FOR DATA, SECURITY & PRIVACY TEAMS

Your Data. *Our Discipline.*

The Highest-Stakes *Question*

Before any client gives an AI platform access to contracts, HR records, customer interactions, or operational data, the same questions surface — and they should. *Where does our data live? Who can see it? Will it be used to train somebody else's model? Can another client of yours ever touch it?*

This page is our answer. ClearData AI is built on a data-lake-as-control-plane architecture — Azure Data Lake Storage Gen2 and Cosmos DB sit at the centre of everything the platform does, and every byte that flows through them is scoped to a single account. Your data is your data. We don't sell it, share it, or train foundation models on it. We engineer for that promise — and we instrument it so you can verify it.

THE TRUST EQUATION

Realized Capability = What We Build × What You Trust Us With

Trust is earned with architecture, not asserted with marketing — every guarantee on this page maps to a real control in the platform.

Isolation by design · Least privilege by default · Observable by audit

OUR COMMITMENTS, AT A GLANCE

100%	0	Least	Full
ACCOUNT-SCOPED DATA ISOLATION	CROSS-CLIENT DATA SHARING — EVER	PRIVILEGE ACCESS, BY DEFAULT	AUDIT TRAIL IN YOUR DATA LAKE

The Architecture of *Trust*

Everything the platform does flows through two stores that you can see, govern, and — on a PaaS deployment — own outright: an Azure Data Lake (ADLS Gen2) and a transactional database (Cosmos DB). Knowing what each one holds is the foundation of every other promise on this page.

The Data Lake (ADLS Gen2)

The control plane and durable store

- **Agent configuration.** Fleet definitions, sub-agent prompts, skills, and tool servers live as files in the lake — versioned, auditable, and account-scoped.
- **Your business artifacts.** Documents, datasets, generated outputs, and source files the agent reads or writes — under paths you control.
- **Knowledge-base vector stores.** RAG indexes are created per account; one client's embeddings are never visible to another.
- **Operational telemetry.** Run logs, token counts, and tool-call audit trails are written to your lake — the same data you use for governance reporting.

The Databases (Cosmos DB)

Conversation state and memory

- ✓ **Checkpoints.** LangGraph thread state — partitioned by thread_id, so one conversation cannot read another's working memory.
- ✓ **Long-term memory.** Per-user preferences and facts the agent has learned — partitioned by user_id, scoped to an account.
- ✓ **Short-term telemetry.** Per-thread tokens, latency, and tool calls — written with a 30-day TTL and partitioned by user_id.
- ✓ **Operational data.** Application-specific records (e.g. HR scoring, contract extracts) — each in its own container, each governed by your RBAC.

How the Lake Connects to the *Application*

The application doesn't bake your data into its codebase. Instead, runtime middleware reads only the files your account is permitted to see from the data lake — agent definitions, prompts, skills, knowledge-base indexes — and loads them dynamically. When the agent finishes, outputs are written back to *your* lake under *your* paths. The lake is the source of truth; the application is just the engine that reads from it. Change a file in your lake, and the next conversation reflects the change — without redeployment, without us touching your environment.

How Your Data Stays Yours

The single most common question we get on a shared infrastructure is the right one: ***if other clients are on the same platform, what stops them from reaching my data?*** The short answer is that isolation is enforced at every layer the data touches — not by policy, but by the way the platform is wired. Each row below maps a layer of the stack to the control that keeps your account's data yours, and yours only.

LAYER	ISOLATION MECHANISM	WHAT THIS MEANS FOR YOU
Identity	Microsoft Entra ID (single- or multi-tenant), signed JWT validated on every request, plus an email & domain allow-list stored in ADLS.	*Only authorized users in your organization reach the platform — no anonymous access, no shared passwords.*
Data Lake paths	All client artifacts, vector stores, and telemetry are written under account-scoped paths; the application resolves paths from the account context, not user input.	*There is no path a user from another account can construct to reach your files.*
Databases	Cosmos DB containers are partitioned by user_id / thread_id, with queries filtered by account_id at the application boundary.	*Cross-account queries are blocked at the data layer, not just the UI.*
Knowledge base / RAG	Each account creates its own vector stores; the Knowledge-Base MCP server resolves stores from the caller's account_id.	*Embeddings and source citations are account-bounded — RAG cannot retrieve another client's content.*
Secrets & keys	Credentials live in Azure Key Vault, accessed by managed identity; per-account API-key storage is segregated under account-scoped settings files.	*Your API keys, model keys, and connector credentials never enter another account's runtime.*
Code execution	Sandboxed Azure Container Apps dynamic sessions, started per-call; safe-AST evaluator on configuration files (no arbitrary code execution).	*Agent-generated code runs in an isolated container — it cannot reach your other systems, or anyone else's.*
Audit trail	Every run writes structured telemetry (tokens, tool calls, prompts, outcomes) to your data lake — not a shared ClearData AI log.	*Every action is observable in infrastructure you own and can hand to an auditor.*

On a PaaS deployment, every one of these stores lives inside your own Azure tenant — your subscription, your region, your encryption keys (customer-managed via Key Vault if you choose). On managed SaaS, the same isolation logic applies, just on infrastructure we operate to the same controls.

Least Privilege & *PII Protection*

Isolation answers *who can reach your data*. Governance answers *what they can do with it once they're inside*. Our position on both is the same: the smallest possible privilege for the shortest necessary time, with personal and sensitive data treated as something the agent should rarely need to see in the first place.

Least-Privilege Governance

- **Role-based access at every tier.** Fleet, agent, MCP server, orchestrator, and skill visibility are all gated by admin + per-user-group flags, AND-ed together — a user only sees what every layer agrees they should see.
- **Password gates on sensitive views.** Memory, knowledge-base, vector stores, billing, and admin tabs each support an optional password gate on top of role permissions — a second factor for the riskiest screens.
- **Read-only by default.** Sub-agents can be flagged read-only; SQL connectors are wired for SELECT-only access; write operations are explicit and auditable.
- **Human-in-the-loop on consequential actions.** Sending emails, calling external systems, writing to production data — every consequential tool can require an explicit human approval before it runs, with session-scoped overrides for trusted patterns.
- **Bounded autonomy.** Hard wall-clock and token budgets cap every autonomous run; the platform halts cleanly when budgets are exceeded — no runaway agents touching data outside scope.

PII & Anonymization

- ✓ **Minimize what the agent sees.** Connectors are scoped to the smallest field set needed; identifiers and free-text PII can be redacted before the data ever reaches the model.
- ✓ **Anonymization middleware (capability).** Middleware on the agent side can mask, tokenize, or pseudonymize PII fields on the way into the prompt and re-attach the originals on the way out — names, emails, IDs, account numbers, addresses are common targets.
- ✓ **Custom redaction policies.** Anonymization rules are defined per fleet / per connector during scoping — what to mask, what to keep, what to drop entirely — and they live in the same versioned configuration as the rest of the agent.
- ✓ **Encryption everywhere.** AES-256 at rest across ADLS and Cosmos DB; TLS 1.2+ in transit; customer-managed keys in Azure Key Vault on PaaS — encryption is the floor, not the ceiling.
- ✓ **No third-party training, ever.** Frontier model providers (Anthropic, OpenAI, Google, Azure OpenAI) run under enterprise contracts that prohibit training on your inputs; we don't train foundation models on your content either.

Anonymization is implemented as middleware on the agent side — meaning the redaction step is part of the agent stack itself, not a separate product. New rules can be added without retraining anything; they take effect on the next conversation, the same way the rest of the configuration does.

Data Mining for Improvement – *Done Right*

To make the product better — fix bugs faster, ship the right features, roadmap with evidence rather than guesswork — we do mine the telemetry the platform produces. But it is worth being precise about *what* we mine, *how* it gets there, and what we will never touch.

What We Mine — Operational Telemetry

- ✓ **Tokens, latency, tool-call counts.** Per-run metrics that tell us where the platform is slow, expensive, or inefficient — and where engineering effort returns the most value.
- ✓ **Tool & skill usage patterns.** Which capabilities get used, which don't, which fail — the signal that drives the roadmap and prioritizes integrations.
- ✓ **Error and retry signatures.** Anonymized stack-level patterns that surface bugs and regressions before they propagate.
- ✓ **Throughput & adoption shape.** Aggregate counts of runs, threads, and active fleets — at the account level, never at the row level.

What We Don't Touch

- ✗ **Your business content.** Documents, contracts, HR records, customer interactions — the agent reads them in your tenant; we don't pull them out for our purposes.
- ✗ **Prompts & conversation transcripts.** User prompts and model outputs stay in your environment. We don't aggregate or train on them.
- ✗ **Personally-identifiable fields.** Names, emails, IDs, phone numbers, account identifiers — anonymized at source where they're touched at all, and never aggregated centrally.
- ✗ **Cross-client joins.** Telemetry from one account is never joined with telemetry from another for product analysis — accounts are analyzed independently or as fully anonymized aggregates.

FOUR HARD RULES ON PRODUCT-IMPROVEMENT MINING

Aggregated	*Anonymized*	*Opt-out*	*Not for training*
Account-level or higher — never per-row	PII stripped before it leaves the account	Off-switch for product analytics on request	Never used to train foundation models

What You Can Verify

None of this is a promise on a slide. Every commitment on this page maps to something you can inspect — in your tenant, in your audit logs, in the configuration you control.

How You Verify It

- ✓ **Open your data lake.** Every artifact, configuration, and audit record the platform produces is a file in your ADLS — readable with your standard tooling.
- ✓ **Read the telemetry.** Per-run tokens, tool calls, latency, and outcomes are in your Cosmos containers — query them, dashboard them, alert on them.
- ✓ **Inspect the configuration.** Fleet definitions, allow-lists, and middleware settings are versioned JSON / Markdown files — diff them like any other code.
- ✓ **Run on your tenant.** On PaaS, the entire stack lives in your subscription — your VNet, your Key Vault, your CMK. There is nothing left to take our word for.

Our Standing Commitments

- ✓ **We will not sell your data.** Not aggregated, not anonymized, not at any price. It is not a product.
- ✓ **We will not share it with other clients.** Isolation is architectural — there is no path for another account to read it.
- ✓ **We will not train foundation models on it.** Not our own models, not the providers' models — model usage runs under enterprise contracts that forbid it.
- ✓ **We will not surprise you.** Material changes to data handling are communicated before they take effect, with a clear path to opt out.
- ✓ **We will help you audit it.** On request, we walk through architecture, controls, and telemetry with your security or DPO team in detail.

YOUR DATA. *YOUR PERIMETER.* YOUR CONTROL.

ClearData AI is built on a data lake that you can see, a database that you can govern, and a discipline of least privilege that doesn't bend — so the answer to every hard question stays the same on day one and day one thousand.

Want the architecture walk-through? We'll bring the diagrams and *your* security team's questions.

This document describes the platform's data-handling architecture at a capability level. Implementation specifics (file paths, schemas, internal middleware names) are shared under NDA during security review.

ClearData AI The AI Operating Layer for Enterprise **info@cleardataai.ai**