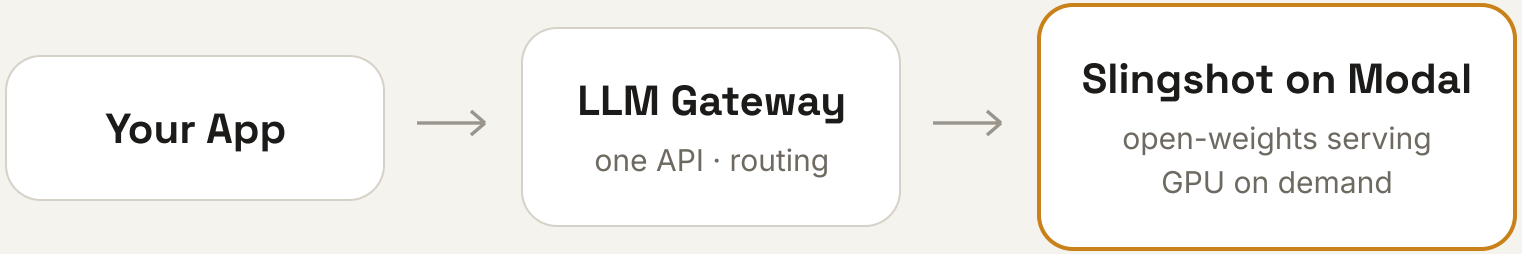
CHAPTER 4

Portability became real

- The long-game bet became urgent
- Open-weights made portability executable
- Gateway let teams switch without rewriting apps

CHAPTER 4 · PROJECT SLINGSHOT

Open-weights on Modal



Online · Batch · Fine-tune

vLLM / SGLang engines

GPU elasticity, no lock-in

Llama · Qwen · Mistral · GLM

CHAPTER 4 · RESULTS

Cheaper, faster — and portability held

- Better developer velocity for batch workloads
- Cheaper and faster in early use cases
- Vendor-first softened; portability held

Millions $

projected annualized savings — single-digit-million range, early adoption

CHAPTER 5

5

Agents changed the primitive again

- Agents brought tool use, streaming, delegated auth, and multi-step workflows
- We expanded scope from MCP Gateway to Agent Gateway
- The platform had to own identity, not just model routing

CHAPTER 5 · CONTEXT

Demand arrived before the platform did

Teams were already building agents while the paved path was still forming.

Early 2024

Teams building independently

Mid 2024

Tool-calling named as a future use case

Late 2024

MCP emerges; no paved path yet

2025

25+ agent projects

CHAPTER 5 · SCOPE EXPANSION

MCP Gateway → Agent Gateway

STARTED WITH

MCP Gateway

- API-key auth
- Coding agents only
- MCP traffic only

→

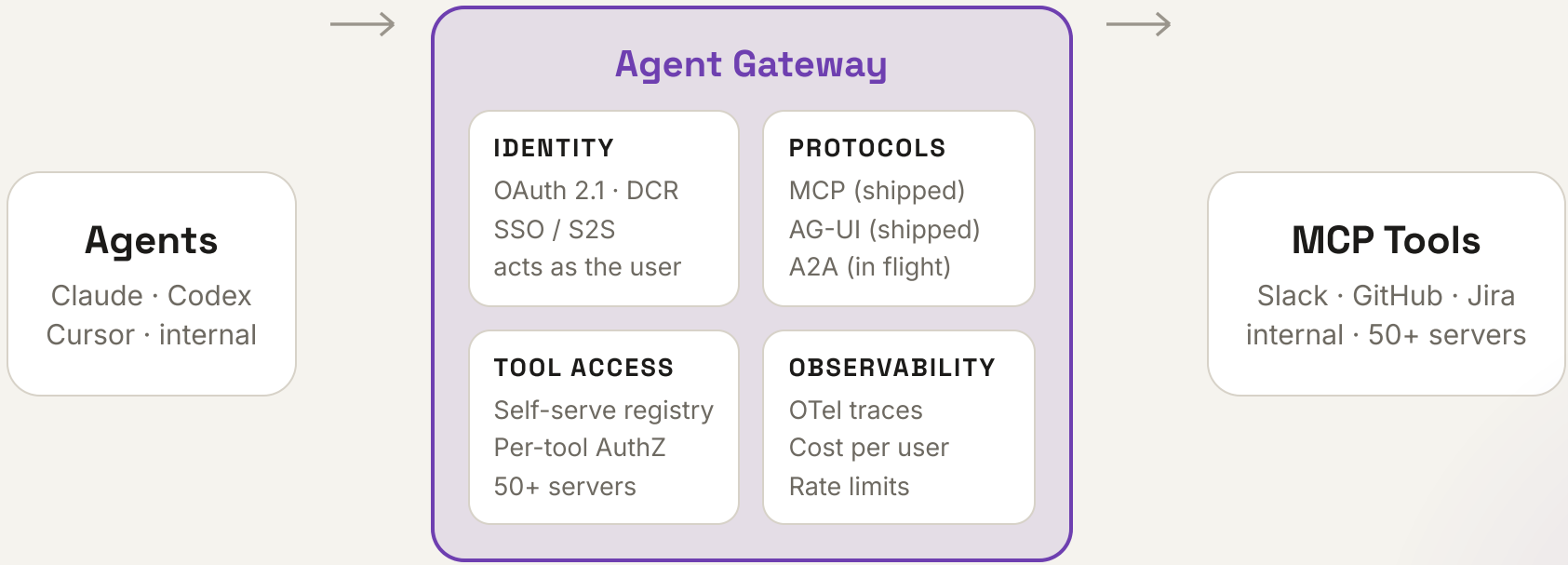
GREW INTO

Agent Gateway

- OAuth 2.1 + DoorDash SSO + S2S
- Self-serve tool registry
- AG-UI streaming · A2A (in flight)
- Per-user RBAC + cost attribution

CHAPTER 5 · AGENT GATEWAY

Agents brought new primitives across the stack



CHAPTER 5 · AI-NATIVE BUILD

Why it connected to how we already built

- Python-first advocacy meant agent frameworks landed on a paved path
- Build Products, Not Systems — auth baked in, so teams ship agents, not plumbing
- Identity primitive — agents act as the right user, not a shared service account

CHAPTER 5 · STAYING NIMBLE

Protocol-agnostic core; new protocols slot in

Designed for portability from day one — the core stays stable as protocols change.

PROTOCOL	STATUS	YEAR	WHAT IT UNLOCKED
MCP	Shipped	2025	Tool access for coding agents
AG-UI	Shipped	2026	Streaming agent UX — idea → prod in 3 weeks
A2A	In flight	2026	Agent-to-agent orchestration
Future	Watching	…	Whatever the next protocol pressure is

CHAPTER 5 · RESULTS

Agent traffic found the paved path

3 weeks

AG-UI streaming: idea → production

300K+

daily tool calls — early adoption

Python

every agent framework was Python-native — the bet paid off

REPORT CARD · THE NUMBERS

Strategy without proof is just a story

5,000+

users · 40% non-engineers · DD · Wolt · Deliveroo · SevenRooms

Millions $

projected annualized savings · embeddings 20× cheaper · visual models 72% cheaper

300K+

daily tool calls · 50+ MCP servers

REPORT CARD · SCORECARD

Explicit strategy let us pivot without starting over

Held

- Product impact first
- APIs-first
- LLM Gateway
- Portability

Softened

- Vendor-first — the right starting move, not a religion

Emerged

- Eval workflow ownership
- Agent Gateway
- Identity as a primitive

WHAT'S NEXT

Directional bets, not a fixed roadmap

- Top-down: agents running parts of the business, agentic commerce
- Bottom-up: build where product teams are blocked
- Directional bets plus signals to watch

THREE TAKEAWAYS

What to remember after the architecture fades

1 · Write strategy early

What to build, what to ignore, and how to judge pivots.

2 · Build around customers

Use cases should pull the platform forward.

3 · Own company-fit surfaces

Auth, cost, evals, observability, governance, DX.

Q&A

Happy to go deeper on LLM Gateway · open-weights · evals · Agent Gateway · strategy & operating-model lessons.

- Siddharth Kodwani & Swaroop Chitlur • QCon AI Boston • June 2, 2026

Thank you

Building GenAI Platform at DoorDash — the bets, what held, and what we learned.

- Siddharth Kodwani & Swaroop Chitlur