# ✅ Canonical MVP Playbook — v2.5

This is the authoritative document for MVP development. If you see another canvas doc, treat it as **ARCHIVED – refer to this MVP Playbook v2.5**.

# MVP Pending Items as of 05-Sep-2025

*Last updated: 08-Sep-2025*

## Changelog (v2.5 quick hits)

* Marked this document as the **canonical MVP Playbook — v2.5**.
* Integrated security updates: Widget Key Gate (optional, per-tenant), X-Debug-Key guard on /debug/\* in prod, step-up auth for sensitive actions, MFA/magic-link + OTP plan, immutable audit logs.
* Added Tenant Onboarding & Offboarding pilot flows (endpoints + acceptance checks) and clarified user lifecycle (invite/verify/disable/transfer owner).
* Confirmed **embeddings on approve = true** and documented reembed\_missing backfill.
* Clarified repo separation (runtime vs admin API vs admin UI) and deployment order for Cursor.

## 1) Executive Summary

**Objective:** Ship a dependable RAG runtime with trustworthy citations, and a lightweight Admin UI for document ops, RAG hygiene, and inbox triage.

**Status:**

* **Runtime**: Hybrid retrieval (BM25 + vectors), concise synthesis, correct fallback to model+inbox when docs are insufficient, citation validation/metrics.
* **Admin API**: Docs stats/upload/reembed; Inbox list + approve/reject; CORS enabled.
* **Admin UI (Next.js)**: Phase 2 & Phase 3 complete (proxy routes, Docs page). Phase 4 (RAG Hygiene) functional with polish items pending. Phase 5 (Inbox) next.

**Why it matters:**

* Faster operations (self-serve upload & re-embed)
* Higher answer trust (validated citations)
* Clear QA workflow (RAG probe, Try /ask, Inbox triage)
* Secure by design (server-side proxies; tokens never in browser)

## 2) Architecture at a Glance

* **Runtime (Ask Service):** /ask, /debug/rag, /debug/rag\_env, /debug/metrics.
* **Admin API:** /admin/docs/\*, /admin/inbox\*.
* **Admin UI (Next.js, App Router, TS, Tailwind, shadcn/ui):**
  + **Server-only helpers:** src/lib/api/admin.ts
  + **Client-safe helpers:** src/lib/api/ask.ts
  + **Proxy route handlers (server):** src/app/api/admin/\*\* → forward to Admin API with Authorization & X-Tenant-Id.
  + **Pages:** /admin/docs, /admin/rag, /admin/inbox.

**Security:** admin token only used in server handlers; CORS allowlist on Admin API; runtime public endpoints remain tenant-scoped.

## 3) Environments & Config

### 3.1 Runtime (.env)

RAG\_TOPK=6  
DOC\_VEC\_W=0.6  
DOC\_TRGM\_W=0.4  
DOC\_MIN\_SCORE=0.22  
  
ASK\_ENABLE\_SYNTH=1  
ASK\_TOPK=6  
ASK\_MIN\_SCORE=0.25  
ASK\_MMR\_LAMBDA=0.7  
ASK\_MAX\_CHUNKS\_PER\_DOC=2  
ASK\_MAX\_DOCS=3  
ASK\_REQUIRE\_CONTEXT=true  
QA\_SIM\_THRESHOLD=0.82  
  
RAG\_ALPHA=0.60  
SYNTH\_MODEL=gpt-4o-mini  
  
CITES\_VALIDATE\_ENABLE=1  
CITES\_TOPN=2  
METRICS\_ENABLE=1  
  
SYNTH\_UNINFORMATIVE\_MIN\_LEN=18  
SYNTH\_UNINFORMATIVE\_PATTERNS=i don't know|dont know|insufficient context|not found in (?:the )?(?:document|documents|docs|kb)|no (?:relevant|sufficient) information|unable to determine|cannot determine|not enough information|no answer found|couldn'?t find|cannot find|the documents? (?:do|does) not contain

### 3.2 Admin UI (.env.local)

**Client+Server**

NEXT\_PUBLIC\_ASK\_BASE=https://ask-abilitix-api.onrender.com  
NEXT\_PUBLIC\_TENANT\_SLUG=abilitix

**Server-only**

ADMIN\_BASE=http://127.0.0.1:8001  
ADMIN\_TOKEN=<local\_admin\_bearer>  
TENANT\_ID=a2fa67c0-d0e5-42db-8d5a-845637190732

### 3.3 Dev server binding

* Localhost-only: npm run dev -- -H 127.0.0.1 -p 3000
* Windows Firewall: allow **Private**, block **Public** (or bind to 127.0.0.1 to avoid LAN exposure).

## 4) APIs & Headers

### 4.1 Runtime (Ask)

* POST /ask (header: x-tenant-slug)
* GET /debug/rag?tenant\_id=...&q=...&topk=...
* GET /debug/rag\_env
* GET /debug/metrics

### 4.2 Admin API

* Headers: Authorization: Bearer <token>, X-Tenant-Id: <uuid>
* Docs: /admin/docs/stats, /admin/docs/upload\_text, /admin/docs/reembed\_missing
* Inbox: /admin/inbox, /admin/inbox/{id}/approve, /admin/inbox/{id}/reject

### 4.3 Admin UI Proxy (Next.js)

* GET /api/admin/docs/stats → Admin API
* POST /api/admin/docs/upload\_text → Admin API
* POST /api/admin/docs/reembed\_missing → Admin API
* Inbox proxies added in Phase 5.

## 5) Delivered (Stage 3 so far)

1. **Citations Fix Pack**
   * Unified synthesis+citations; validation enabled; metrics report 0 bad.
2. **Fallback logic**
   * Uninformative synthesis → model fallback → inbox capture.
3. **Admin API CORS**
   * Allow localhost UI + prod origin.
4. **Admin UI Foundation**
   * Next.js app, helpers, proxy routes ✅
5. **Docs Page (/admin/docs)**
   * StatsCard, UploadForm, ReembedButton; auto-refresh; toasts; works end-to-end ✅
6. **RAG Hygiene Page (/admin/rag)**
   * Functional: query, results table, Try /ask with 2 citations; polish ongoing ✅

## 6) Remaining Scope (to wrap Stage 3)

### Phase 4 – RAG Hygiene (polish)

* Format score/vec\_sim/trgm\_sim to 3 decimals; show top score badge.
* Show match.similarity or – if absent.
* Loading/disabled states + small spinner.
* Copyable citation chips (include citation score).
* Persist query/topK in URL search params.
* Clamp answer to ~6 lines; copy button; badge source (docs.rag, model+inbox\_pending, qa.model).

**Acceptance:** /admin/rag renders cleanly; RAG table + Try /ask work; 2 citations; robust loading/error states.

### Phase 5 – Inbox

**Proxy routes (Next.js):**

* GET /api/admin/inbox?status=pending|approved|rejected
* POST /api/admin/inbox/approve { id }
* POST /api/admin/inbox/reject { id }

\*\*Page \*\*``

* Table: Question | Created | Status | PII | Actions
* Default filter: pending; selectors for approved/rejected.
* Approve/Reject buttons with loading/disabled + toasts; row updates in place.

**Acceptance:** Negative prompt to /ask creates pending item; appears in UI; approve/reject succeed; no secrets in browser.

### Nice-to-have (non-blocking)

* **Docs page**: .txt/.md client-side file → prefill textarea → submit via /upload\_text.
* Styling pass with shadcn/ui components (cards/tables/buttons).

### Deferred (Stage 3.5+)

* **Doc-window merge & de-dup** behind env flags; /debug/rag\_windows; metrics for merged/deduped windows.
* **File uploads** (PDF/DOCX) via /admin/docs/upload\_file (multipart) with server-side parsing, limits, and sanitization.

## 7) Smoke Tests (Windows-friendly)

### 7.1 Runtime/Ask

curl -s -X POST "https://ask-abilitix-api.onrender.com/ask" ^  
 -H "Content-Type: application/json" ^  
 -H "x-tenant-slug: abilitix" ^  
 -d "{\"question\":\"What is our refund policy?\",\"session\_id\":\"ui-rag-test\"}"

curl -s "https://ask-abilitix-api.onrender.com/debug/rag?tenant\_id=a2fa67c0-d0e5-42db-8d5a-845637190732&q=refund%%20policy&topk=8"

### 7.2 Admin UI Proxy

curl -s http://127.0.0.1:3000/api/admin/docs/stats

curl -s -X POST http://127.0.0.1:3000/api/admin/docs/upload\_text ^  
 -H "Content-Type: application/json" ^  
 -d "{\"title\":\"UI Upload Test\",\"text\":\"seeded from UI route\"}"

curl -s -X POST http://127.0.0.1:3000/api/admin/docs/reembed\_missing

### 7.3 Inbox workflow

curl -s -X POST "https://ask-abilitix-api.onrender.com/ask" ^  
 -H "Content-Type: application/json" ^  
 -H "x-tenant-slug: abilitix" ^  
 -d "{\"question\":\"When did we colonize Mars?\",\"session\_id\":\"ui-inbox-1\"}"

curl -s "http://127.0.0.1:3000/api/admin/inbox?status=pending"

curl -s -X POST "http://127.0.0.1:3000/api/admin/inbox/approve" ^  
 -H "Content-Type: application/json" ^  
 -d "{\"id\":\"<INBOX\_ID>\"}"

## 8) Troubleshooting Quick Reference

* **Dev server shows 404 on proxy routes** → Ensure files are under src/app/api/... and named route.ts. Restart dev or clear .next.
* **Token leaked to browser?** → Never import src/lib/api/admin.ts in client components; proxy calls only.
* **CORS errors** → Update Admin API CORS allowlist for UI origin/port changes.
* **Windows firewall prompt** → Prefer binding Next.js to 127.0.0.1. If LAN access needed, allow **Private** only.
* **Uninformative doc answers** → Confirm SYNTH\_UNINFORMATIVE\_\* regex/len are set; fallback to model+inbox\_pending should trigger.
* **Citations count** wrong → Check CITES\_TOPN (runtime env) and UI rendering caps.

## 9) Readme Notes / Runbook

* **Start Admin UI (localhost-only)**: npm run dev -- -H 127.0.0.1 -p 3000
* **Admin UI env**: .env.local with ADMIN\_BASE, ADMIN\_TOKEN, TENANT\_ID, NEXT\_PUBLIC\_\*.
* **Tenant values**: slug abilitix, id a2fa67c0-d0e5-42db-8d5a-845637190732.
* **Runtime metrics**: GET /debug/metrics → monitor docs\_answered, fallback\_model, cites\_good/bad.
* **Interim Refunds doc**: present; answers refund/cancellation Qs with 2 citations.

## 10) Definition of Done (Stage 3)

1. /admin/docs complete and reliable (stats/upload/re-embed; polished UX).
2. /admin/rag functional + polish shipped (table, Try /ask, citations, states).
3. /admin/inbox live (pending list; approve/reject; toasts; updates).
4. Security maintained (proxies, no secrets in browser, CORS OK).
5. Observability OK (cites\_bad=0; counters move during tests).
6. Smoke suite documented and passing on local + Render.

## 11) Backlog (Stage 3.5+)

* Doc-window merge & de-dup (feature-flagged) with /debug/rag\_windows and metrics.
* File upload ingestion (PDF/DOCX) endpoint with server-side parsing, size limits, and sanitization.
* Operator audit log and basic auth for Admin UI (optional if exposed beyond localhost).

## 12) Document Lifecycle (Additive)

### Wave A — Archive & Validity (Stage 3.5, tiny)

**Goal:** Safely turn off stale docs; retrieval ignores archived/expired content.

**DB (documents):** add status text default 'active', valid\_from timestamptz null, valid\_to timestamptz null.

**Runtime:** filter active/valid docs only (env-gated):

RETRIEVAL\_REQUIRE\_ACTIVE\_DOCS=true  
RETRIEVAL\_REQUIRE\_VALIDITY=true

**Admin API:**

* POST /admin/docs/archive { id, reason? }
* POST /admin/docs/unarchive { id }
* PATCH /admin/docs/{id}/metadata { valid\_from?, valid\_to? }

**Admin UI:** On /admin/docs, add Status pill + Archive/Unarchive + Edit validity modal.

**Smokes:** Archive → ask no longer cites it; Unarchive → ask cites again. Set valid\_to in past → ignored by retrieval.

### Wave B — Supersede & Versioning (Stage 4)

**Goal:** Prefer the newest truth per topic; keep audit trail.

**DB (documents):** add topic\_key text, version text, superseded\_by uuid (+ index on (tenant\_id, topic\_key, status)).

**Admin API:**

* POST /admin/docs/supersede { new\_doc\_id, obsolete\_doc\_ids[], reason? }
* PATCH /admin/docs/{id}/metadata { topic\_key?, version? }

**Admin UI:** On /admin/docs, show topic\_key, version; add **Supersede…** dialog to mark older docs superseded by the new one.

**(Optional)** Freshness boost knobs:

FRESHNESS\_BOOST\_ENABLE=false  
FRESHNESS\_BOOST\_HALF\_LIFE\_DAYS=180

**Smokes:** Upload v2 (same topic\_key), supersede v1 → only v2 retrieved/cited. Restore by changing status if needed.

## 13) Smart Executables (Stage 4+)

**Purpose:** Lightweight, per-tenant jobs that analyze usage and content, propose actionable suggestions, and apply them with one click. No need to build an LLM; use heuristics first, then small, budget-capped model calls.

### Phased approach

* **Phase A — Heuristics-first (no model cost)**
  + *Insights:* Gap Finder (top unanswered intents), Alias/Synonym suggestions, Doc Staleness/Conflicts, Threshold Tuner (safe recommendations), RAG Hygiene Watch (frequent low-score hits).
  + *Storage:* tenant\_insights(id, tenant\_id, kind, payload\_json, created\_at, status=pending|applied|dismissed).
  + *Admin UI:* /admin/insights page with cards (Apply / Dismiss / View details).
  + *API:* POST /admin/insights/run, GET /admin/insights?status=..., POST /admin/insights/{id}/apply, POST /admin/insights/{id}/dismiss.
  + *Worker:* background schedule every 6–12h per tenant (pure SQL/heuristics).
* **Phase B — LLM‑lite assist (capped & optional)**
  + *Executables:* Draft Q&A from docs (top gaps, 2 citations), Draft doc outlines, Canonical intent rewrites (aliases).
  + *Budgets:* ASSIST\_ENABLE=0|1, ASSIST\_MAX\_CALLS\_PER\_DAY, ASSIST\_MAX\_TOKENS\_PER\_DAY (small values, per-tenant).
  + *Human-in-the-loop:* Drafts appear in Inbox as “Assistant Draft” for approval; never auto-publish.
* **Phase C — One‑click Actions**
  + *Actions:* alias.create\_many, docs.archive\_many / docs.supersede, settings.update, qa.draft\_many.
  + *UX:* Apply opens a diff preview → confirm → execute via server proxies.

### Data & Config

* *Tables:* tenant\_insights (required), tenant\_profiles (optional rolling stats).
* *Env:* INSIGHTS\_ENABLE=1, INSIGHTS\_SCHEDULE\_CRON="0 \*/6 \* \* \*", ASSIST\_ENABLE=0, ASSIST\_MAX\_CALLS\_PER\_DAY=10, ASSIST\_MAX\_TOKENS\_PER\_DAY=20000.
* *Per‑tenant overrides:* enable/disable and budgets can live in tenant\_settings.overrides.

### Safety & Governance

* Opt‑in per tenant; human approval required; PII‑safe (use sanitized fields); zero‑downtime (purely additive endpoints + worker); audit via tenant\_insights records.

### Quick smokes (examples)

:: Run insights now for a tenant  
curl -s -X POST "http://127.0.0.1:3000/api/admin/insights/run" ^  
 -H "Content-Type: application/json" ^  
 -d "{\"tenant\_id\":\"%TENANT\_ID%\",\"kinds\":[\"gap\_finder\",\"alias\_suggest\",\"doc\_staleness\",\"threshold\_tune\"]}"  
  
:: List pending insights  
curl -s "http://127.0.0.1:3000/api/admin/insights?status=pending"  
  
:: Apply a suggestion (copy id from list)  
curl -s -X POST "http://127.0.0.1:3000/api/admin/insights/<INSIGHT\_ID>/apply"  
  
:: (Optional) Enable assistant budgets for a tenant  
curl -s -X PATCH "http://127.0.0.1:3000/api/admin/tenants/%TENANT\_ID%/settings" ^  
 -H "Content-Type: application/json" ^  
 -d "{\"ASSIST\_ENABLE\":1,\"ASSIST\_MAX\_CALLS\_PER\_DAY\":5}"

**Outcome:** Abilitix “learns” per tenant by turning analytics into proposals and safe, one‑click actions, starting with free heuristics and layering in small model calls only where they save real time.

Security Posture & Controls (v2.5 update) Status legend: ✅ shipped now (Stage 3) · 🔜 planned Stage 4 (near‑term) · 🧭 backlog/optional

Identity & Access • ✅ Admin API auth via Bearer token + X‑Tenant‑Id; all DB ops enforced by RLS and tenant claims. • 🔜 Per‑user JWTs with short TTL (exp), roles (owner/admin/curator/viewer), and email verification before any admin access. • 🔜 Step‑up auth (magic link/OTP re‑verify) required for sensitive actions (key rotation, purge, ownership transfer). • 🔜 MFA enforcement (TOTP/magic‑link + OTP fallback) for Admin UI.

Keys & Runtime Gatekeeping • 🔜 Separate per‑tenant keys: Widget Key (runtime) and Admin Key (admin API). One‑time reveal on rotate; hashes stored; all rotations audited. • 🔜 Optional Widget Key Gate: /ask requires X‑Widget‑Key when enabled (tenant setting). • ✅ /ask, /debug endpoints require auth; • 🔜 Production /debug requires extra X‑Debug‑Key header in addition to auth & CORS.

Network & Transport • ✅ TLS 1.2+ in transit (Render/Supabase). AES‑256 at rest (Supabase storage, managed PG). • 🔜 WAF + IP rate‑limit profiles on Render/Cloudflare; tighten per‑route rate limits in runtime.

Data Protection & Privacy • ✅ No raw PII stored in knowledge items or chunks; uploads pass through sanitizer. • ✅ Citations and answers are derived from sanitized text; PII flags propagate to inbox. • 🔜 Retention setting per tenant (read‑only at pilot); default 90 days; delete/purge flows.

Audit, Logging, and Operations • ✅ Structured logs for key flows (ask.match, docs\_answered, cites\_checked). Metrics at /debug/metrics. • 🔜 Append‑only audit\_log table (answer\_approved/rejected, key\_rotated, doc\_uploaded, settings\_changed). • 🔜 Secrets rotation policy: every 90 days; documented playbook and checklist. • 🧭 RPO/RTO targets (≤15m / ≤4h) with quarterly restore drills; breach response runbook (Detect→Contain→Notify ≤72h).

KB & RAG Safety Gates (recap) • ✅ Docs RAG requires hybrid score ≥ DOC\_MIN\_SCORE and non‑uninformative synthesis. • ✅ KB trigram requires top\_trgm ≥ QA\_TRGM\_SOLO\_MIN. • ✅ KB vector runs only after docs fail and requires vec\_sim ≥ QA\_SIM\_THRESHOLD; generic follow‑ups are blocked.

Tenant Onboarding / Offboarding (pilot‑minimum) Minimal data model • tenants(id, name, slug, status{pending|active|suspended|closing|closed}, created\_at, deleted\_at) • users(id, tenant\_id, email, role{owner|admin|curator|viewer}, email\_verified\_at, disabled\_at, deleted\_at) • tenant\_settings(tenant\_id, key, value jsonb, PK (tenant\_id,key)) • audit\_logs(tenant\_id, user\_id, action, metadata, ts)

Self‑serve onboarding (additional thin layer on top of Stage 3) • 🔜 Sign‑up → create tenant(status=pending) + owner user; send verify link; on verify → status=active. • 🔜 Default security profile applied: strict review ON (locked for pilot), RLS enforced, keys issued. • 🔜 Welcome checklist: upload ≥1 doc → approve ≥1 Q&A → test widget.

Offboarding & export • 🔜 POST /tenants/{id}/cancel → locks widget, marks status=closing, offers export bundle (async job), 30‑day grace. • 🔜 POST /tenants/{id}/restore within grace; POST /tenants/{id}/purge after grace.

User lifecycle (tenant‑scoped) • 🔜 POST /users/invite, GET /auth/verify, POST /users/{id}/role, POST /users/{id}/disable|enable, DELETE /users/{id} (soft), POST /users/transfer\_owner.

Acceptance checks (pilot) • Create tenant → verify → Admin Console loads with checklist. • Invite curator → verify → can upload docs but cannot change settings. • “Close workspace” stops widget; restore works in grace; purge completes after grace.

Documentation Updates (to keep repo truths aligned) • HLSA.md → Add Security Posture matrix (Now/Stage 4/Backlog) and Tenant Lifecycle flows (onboard/offboard, user join/leave). Note RPO/RTO targets and breach response. • PRD.md → Expand “Admin Console” and “Tenant Settings & Keys” sections; add pilot vs MVP scope and acceptance criteria. • DEVELOPER\_GUIDE.md → Running three repos, envs, CORS, auth headers; key rotation flow; smoke tests; operations checklists (secrets rotation, restore drill). • Tenant Onboarding & Access Model (Pilot).md → one‑pager with endpoints, minimal schema, and curl tests (new doc).

Repo Layout & Deployment Order (for Cursor) • Runtime API (ask‑abilitix): ship gating + settings resolver first (Stage 4A). • Admin API (ask‑abilitix‑admin‑api): add /admin/tenants, /admin/settings, audit\_log writes. • Admin UI (admin‑ui): settings pages (per‑tenant overrides), docs lifecycle badges/actions. Sequence: 1) Admin API settings + audit → 2) Runtime settings resolver → 3) Admin UI settings → 4) Docs lifecycle actions → 5) Optional: Widget Key Gate + X‑Debug‑Key.

Cursor Execution Notes (immediate) • Implement Tenant Settings (DB + Admin API + Runtime resolver) with caching (TTL=60s) and expose via /debug/rag\_env. • Add optional Widget Key Gate and X‑Debug‑Key enforcement behind env flags; toggle per tenant. • Introduce audit\_logs table and write entries on approve/reject, upload, settings change, key rotation. • Prepare smoke tests for: settings override affecting /ask, widget key gating, audit log inserts, and docs supersede/delete/restore.

# Addendum – Security & Tenant Lifecycle (Updated for v2.5)

## Security Posture (current ➜ near‑term)

**What’s live now (Stage 3):**

* **JWT admin auth + RBAC/ABAC**: short‑lived JWTs for Admin API; roles (owner/admin/curator/viewer) enforced in code; RLS by tenant\_id across tables.
* **Separate keys**: distinct **Widget** (runtime) vs **Admin** (admin API) keys; used via headers; stored server‑side; never echoed.
* **HTTPS/TLS**: enforced end‑to‑end; Supabase at‑rest encryption (AES‑256 managed by provider).
* **No raw PII**: ingestion sanitizer on Admin API; inbox shows sanitized text only.
* **Rate‑limit baseline**: capture limiter (e.g., CAPTURE\_RATE\_LIMIT\_PER\_MIN) active in runtime.
* **Auditability (baseline)**: structured JSON logs for inbox insert, approvals, docs upload, citations/metrics.

**Coming next (Stage 4, additive & config‑gated):**

* **MFA everywhere (admin)**: passwordless **magic link + OTP**; enforce **email verification** prior to any admin access.
* **Widget Key Gate (optional, per‑tenant)**: runtime requires X-Widget-Key when enabled; key stored as **hash**; one‑time reveal on rotate.
* **Step‑up auth** for sensitive actions (key rotation, purge, ownership transfer): short‑TTL challenge via magic link/OTP.
* **WAF + enhanced limiting**: provider WAF + IP throttling; per‑tenant burst caps for /ask.
* **Immutable audit logs**: append‑only audit\_logs table with actor/tenant/action/metadata; export on request.
* **Debug shielding**: production /debug/\* requires X-Debug-Key header **in addition** to auth & CORS.
* **Secrets rotation**: rotate admin/widget keys and service secrets on a schedule (target ≤90 days), all rotations audited.
* **Resilience SLOs**: document **RPO ≤15 min / RTO ≤4 hrs**; quarterly restore drills.
* **Breach response**: playbook (Detect → Contain → Notify ≤72h) and runbook links in the repo.

**Env knobs (planned in Stage 4)**

* WIDGET\_KEY\_GATE\_ENABLE=0|1 (default 0)
* X\_DEBUG\_KEY=<random> (prod only)
* ADMIN\_JWT\_TTL\_MIN=15, STEP\_UP\_TTL\_MIN=10
* REQUIRE\_EMAIL\_VERIFIED=1 (admin)

## Tenant Lifecycle (Pilot‑grade)

**Onboarding (self‑serve or operator‑led):**

1. **Create tenant + Owner** → tenants{id,name,slug,status='pending'} and first users{role='owner'}.
2. **Verify email** (magic link/OTP) → status='active'; issue admin & widget keys.
3. **Default security profile**: strict review ON; RLS enforced; short‑lived JWTs; keys scoped to tenant.
4. **Welcome checklist**: Upload 1–2 docs → approve 1 Q&A → test /ask widget.
5. **Settings (pilot subset)**: branding (name/logo), read‑only thresholds, key rotate, contact email.

**Offboarding:**

* **Start cancellation** (POST /tenants/{id}/cancel): lock new Q&A; revoke widget key; status='closing'; offer export bundle.
* **Restore** within grace (POST /tenants/{id}/restore).
* **Finalize purge** (POST /tenants/{id}/purge): soft‑delete rows, schedule physical purge per retention; audit tenant\_purged.

**User lifecycle (within a tenant):**

* **Invite/verify** (POST /users/invite → verify endpoint). Roles: owner, admin, optional curator/viewer.
* **Disable/enable** (POST /users/{id}/disable|enable), **soft‑delete** (DELETE /users/{id}).
* **Ownership transfer** (POST /users/transfer\_owner) with **step‑up auth**.

**Minimal data model additions:**

* tenants: status {pending|active|suspended|closing|closed}, deleted\_at.
* users: email\_verified\_at, disabled\_at, deleted\_at.
* tenant\_settings: widget\_key, admin\_key, logo\_url, allowed\_domains[], retention\_days.
* audit\_logs: tenant\_id,user\_id,action,metadata,ts (append‑only).

## Docs Lifecycle (clarified)

* doc\_status ENUM('active','superseded','deleted'), superseded\_by, updated\_at/by.
* Retrieval excludes superseded/deleted by default; Admin UI shows badges and actions (supersede, restore, soft delete).

## Documentation updates (to be committed with Stage 4)

* **HLSA.md** → add “Security posture” and “Tenant lifecycle & trust gates” sections; diagram for key flows.
* **PRD.md** → “Admin Console” & “Tenant Onboarding (Pilot)” scope/flows; non‑goals.
* **DEVELOPER\_GUIDE.md** → setup of Admin UI + envs; running smoke tests; security headers (X-Widget-Key, X-Debug-Key).
* **Tenant Onboarding & Access Model (Pilot).md** → 1‑pager (endpoints, states, acceptance tests).

## Repo separation & execution order (Cursor‑ready)

1. **Admin UI (Next.js)** – separate repo (admin-ui):
   * Add **Settings** (tenant overrides), **Tenants** list, **Onboarding wizard** (welcome checklist).
   * Show audit excerpts on action toasts.
2. **Admin API (FastAPI)** – repo ask-abilitix-admin-api:
   * Endpoints: /admin/tenants, /admin/settings, lifecycle ops (cancel/restore/purge/export).
   * Enforce verified‑email + role checks; write to audit\_logs.
3. **Runtime (FastAPI)** – repo ask-abilitix:
   * Tenant overrides fetch/cache; **Widget Key Gate**; /debug header guard; metrics tags per tenant.

**Acceptance smoke (to add when implemented):**

* Widget Gate: curl -H "X-Widget-Key: <bad>" → 401; with correct → 200.
* Debug Key: /debug/rag\_env w/o X-Debug-Key → 403 in prod; with header → 200.
* Settings override: set DOC\_MIN\_SCORE=0.45 for tenant; /ask shows stricter gating immediately (≤60s cache).