



SECURITY & DATA HANDLING

Security Architecture & Data Handling

Document type Technical security overview

Audience CISO / CTO / Security & IT due diligence

Version 1.0 · **Classification** Customer-shareable overview

What this covers, and who it's for

Retail operations data is sensitive. Store imagery, point-of-sale figures, cash positions, manager commentary, and a brand's operating model are commercially valuable and, in many cases, regulated. Before adopting ReStore, security and engineering leaders rightly want to know exactly where their data lives, who can reach it, what leaves the platform, and how it is handled when artificial intelligence is involved.

This document answers those questions directly. It describes ReStore's layered security model, how customer data is isolated between tenants, how information moves through the platform, and — importantly — precisely what is and is not sent to external AI providers. It is deliberately transparent about design tradeoffs, because informed evaluation is faster than reassurance.

Contents

01 Executive summary	03
02 Frequently asked questions	04
03 System landscape & trust boundaries	05
04 Identity, authentication & sessions	07
05 Authorization & tenant isolation	08
06 Data ownership, isolation & encryption	09
07 Application & transport security	10
08 How data flows through the platform	11
09 AI data handling & external egress	12
10 Notification & messaging security	14
11 Infrastructure, deployment & secrets	14
12 Monitoring, resilience & operations	15
13 Transparency: design tradeoffs we disclose	15
14 Compliance posture & roadmap	16
15 Evidence & next package for buyers	16

SECTION 01

Executive summary

ReStore is built on defense in depth: controls are enforced at every layer, so a failure in one does not, by itself, expose customer data.

The platform applies independent controls across identity, application, data, infrastructure, and asynchronous processing. The principle throughout is that a bypass at one layer remains constrained by the layers beneath it — most importantly by database-enforced isolation, which is the authoritative boundary between tenants.

Highlights of platform security and data protection

- **Database-enforced tenant isolation.** Customer separation is enforced inside PostgreSQL using forced Row-Level Security (RLS), not only in application code — so isolation holds even if an application-layer check is missed.
- **Per-request identity verification.** Every request's token is verified against the identity provider's published keys, and the database session is then explicitly scoped to that caller before any query runs.
- **Strict transport, browser, and CORS protections** across the API and web surfaces, with production startup validation that refuses unsafe configuration.
- **Centralized secret management** for runtime credentials in Google Cloud Secret Manager, with rotation and source-of-truth discipline.
- **Durable asynchronous processing** with retry, backoff, and dead-letter handling, isolating long-running and AI work from the user request path.
- **Explicit provider adapters** for AI, email, and push — giving ReStore deliberate, auditable control over every byte that leaves the platform.

WHAT THIS MEANS FOR YOUR DATA

Your operational data is stored within ReStore's cloud environment, isolated to your brand at the database level, encrypted in transit and at rest, and only ever sent outside the platform through named, server-mediated integrations that this document describes in full.

Frequently asked questions

Six questions come up in almost every evaluation. Here are the short answers; the rest of the document is the evidence behind them.

■ Where does our data live?

In ReStore's cloud environment on Google Cloud — a managed PostgreSQL database (Cloud SQL) for structured data and Google Cloud Storage for uploads and media. Infrastructure is defined and governed as code, so the running environment matches an auditable, version-controlled definition.

■ Who can see it?

Access is scoped to your brand and enforced in three layers — application authentication and role checks, business-rule enforcement in services, and, authoritatively, forced Row-Level Security inside the database. One customer cannot see another's data, and roles constrain what each user sees within your organization.

■ What leaves the platform?

Only specific, server-mediated calls to named AI, email, and push providers. For AI, that is task-relevant prompt text, the image being analyzed, and the business context that task requires — never an open pass-through of your database. Section 9 details exactly what is sent and to whom.

■ Is our brand IP protected?

Yes. Your operating model, brand rules, and operational taxonomies are treated as your data: isolated to your tenant, never shared with or exposed to other customers, and never repackaged into a shared product. They are the configuration that makes ReStore yours.

■ How is it protected in transit & at rest?

Transport is encrypted end to end with strict transport security enforced in production. Data at rest uses the managed encryption of the underlying Google Cloud services, and runtime secrets are held in a dedicated secret manager rather than in code or configuration files.

■ What is your compliance posture?

ReStore and its development team are SOC 2 certified for the current platform. The same controls and standards are being applied throughout the development of AI features, and ReStore will re-certify as soon as the AI capabilities are complete. Data Processing Addenda and a subprocessor list for AI, email, and push providers are available under NDA.

THE "SECRET SAUCE" QUESTION, ANSWERED PLAINLY

The most common concern from technical buyers is: *"How are you not giving away my secret sauce?"* ReStore's value is applying **your** brand and operating model to incoming operational data. That configuration is your intellectual property. It stays inside your tenant, is not commingled with other customers, and is governed by your agreement. Foundation-model providers are used as processors under contractual terms — not as a place your strategy is deposited.

SECTION 03

System landscape & trust boundaries

A small number of well-defined components, with every outbound integration encapsulated behind an adapter boundary in the API.

Core platform components

Identity provider	OIDC issuer and token services (restore-iam)
API tier	GraphQL API, domain services, and queue orchestration (restore-api)
Web & admin clients	Customer web app and administrative console
Mobile client	Native store-team application
Data layer	PostgreSQL on Cloud SQL with forced Row-Level Security
Async layer	Redis with BullMQ workers for background processing
Storage layer	Google Cloud Storage for uploads, media, and pipeline artifacts
Infrastructure	Terraform-managed Google Cloud infrastructure (defined as code)

External service boundaries (subprocessorsors)

Three categories of external service are used, each behind an explicit adapter so that data egress is controlled and observable:



Figure 1 — Trust boundaries. Clients reach the platform only through the API; all external egress passes through adapters.

The layered security model

Four layers operate independently, so that a bypass at one is still constrained by those below it:

1. **Network & infrastructure** — private networking and infrastructure-as-code governance.
2. **Identity & authentication** — OIDC tokens verified on every request.
3. **Application controls** — role checks, rate limiting, query governance, hardened headers.
4. **Database authorization & isolation** — forced RLS as the authoritative tenant boundary.

Identity, authentication & sessions

An internal OIDC provider issues and verifies tokens consistently across web, admin, and mobile surfaces.

Token model

- The OIDC provider and its clients are configured centrally in the identity service.
- The public mobile client uses **PKCE (S256)** and carries no client secret, the recommended pattern for public clients.
- The API verifies every JSON Web Token against the provider's remote published keys (JWKS), with issuer and audience checks.
- WebSocket connections are authenticated separately; unauthenticated connections are rejected.

Signing keys

- Production uses stable signing keys provided through the environment; the public key set is exposed so verifiers can establish trust.
- Ephemeral keys exist only as fallback behavior when a persistent key is absent — not a production path.

Roles & hierarchy

An explicit role hierarchy runs from system administrator through to store associate, alongside internal operator roles, with helper logic for level and reporting-chain checks. Roles are one input to authorization; the database boundary described next is the authoritative one.

Authorization & tenant isolation

Tenant separation is brand-scoped and enforced in three layers, with the database as the final, authoritative arbiter.

Three layers of enforcement

1. **Resolver layer** — authentication and role checks at the API entry point.
2. **Service layer** — business-rule enforcement within domain services.
3. **Database layer** — forced Row-Level Security, the authoritative boundary that holds even if a higher layer is bypassed.

RLS as the authoritative control

- RLS is **forced** on protected tables — it cannot be silently skipped by ordinary queries.
- Policies constrain both reads and writes, keyed off session variables such as brand, user, and role.
- Helper functions determine the current brand, user, and any administrative state.
- Sensitive tables such as notifications and push tokens carry additional user-bound and brand-bound checks.

Runtime context binding

For each authenticated request, the API opens a database transaction and sets the session's brand, user, role, and (where applicable) assigned-brand context *before* any repository or service code runs. Query visibility is therefore bound to the caller from the first statement — not assembled defensively in application code afterwards.

BACKGROUND WORKERS — AN EXPLICIT, CONTROLLED EXCEPTION

Background jobs run outside a user's request context and therefore use a dedicated database client that operates with elevated privilege per transaction. This is a deliberate, documented path — disclosed in full in Section 13 — and it is a high-scrutiny area in code review precisely because it sits outside the normal tenant-scoped flow.

Data ownership, isolation & encryption

Your data — including the brand model that configures the platform — is yours, isolated to your tenant, and encrypted throughout.

Ownership and the brand model

ReStore distinguishes three intake channels for store data: **visual** information derived from photographs, **source-system** data such as point-of-sale and inventory feeds, and **human judgment** captured through structured input. All of it belongs to the customer. So does the configuration layer that makes the platform valuable — your brand rules, operational taxonomies, and operating model. That configuration is isolated to your tenant, never commingled with another customer's, and never resold as a shared capability.

Isolation

- Tenant isolation is brand-scoped and enforced at the database via forced RLS (Section 5).
- Object storage for uploads and media is organized so that access follows the same tenant scoping.
- One customer's users, imagery, findings, and configuration are not visible to another customer.

Encryption & secrets

In transit	TLS across client, API, and internal service communication; HTTP Strict Transport Security enforced in production.
At rest	Managed encryption provided by the underlying Google Cloud data and storage services.
Runtime secrets	Held in Google Cloud Secret Manager — not embedded in source code or configuration files — with rotation and source-of-truth discipline.
CI/CD secrets	Scoped separately to the build system, distinct from runtime application credentials.

DATA MINIMIZATION IN AI CALLS

Because AI processing involves sending data to a provider, ReStore's design intent is to send only what a given task requires — the relevant image and the specific business context — rather than broad extracts. The full detail of AI egress is in Section 9.

Application & transport security

Runtime protections on the API and hardened browser surfaces, validated at startup.

API runtime protections

- **Rate limiting** to constrain abusive or runaway request volume.
- **Query depth and complexity limits** to prevent expensive or malicious GraphQL queries.
- **Production error sanitization** so internal detail is not leaked in responses.
- **Structured request and error logging** for traceability.

Browser & HTTP hardening

The API and web surfaces set a strong header baseline, including Strict-Transport-Security (production), X-Content-Type-Options, X-Frame-Options, Referrer-Policy, Permissions-Policy, and Content-Security-Policy variants tuned per surface.

CORS & environment validation

- Explicit origin allowlists in production; wildcard origins are blocked by startup validation.
- Permissive development behavior is isolated to non-production mode only.
- Production startup validates critical configuration, including HTTPS-only identity endpoints — the service refuses to start misconfigured.

SECTION 08

How data flows through the platform

Three primary paths: the authenticated request, asynchronous background work, and upload & photo ingestion.

The authenticated request path



Figure 2 — Every request is verified and then scoped to the caller's tenant before any data is touched.

The asynchronous work path

- Operations that should not block the user enqueue background work onto the queue.
- Workers process jobs with retry, backoff, and dead-letter handling for durability.
- Workers write state and audit artifacts and trigger downstream steps.
- Notifications and AI analysis run here — off the user-latency path.

Upload & photo ingestion

Uploads are validated by content (magic-byte checks), have filenames sanitized, and are size-verified after upload against declared values. Photos then move through a nine-stage conveyor, where stage isolation limits blast radius and supports precise, surgical retries:

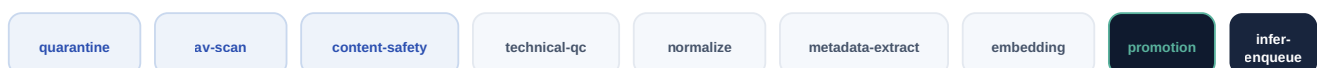


Figure 3 — The nine-stage ingestion conveyor. Safety stages (quarantine, AV, content-safety) run before any analysis.

AI data handling & external egress

This is the question most technical buyers care about most: what exactly leaves the platform, where does it go, and how is it controlled?

AI is applied in stages, not a single black box

ReStore does not make one opaque call to "an AI." Intelligence is applied across distinct operational stages, and different providers do different jobs. In a representative flow, **Anthropic Claude** composes the instruction that tells the image model what to look for and how to report it, and later **adjudicates** the results into structured findings; **Google Vertex / Gemini** performs the **multimodal image processing**. Provider and model are resolved through a governed model registry and can be set per brand — selection is configuration, not ad-hoc code.

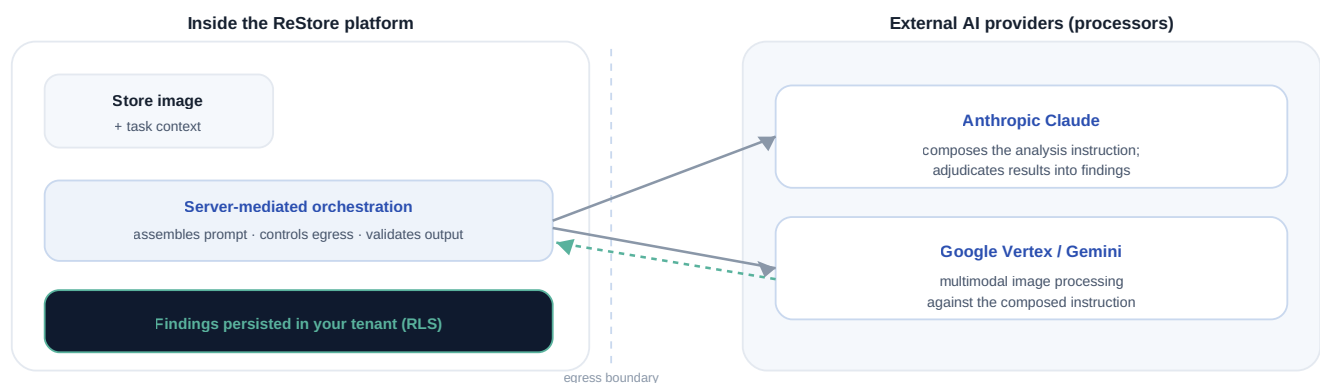


Figure 4 — Representative AI flow. Only the task-relevant image and context cross the egress boundary; results return for validation and tenant-scoped storage.

What is sent to AI providers

Depending on the flow, the payload may include:

- Prompt text assembled **server-side** from templates and the relevant context.
- Image data — frequently fetched and sent inline as part of a multimodal request.
- The specific business context a task requires to produce a useful result.

What is *not* done

- Clients do not pass data directly to providers — every call is server-mediated and controlled by the platform.
- There is no open pass-through of your database; calls carry task-scoped payloads, not broad extracts.
- Model outputs are validated against schemas before they are persisted or acted upon — the platform does not blindly trust generated content.

Controls on AI egress

Server-mediated	All provider calls originate from the platform, never from an unrestricted client path.
Output validation	Generated results are schema-checked before persistence or use.
Observability	Token usage and processing metrics are logged for monitoring and cost governance.
Governed selection	Prompt templates and model/provider choice are configuration-governed via a registry, overridable per brand.

RESIDENCY, RETENTION & TRAINING — THE PROCUREMENT VIEW

Because images and business context can be commercially sensitive, the precise treatment by each AI provider — data residency, retention windows, use-for-training terms, and subprocessor transparency — is governed by that provider's data processing agreement. ReStore's posture is to operate on enterprise terms that support these protections, and to make the relevant DPAs and the subprocessor list available for your review. Confirming these terms is a standard, expected part of commercial due diligence, and we support it directly.

SECTION 10

Notification & messaging security

Direct provider adapters, not opaque orchestration middleware.

Delivery architecture

Email · Resend

Push · Firebase Cloud Messaging

In-app · platform database path

SMS · deferred

- Adapters can run in an explicit stub mode when credentials are absent — useful for safe non-production environments.
- Push delivery prunes invalid device tokens automatically.
- Notification data paths remain auditable within platform services and queues.

SECTION 11

Infrastructure, deployment & secrets

Infrastructure defined as code, with disciplined secret management and a documented environment strategy.

Governance & environments

- Infrastructure is managed with Terraform; runtime changes flow through apply workflows to avoid drift between intended and actual state.
- The environment model and a phased isolation strategy are documented in an architecture decision record, with progression toward stricter separation as scale and compliance needs grow.

Secret management (tiered)

CI/CD authentication	Build-system secrets scoped to the build environment.
Runtime application	Held in Google Cloud Secret Manager, separate from build secrets.
Rotation & ownership	Rotation cadence and source-of-truth discipline documented in the environment/secrets decision record.

SECTION 12

Monitoring, resilience & operations

Operational visibility and disciplined failure handling.

- Structured logging across the API and services.
- Queue health and job-state observability.
- Durable retries with a dead-letter strategy for failed work.
- Documented "rainy-day" scenarios and remediation planning for external-service outages, pipeline integrity faults, and silent degradation modes.

SECTION 13

Transparency: design tradeoffs we disclose

These are deliberate engineering choices with documented context and compensating controls — disclosed up front, not discovered later.

1 • Privileged background-worker database path

Asynchronous jobs run without a user's token, so a dedicated worker client operates with elevated privilege per transaction.

Compensating controls: the path is intentional and explicit; worker code and query pathways are a high-scrutiny area in secure-SDLC review.

2 • Database TLS verification mode in private deployments

In one private-network deployment path, certificate verification is relaxed due to private certificate-authority trust-chain behavior.

Compensating controls: private networking isolates the path, and the infrastructure is Terraform-governed; the connection is not exposed to the public internet.

3 • Admin console content-security-policy tuning

The administrative web console's CSP currently permits inline scripts and styles — common operationally, and an opportunity to tighten further.

Compensating controls: the admin surface is access-controlled and hardened with the broader header baseline; CSP tightening is a tracked improvement.

These are not hidden issues. Disclosing them is part of how ReStore approaches security relationships.

SECTION 14

Compliance posture & roadmap

What is in place today, and what is on the near-term path.

In place today	SOC 2 certification for the current platform; secure SDLC practices; infrastructure-as-code governance; database-enforced tenant isolation; centralized secret management; documented architecture decision records.
On the roadmap	Re-certification on completion of the AI feature set, with the same SOC 2 controls applied throughout AI development; progression toward stricter environment separation as scale and customer needs require.
Available on request	Data Processing Addenda and subprocessor list for AI, email, and push providers.

Compliance status reflects the platform's current generation and is updated as milestones are reached; ask your ReStore contact for the latest status before relying on any item above.

SECTION 15

Evidence & next package for buyers

This overview is the front door. Deeper artifacts are available to qualified evaluators under NDA.

Control-to-evidence mapping

Tenant isolation & RLS	Database migration policies; API context-builder and RLS plugin source; platform standards documentation.
Identity & OIDC	Identity-provider configuration, claims construction, and role model source.
Application protections	Rate-limit and query-governance plugins; environment validator; security header configuration.
Upload & ingestion	Upload-security module; ingestion-pipeline definition; antivirus/threat-detection decision record.
AI orchestration	Model registry and orchestration source; photo-analysis services; AI framework decision record.
Infrastructure & secrets	Terraform configuration; environment/secrets and deployment strategy decision records.



Let's go deeper.

This overview is designed to answer the first wave of questions quickly. Your ReStore contact can arrange a technical deep-dive with our engineering team, provide control evidence and architecture decision records under NDA, and return a completed security questionnaire mapped to the sections in this document.