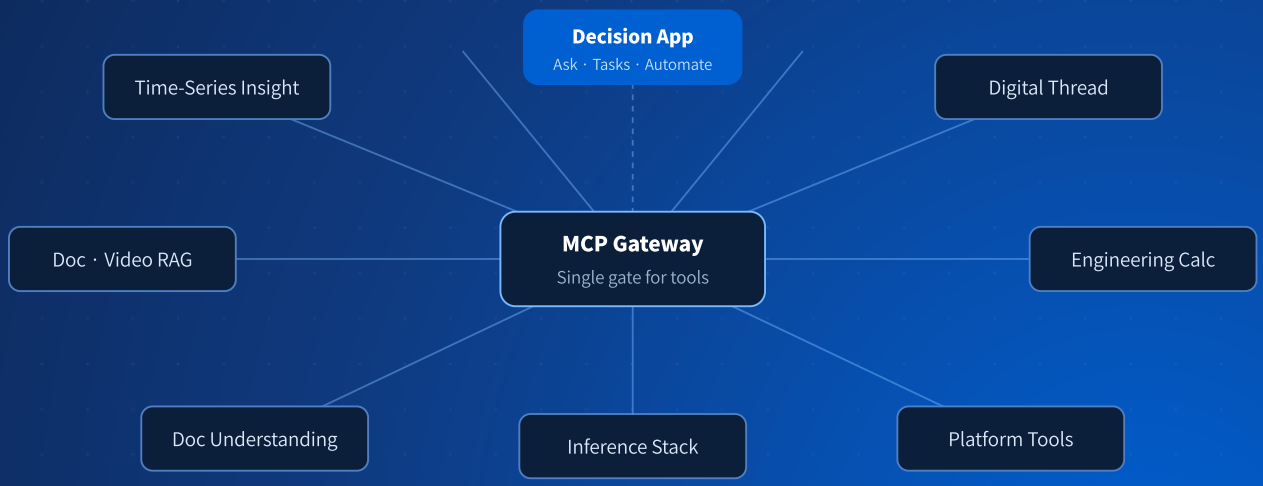
PlantPulse® **AI Platform**

TECHNICAL WHITEPAPER

TECHNICAL WHITE PAPER · REV. 2026.11 · DRAFT

Read through APIs, answer from evidence, and let people approve.

PlantPulse AI is an on-premises AI platform made of one **shop-floor decision app** and **several API modules** behind it. Each module has exactly one job and owns exactly one body of data, and the tools the model uses are reached only through **a single gateway**. This paper describes that **module structure** and the **safeguards** that keep answers correct and keep the AI from writing to the plant on its own.



22*
default services · one compose file

1
gate for all model tool calls

27
sandbox escape tests · release gate

Monthly
auto-release · signed air-gap bundle · unattended

Sources: plantpulse-ai-docker MANUAL § 1 (service count) · § 6.1 (escape tests · monthly release) · MCP Gateway MANUAL § 1 (single gate)
* Besides the 22 default services, 4 more services are enabled through profiles — plantpulse-ai-docker MANUAL «22 default services (+ 4 profile)»

November 2026

PlantPulse AI · on-premises / air-gapped · for technical reviewers

kopens®

00 · SUMMARY

Summary — three things AI must respect inside a plant

Where AI becomes dangerous in a plant is usually not the model's skill but its **wiring**. Attached directly to the ledger DB, it breaks silently when the schema changes; with tools scattered across services, nobody can see at a glance who may call what; and somewhere a **path for the model to edit work orders directly** is left open. PlantPulse AI blocks all three **by structure** — each module owns one body of data, tools pass through one gate, and writes to the plant go out only as cards a person has approved.

READ

Platform data via APIs

Ledger, time-series and events are read through APIs and tools the platform publishes. Direct DB access is being removed table by table with one switch each (Ch. 4).

WRITE

AI writes only to its own store

What the AI produces goes to the AI-owned `pi` store. In customer systems it creates only **drafts and requests**; a person confirms them.

ACT

Writes only via approval cards

The model has no path to execute anything itself. Proposal → human approval → execution → result record; high-impact actions need **two-person approval**.

Contents

01	Problem and approach — decision app · three principles	3
02	Overall structure — layers · gate · connections	4
03	Module map — one owner per body of data	5
04	Decision app — Ask · Tasks · Automate	6
05	External entry — public API · on-behalf	7
06	MCP gateway — tools · permissions · search	8
07	Time-series insight — findings · forecast	9
08	Digital thread — graph and connectors	10
09	RAG — documents · video · drawings	11
10	Document understanding · vision	12
11	Engineering calc · automation sandbox	13
12	Voice · file storage · long-term memory	14
13	Inference stack · admin console · edge proxy	15
14	Safeguards ① — evidence guard · «unknown»	16
15	Safeguards ② — scope · approval · audit	17
16	Deployment — on-prem · air-gapped · signed	18
17	Monthly release and hardware tiers	19
18	Evaluation ① — machine-scored benchmarks	20
19	Evaluation ② — measured · what's next	21
—	Contact	22

Basis and markings of this edition. The basis is the documentation and code in each module repository as of November 1, 2026. What is on by default is stated as is; what is implemented but off in the default install is marked OFF BY DEFAULT, and what ships in the next release is marked PLANNED. Only numbers the documents record as **measured** are carried over, each with a small source note (document · section). Configured limits are labeled «configurable».

Audience. Technical reviewers · information security · platform operators. This is not sales material — for each module it states **what it does, which APIs it exposes, and which data it owns**, in enough detail to review before installation. Training and evaluation of the models themselves are covered in a separate model technical white paper.

Terms used in this paper

`pi`	AI-owned store. One schema per app inside the PostgreSQL DB `pi`; analytics data in the Cassandra keyspace `pi`
MCP	Standard protocol by which a model finds and calls tools. On this platform the model reaches tools **only through one gateway**
Finding	Unit of judgment produced by time-series insight — numbers and fact codes, under a machine-readable contract (`findings/1`)
Shadow	State in which results are computed and recorded but not shown on the floor, only compared. Results reach the floor only after passing a gate
Approval card · L3 action	Proposed action that creates a draft or request in a customer system. The digital thread executes it only after a person approves
Scope	The plant buildings, lines and assets a person or key may see. If it cannot be determined, the answer is «not visible»

01 · PROBLEM AND APPROACH

Not a chatbot, but a «shop-floor decision app»

PlantPulse AI is an app that **finds problems in the plant and proposes fixes with evidence, lets people approve and close them, asks when it does not know, and hands repetitive work to automations**. Chat is just one of its entrances. It does not replace the customer's MES or ERP; it attaches beside them as a **layer that supports decisions**.

MENU ①
Ask
Chat · voice · photos. The start screen is a «today's work» summary — suggested tasks · anomalies to check · issue reports · automation suggestions.

MENU ②
Tasks
Planned work and incidents share one list. A single proposal flows through detection → approval → action → result → **recurrence check** on one detail page.

MENU ③
Automate
Industrial automation built in plain words. Build · dry run · run history · version history. Runs with its creator's permissions and uses read tools only.

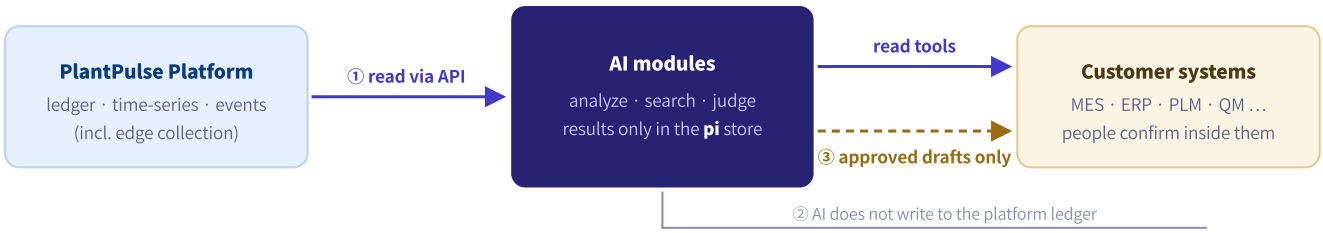
Sources: copilot BUSINESS-PROCESS § 1 · MANUAL § 1a (three menus — Ask `/chat` · Tasks `/tasks` · Automate `/automations`) · § 5.30 (automation)

We first decided what it is «not»

Not	Why
MES · ERP	The system of record for plans, work orders, actuals, lots and due dates is the customer's system. If the AI held a second record, one of the two would be wrong
Equipment control	Changing setpoints is out of scope. **The AI proposes; people decide**
BI dashboard	Screens that display numbers live in the platform. Here, numbers are linked to **decisions and actions**
Chatbot	Chat is one entrance. Even if the LLM stops, **deterministic screens** such as the task list and automations keep working

Sources: plantpulse-ai-master ECOSYSTEM § 11.0 (product position) · § 12.8 (decision app — «is not» · principles)

Three principles — where data comes from, and where it goes



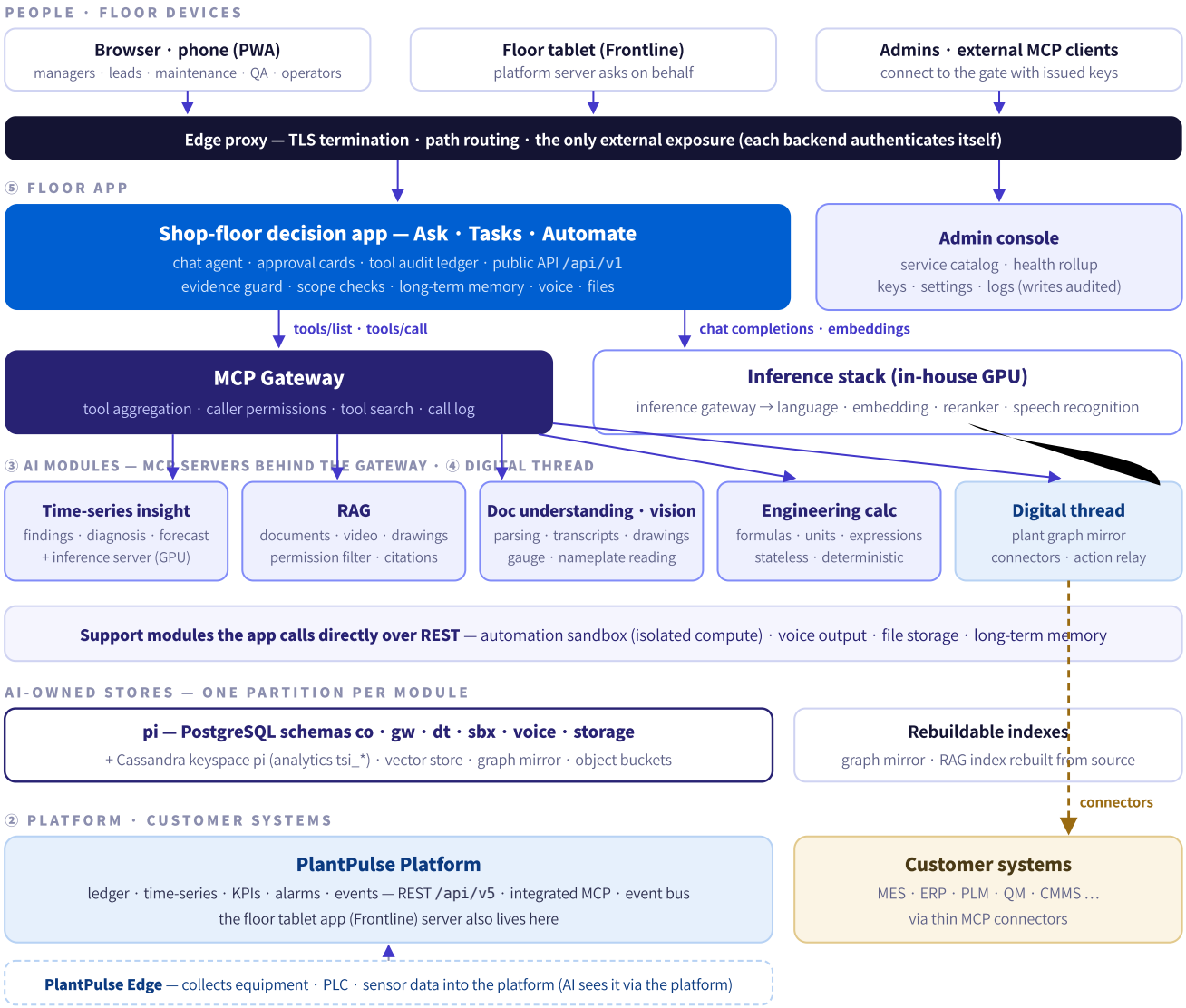
① Read via API	Platform ledger and events come through the platform REST API, the integrated MCP and the event bus (decided 2026-09-28). This is **in transition** — each table has a `db` · `api` · `shadow` switch (shadow reads both paths and compares), and tables move over once they are clean in shadow
② Write only to pi	AI data lives in the DB `pi` shared by the AI apps, with one schema per app. Writes on the platform side are limited to **one agreed exception** and one item scheduled to move into `pi`
③ People approve	Actions on the plant go proposal card → human approval → execution → result record. In customer systems only drafts and requests are created

Sources: plantpulse-ai-master docs/platform-data-via-api § 0 (five decisions) · ECOSYSTEM § 11.4-k (HITL) · § 11.7 (AI → platform writes, currently 2) · copilot PG-SCHEMA § 1 (`pi` schema rules)

02 · OVERALL STRUCTURE

Layers call only downward, and tools pass through one gate

There is **one proxy** open to the outside. User questions go to the decision app, which calls models only through the **inference stack** and tools only through the **MCP gateway**. Only **the digital thread** calls customer systems.



Sources: ECOSYSTEM § 1 (structure at a glance) · § 11.1 (layering · layer rules 1–4) · plantpulse-ai-proxy MANUAL § 1 · copilot MANUAL § 1 · v1-chat-on-behalf preamble (Frontline)

Where it touches the outside

Counterpart	Mechanism	Direction
PlantPulse Platform	REST `/api/v5` · integrated MCP (via the gateway) · event bus. Credentials are **obtained from the platform at startup**; if it is unreachable, the stack does not start	Read — some tables moving off direct DB via switches
PlantPulse Edge	No direct connection — through data loaded into the platform and platform tools (collection link status, etc.)	Read
Floor tablet	Platform tablet server → public chat v1 (`on_behalf`)	Question → answer, no write tools
Customer systems	Digital thread connectors (thin MCP servers)	Read · approved drafts only
External MCP clients	Gateway `/mcp/` + issued key (scoped)	Permitted tool calls

Sources: ECOSYSTEM preamble (credentials — received at startup) · § 11.2 · § 11.4 · docs/platform-data-via-api § 0 · copilot MANUAL § 4.4 · gateway MANUAL § 6.2

03 · MODULE MAP

One owner per body of data — where the fourteen modules sit

Each module is **one repository · one image** and writes only the data it owns. Other modules' data is read through the APIs and tools their owners publish. Every MCP server has a **name prefix** for its tools, so a tool's name alone tells you which module it belongs to.

Module	Job	Main API · MCP	Data it owns
Decision app	Ask · Tasks · Automate, chat agent, approval · audit	`/api/v1` · `/api/ui` · 3 read MCP tools	PG `pi.co` — users · conversations · proposals · results · tool audit ledger
MCP gateway	Tool aggregation · caller permissions · issued keys · tool search	`/mcp/` · `/api/v1/tools` · `/search`	PG `pi.gw` — servers · tools · permissions · keys · call log
Time-series insight	Anomaly · forecast · health index · findings feed · fault diagnosis	`/api/v1/findings` · `/forecast` · `insight_*`	Cassandra `pi.tsi_*` — baselines · findings · forecast ledger
TS inference server	Anomaly scores · forecasts · waveform embedding inference (GPU)	`/infer/*` · `/finetune/*` (internal only)	On-site fine-tuned weights (volume)
Digital thread	Plant graph mirror · connectors · impact range · action relay	`/api/v1/timeline` · `/impact` · `thread_*`	Graph `:Dt*` (rebuildable) · PG `pi.dt`
RAG	Indexing and search of documents · video · drawings, permission filter	`/api/v1/query` · 2 `rag_*` tools	State DB · vectors · object buckets (documents · video · page images)
Doc understanding	Parsing profiles · video transcripts · drawing parsing · revision diff	`/api/v1/understand` · `/transcribe` · 2 `document_*`	None (stateless)
Vision	Classify · detect · inspect · OCR · gauge reading · analyze	`/api/v1/read_meter` etc. · 5 `vision_*`	None (stateless — photos discarded after processing)
Engineering calc	Formulas · unit conversion · allow-listed expressions	4 `eng_*` tools	None (stateless)
Automation sandbox	Isolated execution of Python steps in approved plans	`/api/v1/run` (REST only)	PG `pi.sbx` — run records (no contents kept)
Voice output	Speakable-text normalization · per-sentence synthesis · cache	`/api/v1/speech`	PG `pi.voice` · sentence cache
File storage	Original bytes · retention classes · personal-data erasure · audit	`/api/v1/objects` · `/uploads`	PG `pi.storage` · original / derived buckets
Long-term memory	Storage and semantic search of user- and automation-scoped memories	`/memories` · `/search`	Own vector collection · change history
Admin console	Catalog · health · keys · settings · logs (admins only)	`/api/v1/{service}/…` relay	None (settings in install env)

Sources: each repository's MANUAL § 1 · § 4 · § 5 · ECOSYSTEM § 11.2 (ownership table) · repos.tsv. In addition, the operational components are the inference stack · edge proxy (Ch. 13) and the deployment hub (Ch. 16).

Contracts live with the owner

The owner holds the contract and consumers pin a version. Tests sit on both sides — the owner checks «my output matches the schema», the consumer checks «I can read the pinned version».

Calls go top-down

Lower layers do not know upper layers' addresses. Calls within the same layer exist **only if declared** — currently two (vision → doc understanding for OCR, RAG → insight for asset lookup).

Health measures «auth»

«Alive» means one authenticated read succeeded, not merely a connection. We set this rule after a 401 once showed up as UP.

Sources: ECOSYSTEM § 11.3 (contract rules) · § 11.1 (layer rules 1 · 2) · § 11.4-d (health)

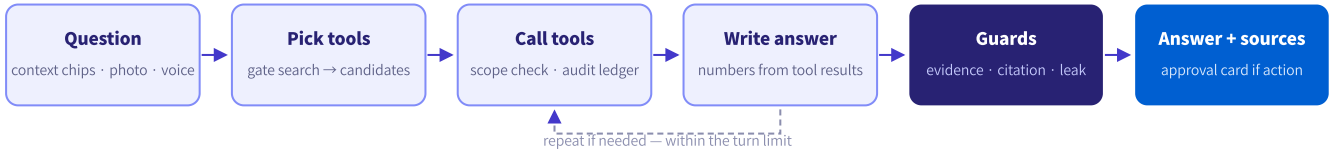
04 · DECISION APP

One place calls the model; people make the decisions

Form	**Web app · PWA**
Public contract	**/api/v1/****
Store	**PG pi.co**
Tool path	**One gateway**
Model path	**Inference stack**

The decision app is the only screen users meet, and the **orchestrator** that ties the analytics modules together. Its agent loop gives the model a tool list, calls the tool the model picks through the gateway, and feeds the result back to build the answer.

The path of one question



Main APIs

`/api/v1/**`	**The only public contract**. External systems use `X-API-Key` (stored only as a hash, revocation takes effect at once). It opens no path to «call» tools
`/api/ui/**`	Screen-only API — not a contract
`/admin/api/**`	Admin REST — settings · prompts · audit
`POST /mcp`	MCP the app «exposes» — `ask_plant` · `list_plant_tools` · `daily_tasks`, all three read-only

Data it owns — PG `pi.co`

Conversation · app	Users · chats · prompts (single source in the DB) · settings
Decisions	Proposals · results · issue reports · Episodes
Audit	Tool audit ledger (no expiry) · setting changes · view records
Secrets	AES-256-GCM at rest — **no plaintext in the DB**

Automations — built in words, run with the creator's rights

Users build an automation by filling five slots: «when · where · with what · how · to where». An automation runs with **its creator's permissions** and uses **read tools only** to produce proposal cards. Plan-based automations switch on **only once the plan is approved**, and write tools are **never executed** by automations. Each run has turn · time · token limits (configurable).

It only receives «findings»

The source of truth for anomaly · delay · quality signals is time-series insight's **findings feed**. The app validates what it receives against a byte copy of the contract (version `findings/1`) and adds only its own columns (who looked and what they did). To the digital thread it sends **references only** — never diagnoses, proposal text or notes.

Floor work continues even if the LLM stops. The task list, issue reports and automation run history are deterministic screens. The model handles explanations, summaries and drafts; people approve writes. The status of external dependencies (platform API · alarm channels · LLM) is reported separately as its own health layer.

Sources: copilot MANUAL § 1 · § 2 · § 4.3 · § 5.4 · § 5.6 · § 5.18 · § 5.30 · § 5.30.1 · ECOSYSTEM preamble (health layers) · § 11.3 (findings contract) · § 12.8 (principles)

What the app calls, and the limits set on it

Target	For what	How · limits (configurable)
MCP gateway	All of the model's tools	Tool search per question (budget 20). If search is down, the turn ends with an error — it does not silently load every tool
Inference stack	Chat completions	4,096 output tokens · tool results truncated to 24,000 chars · 5 consecutive failures block for 30 s
Time-series insight	Findings feed	REST cursor feed — validated against the contract copy
Sandbox · voice · files · memory	Compute · read-aloud · attachments · memory	REST — parts of the app, not model tools (Ch. 11 · 12)
Platform	Alarms · screen data	V5 REST — some in transition via the `db` · `api` · `shadow` switch

Sources: copilot MANUAL § 4 · § 4.4 · § 5.5 · § 5.21 · § 5.33

05 · EXTERNAL ENTRY

The floor tablet asks «on behalf of the operator» — with scope kept closed

The platform's floor tablet app (Frontline) does not attach AI directly to its screen. The tablet's server (BFF) **keeps the key on the server only**, calls the public chat API, and sends along via `on_behalf` who the asking operator is and how far they may see. PLANNED — NEXT RE-RELEASE

Public chat v1

`GET /api/v1/chat/capabilities`	Reports the contract version. An older server returns 404 → the consumer hides chat
`POST /api/v1/chat`	Answer in one response
`POST /api/v1/chat/stream`	Streamed answer over SSE
`POST /api/v1/chat/cancel`	Cancel the turn in progress

What a request carries

on_behalf	Employee no. · display name · role · **scope** (required)
context	The asset · order · shift · screen being viewed
options	Language · answer length (short / normal) · voice
Attachments	Photos (asset reading) · voice question and voice answer VOICE PLANNED

Scope is decided on the «closed» side

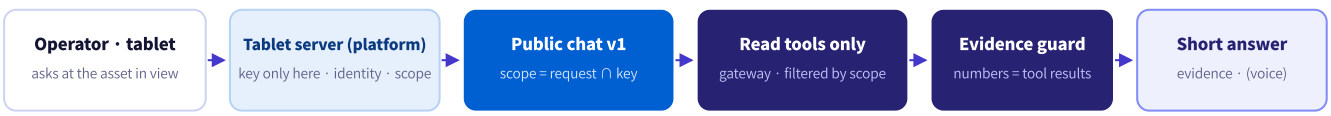
- `scope` is **exactly one** of `all` and `nodes`
- **An empty list is rejected** (400) — it is never read as «everything»
- At most 200 nodes; any unknown node yields 400 (configurable)
- The effective scope is **the intersection with the key's scope** — a request cannot widen it
- A conversation is owned by «key + operator». Other operators' conversations on the same key are invisible
- **No write tools are opened on this path** (the tool-call guard cannot be turned off)

Evidence guard on answers

If the answer claims data **while no tool succeeded** in this turn, the model is asked once more; if unchanged, the answer becomes «could not query, so cannot answer with figures». Numbers absent from tool results trigger one rewrite; any that remain get an «unverified number» warning line. **Earlier answers are not evidence** (Ch. 14).

Not writing to the platform is part of the contract. When platform tools are called during an on-behalf question, no «performed by» field is sent — because the AI does not write to the platform. The contract is versioned (`2026-10-30`) in agreement with the platform side and guarded by three tests on the owner side.

One on-behalf question



conversation owner = key + operator · empty scope or unknown node → 400 · older server → consumer hides chat

How it connects to the edge

The AI does not attach to the edge directly. Equipment data collected by the edge is loaded into the platform, and the AI sees it through **platform tools** (collection link status · latest tag values, etc.). Adding edges does not change the AI-side wiring.

Sources: copilot docs/design/v1-chat-on-behalf preamble (implemented · on staging, voice § 11 not yet, ships in the re-release) · § 0 · § 2.1 · § 5 · § 9.2 · § 10 · § 11 · MANUAL § 4.4 · ECOSYSTEM § 11.3 (AI public chat v1)

06 · MCP GATEWAY

Many tools, one gate

Protocol	**MCP (streamable HTTP)**
Exposed	**/mcp/ only**
Store	**PG pi.gw**
Tool prefix	**One per server**

It puts **a single address** in front of every MCP server in the installation — platform integration tools, built-in AI modules, and external servers registered by operators. It aggregates the lists, passes calls through **without altering results**, decides who may call what, and picks the tools that fit each question.

Main APIs

`/mcp/`	`tools/list` · `tools/call`. Only this path is opened to the outside; REST is not
`/api/v1/servers`	Server registration · auth · discovery · per-tool on/off
`/api/v1/tools`	Full tool inventory — source of truth for enabled state · permissions
`/api/v1/permissions`	Caller × server · tool permissions, effective-permission lookup
`/api/v1/api-keys`	Issued keys for external clients (only SHA-256 hashes stored)
`POST /api/v1/search`	Question → tool candidates (keyword + semantic hybrid)
`/api/v1/calls`	Call log — arguments · results · auth values are not kept

Closed is the default

- If the inbound key is empty, it **refuses to start**
- New issued keys are born **with no permissions and no scope**. A scope-less key calling built-in · platform tools is rejected (`forbidden_scope`)
- Issued keys reach `/mcp/` only — admin REST returns 403
- External servers · tools arrive **switched off** and are enabled after admin review. Tools suspected of writing cannot be enabled, and **they switch themselves off when their definition changes**
- External server credentials are sealed with AES-256-GCM; loopback · link-local · metadata addresses are refused

Tool search — the model's list stays short as tools grow

Small models pick tools worse the more tools they see at once. So for each question the gate selects candidates up to a **budget (default 20, configurable)**, and core tools (asset lookup · document search) are always included regardless of budget. Below are values measured with a 36-question tool-call bench over 76 tools.

Method	Recall@10	Recall@20	hit@1	hit@3	hit@10	MRR
Keyword (BM25)	0.806	0.944	0.722	0.917	0.972	0.816
Keyword + semantic hybrid	**0.889**	**0.972**	0.722	0.917	**1.000**	**0.828**

Scored through the app path, both budget 20 and 10 gave 36/36. Removing the search terms declared by servers drops keyword Recall@10 to 0.750 — **writing tool descriptions is search quality**. To stop keyword stuffing, there is a limit of 8 search terms · 64 chars per tool (configurable).

Sources: plantpulse-mcp-gateway-api MANUAL § 1 · § 2 · § 5.1 · § 5.2 (measured 2026-10-18 — 76 tools · 36 questions) · § 6.1 · § 6.2 · § 6.4 · CONTRACT § 2 · § 3

Tools enabled by default. Time-series insight 34 · digital thread 9 · vision 5 · engineering calc 4 · RAG 2 · doc understanding 2, plus platform integration tools. The release gate counts the gate's tool list directly and stops the release if insight falls below 34, thread below 9 or engineering calc below 4.

Sources: gateway MANUAL § 1 (on by default) · plantpulse-ai-docker MANUAL § 6.1 (`probe-gateway-tools` gate)

07 · TIME-SERIES INSIGHT

It emits numbers and fact codes — not sentences

MCP tools	**42**
Findings contract	**findings/1**
Store	**Cassandra pi.tsi_***
Inference	**Inference server (GPU · CPU)**

It reads equipment data (sensors · OEE · availability · energy · work orders · alarms) and produces anomalies · forecasts · health indices · findings. Model weights live in a **separate inference server**, which this module calls through a client with timeouts · retries · a circuit breaker.

Main APIs

`/api/v1/findings`	Findings feed — cursor-based, no gaps or overlaps; contract at `/findings/contract`
`/api/v1/anomaly`	Asset · system · OEE · availability · energy anomalies
`/api/v1/forecast`	Forecast — point + interval, with engine and version used
`/api/v1/intel/*`	Watch list · explanation · hypothesis test · fault types · graph
`/api/v1/asset/health`	Asset health index
`/api/v1/order/completion`	Work order completion forecast

The 42 MCP tools split into alarm · health 6 · analytics 18 · feature store 7 · intel 9 · profile 2. Tool descriptions are limited to 480 chars per tool · 8,000 chars total (configurable).

Findings contract — 15 kinds, machine-readable schema

Anomaly · work order shortfall · data gap · data quality · cascading stop · failure risk · limit reach expected · energy peak · OEE loss · SPC signal · setup time overrun · AI change · delay propagation · delay recovery, and more. A finding carries **numbers and fact codes** `{code, params, text}` and **inputs it could not use** `missing[]`. The app writes the sentences.

«Unknown» is not «not seen»

In a run where the source could not be read, open findings are **not closed**. Closing an unread source as «no anomaly» would make an asset nobody looked at count as healthy.

Diagnosis — the model proposes «hypotheses», the numbers give the «verdict»

Hypothesis test	Hypotheses from the LLM (max 3) are accepted only as **tags and expected shapes** (rising · falling · stuck · leading · decoupled) — no free text. Each symptom gets `supported` · `refuted` · `unmeasurable` with the supporting numbers. The score is a count, **not a probability**
Fault type diagnosis	A dictionary of 9 equipment types · 27 failure modes (mapped only to concept codes in international standards). To become a candidate, all key checks must be true and **at least two tags must have moved — a single sensor never names a cause**. Otherwise «insufficient cause candidates» with a reason code

Forecasting — answer with one engine only, and score every forecast issued

There is no silent fallback. With too little data it returns 422; if the auxiliary engine fails and the base engine answers, it says so with `degraded`. Every forecast issued stays in the **forecast ledger** (`pi.tsi_forecast_ledger`) to be checked against actuals later; if actuals cannot be obtained it is `unresolvable` — **never filled with 0 or scored approximately**.

> **New judgments start in shadow.** Live mode switches on only after passing a **pre-registered gate** and human approval, and who changed what and when is recorded. Shadow findings are excluded from chat tools — so unverified judgments are not passed on as fact.

Sources: plantpulse-timeseries-insight-api MANUAL § 1 · § 3 · § 5 (generated block of 42 tools) · INTEL «findings contract» · «fault hypothesis test» · «fault type dictionary · diagnosis» · «forecast engines» · «forecast ledger» · «live mode» · insight-server MANUAL § 1 · § 5

08 · DIGITAL THREAD

Values stay at the source; only the «links» are indexed

Graph	**Reference mirror :Dt***
Refresh	**Periodic sync per series freshness**
Store	**PG pi.dt**
Customer systems	**Only this module calls**

A plant graph on a time axis. It links in one graph the **keys and relations** of the platform ledger, objects from customer systems attached via connectors, and **references** to events · findings · proposals. It keeps no payload copies, so it can **always be rebuilt** from the sources.

Main APIs

`/api/v1/timeline`	Chronological events · findings for an object
`/api/v1/neighbors` · `/trace`	Follow links
`/api/v1/impact`	Impact range — part delay → job → customer order due date
`/api/v1/readiness`	Job readiness verdict (Go / No-Go)
`/api/v1/connectors`	Spec · registration · discovery · conformance · binding · read · action
`/api/v1/optimize/recovery`	Recovery plan computation · schedule scenario «draft»

The tools exposed to chat are fixed to **a few** (`thread_get_timeline` · `thread_trace_links` · `thread_impact_range` , etc.). However many connectors are attached, the number of tools the model sees does not grow.

Standard connector spec — thin MCP servers

Customer systems attach as **thin MCP servers** following per-category specs (MES · ERP · PLM · QM · CMMS · tooling · warehouse · lab · EHS, etc.). A connector must pass a **conformance test** before registration, and bindings declare «external key ↔ our key». Connector secrets are stored encrypted, with no plaintext fallback.

Commercial product adapter	Status
SAP S/4HANA	Passed the spec conformance test **against mock servers. Verification against the real products happens at pilot sites** PLANNED
IBM Maximo	
Siemens Teamcenter	
Dassault 3DEXPERIENCE	

The only path that «writes» to customer systems — approved L3 actions

Five required approval fields	`proposal_id` · `approved_by` · `approved_at` · `reason` · `idempotency_key` . If any is missing, nothing runs
What it creates	**Drafts and requests only** — supplier expedite request drafts, maintenance requests, schedule scenario drafts. It never confirms or publishes a plan
Admin console	Cannot call action tools. Actions start only from approval cards in the app
«Unknown» is not Go	In readiness verdicts, items whose source could not be read do not count as passed, and stale connector data yields no numbers (`stale_inputs`)

> **The ledger is read in read-only transactions.** The DB path to the platform ledger uses `READ ONLY` transactions, and the path via the platform V5 API sits behind a switch. The DB path is removed only after 14 clean days of shadow comparison (in transition).

Sources: plantpulse-digital-thread-api MANUAL § 1 · § 2 · § 4 · § 5 · § 6.3.1 · § 6.4 · § 6.5 · § 6.14 · plantpulse-dt-connectors FEATURES «what is real» · ECOSYSTEM § 11.5 · § 11.4-m

Three things the graph holds — and what it does not

Nodes	Examples	Original lives in
Ledger key projection	Sites · assets · tags · asset relations	Platform ledger
Connector objects	Work orders · operations · parts · lots · serial numbers · PO lines · suppliers · drawings · inspection characteristics · maintenance orders	Customer systems
Time-axis references	Events · insight findings and learned links · app Episodes · proposals (`id` only)	Platform · insight · app

Relations carry names such as «requires» · «qualifies» · «assigned» · «changes» · «component» · «nonconformance» · «blocks» · «inspects» · «binding». Not held — event summaries · attributes · values returned by connectors (read from the source on demand), analytics (computed by insight), human decisions (held by the app). Questions the ledger alone can answer go to platform tools.

Sources: digital-thread-api MANUAL § 5 · § 6.4 · § 6.5 · § 6.12 · ECOSYSTEM § 11.5 (what it does · does not do) · § 11.2 O9

09 · RAG

Permissions are filtered «before the answer is written»

MCP tools	**2 rag_ (read)**
Sync	**With ledger every 600 s**
Video	**Off by default**
Drawings	**Off by default**

It indexes equipment documents · manuals · SOPs and answers questions with evidence. The graph-based retrieval engine sits **only behind this gate**, so no consumer calls the engine directly. The UI lives in the admin console.

DOCUMENTS ON BY DEFAULT

Follows the ledger's documents

Compares the platform ledger's equipment document list and status to apply new · changed · removed items. Per-run and file-size limits apply (configurable).

VIDEO OFF BY DEFAULT

Transcript → segments → scene chips

Transcribes audio, indexes it in segments of about 120 s, and attaches **scene time chips** (start · end · thumbnail) to answers. Processing runs only in the night window.

DRAWINGS OFF BY DEFAULT

Title block · BOM · dimensions

Extracts title block · BOM · zones · dimensions from DXF · vector PDF drawings and ranks exact dimension · number matches first. Supports **revision diff**.

Sources: plantpulse-rag-api MANUAL § 1 · § 4 (`MEDIA_INGEST_ENABLED` · `DRAWING_INGEST_ENABLED` default false) · § 5 · § 9 · plantpulse-ai-docker MANUAL § 5.10

Main APIs

`POST /api/v1/query`	Question → answer + sources (citations `[n]`)
`/api/v1/documents` · `/sync`	Index status · ledger sync runs
`/api/v1/docs/...`	Upload · reindex · exclude · download
`/api/v1/grades`	Document classification
`/api/v1/media/...`	Video jobs · transcripts · thumbnails · integrity
`/api/v1/drawings/...`	Page images · revision state · revision diff

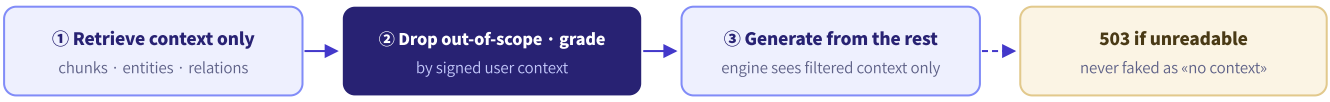
Citations: «no passage beats a wrong passage»

Each source gets one line of original text (max 200 chars) as a passage, but if it does not overlap the question enough, **no passage is attached**. Page numbers are given only when there is exactly one match. Chat counts the citation marks in the body in code and reconciles them with the source cards.

Four classification levels

General · internal · confidential · restricted. Restricted originals **cannot be downloaded** (403), and previews carry a watermark.

Order of the permission filter



Filtering «after» generation cannot remove content already mixed into the answer, so it is removed «before» generation. A denial has the same shape as «not found» (404), so even **the existence** of an out-of-scope document does not leak. If permission inputs cannot be read, the result is `ACCESS_UNAVAILABLE` (503).

Sources: plantpulse-rag-api MANUAL § 5 · § 5.1 (permission filter · grades · denial shape) · copilot MANUAL § 5.2b · ECOSYSTEM § 12.4-3 (citation check)

What it reads from elsewhere

Equipment doc list	Reads one platform ledger table directly — counted as an API migration target
Original download	Platform V5 REST
Asset lookup	Time-series insight lookup (declared same-layer call), else platform MCP
Permission check	Asset tree (60 s cache) + signed user context

Limits (configurable)

Query	3–4,000 chars · top_k 1–200
Video	≤ 2,048 MB · ≤ 120 min, processing window 20:00–06:00 by default
Removal guard	If too many items vanish in one run (ratio 0.25 · min 5), the delete step is skipped
Answer chips	At most 3 video scene chips

Sources: rag-api MANUAL § 2 · § 4 · § 5 · § 5.1 · § 6 ⑨ · ECOSYSTEM § 11.1 rule 2 · § 11.2 O7

10 · DOC UNDERSTANDING · VISION

Two reading modules — parsing by policy, reading by consensus

Both are **stateless**. Neither documents nor photos are kept after processing. Instead, these modules decide «how to read» and «when to say it could not be read».

Document understanding

The gate in front of the document conversion engine. It owns parsing policy as **four profiles** (default — same parsing as the RAG index · fast · OCR · scan), so RAG and vision read by the same rules. Video · audio transcription and drawing parsing · revision diff also happen here.

`/api/v1/understand`	Document → structured text (profile selectable)
`/api/v1/transcribe`	Submit · query · cancel transcripts OFF BY DEFAULT
`/api/v1/drawings/parse`	Drawing parsing (DXF · vector PDF)
`/api/v1/drawings/diff`	Difference between two revisions — dimensions · BOM · notes
MCP	`document_understand` · `document_profiles` (read-only)

Keeping transcripts from inventing

- Voice activity detection → cut at silence → recognition → **hallucination filter** (five reasons)
- Segments dropped by mistake are restored with re-aligned timestamps
- Segments whose language strays from the video's main language are recognized again
- Bulk and interactive lanes are split, so long videos never block questions

Hostile input faces size · decompression · time · concurrency limits, and URLs are blocked from reaching internal addresses. «Succeeded but empty» is not a success; it is `ok:false`.

Vision

Takes a photo, asks a vision-language model and returns a verdict. It asks the same question **several times** and tallies the votes; **if they split, it refuses** (`readable=false`).

`/api/v1/read_meter`	Gauge reading — 4 of 5 must agree (configurable)
`/api/v1/inspect`	Inspection — defect findings (max 8)
`/api/v1/ocr`	OCR — per-line consensus, partial results allowed
`/api/v1/classify` · `/detect`	Classify · detect
`/api/v1/analyze`	Free description — no values · verdicts
MCP	5 `vision_*` tools (detect is REST only)

«Could not read» is also an answer

Gauge reading first asks «is this a gauge?» three times and reads only after two «yes» votes. Ties are not consensus, and model output is validated against a schema. Refusal reasons are split into codes — model refusal · no consensus · no valid reading · model unreachable.

Measured (validation rig)	Result
Gauge reading baseline	4 correct · **2 refused · 0 wrong**
Nameplate fields	10 / 10

Source: plantpulse-vision-api MANUAL § 6.4.1 (measured 2026-10-04)

> **Refusing beats being wrong**
>
> If a gauge in a floor photo is misread «plausibly», that number ends up in a report. A refusal makes a person look once more.

Sources: plantpulse-document-understanding-api MANUAL § 1 · § 4 · § 5 · § 10.1 · § 10.3 · § 10.5 · plantpulse-vision-api MANUAL § 1 · § 4 · § 5

Where the reading modules connect to others

Input	Path	Where the result is used
Photo posted in chat	App → vision reading (gauge · nameplate)	Numbers in the answer · source card
Text in a photo	Vision OCR → delegated to doc understanding (declared same-layer call)	OCR result · partial consensus
Procedure video · audio	RAG → doc understanding transcript → segment index OFF BY DEFAULT	Scene time chips in answers
Two drawing revisions	RAG · app → doc understanding parsing · revision diff	Drawing card · revision badge · digital thread reference
Spoken floor question	App → doc understanding transcript → question	Ask (large buttons sized for gloved hands)

Sources: ECOSYSTEM § 11.1 rule 2 · § 12.2-1 · § 12.3-4 · § 12.4-5 · rag-api MANUAL § 4 · § 5

11 · ENGINEERING CALC · AUTOMATION SANDBOX

Arithmetic is not left to the model

A language model is useful for «choosing» a formula, but its decimal arithmetic cannot be trusted. So calculations are done by a **deterministic server**, and code that handles data directly runs only in an **isolated sandbox**.

Engineering calc

The model only chooses the formula and inputs; the server does the arithmetic and unit conversion. Results carry **value · substituted formula · basis**, so a person can redo the calculation. It uses no DB · LLM · GPU.

`eng_list_formulas`	Formula list — 30 formulas in 8 fields
`eng_formula`	Formula calculation (value · formula · basis)
`eng_unit_convert`	Unit conversion
`eng_calc_eval`	Allow-listed expression evaluation

Expressions only from an «allow list»

Only numbers · arithmetic · powers · parentheses · fixed functions · constants are accepted. Attribute access · subscripts · strings · comparisons · lambdas are refused, and expression length · node count · depth · evaluation time are capped (configurable).

Out of range: «warning»; beyond physics: «impossible»

Inputs outside the usual range are not rejected but get a warning; physically impossible values return `impossible`. The documentation states that unit mistakes «within» range (e.g. kN·m entered in an N·m field) are not caught.

Sources: plantpulse-engcalc-api MANUAL § 1 · § 2 · § 4 · § 5.1

Automation sandbox

Takes the Python compute steps of an approved, pinned automation plan, runs each in **fresh isolation per run**, and returns only output (JSON · tables) · figures · result codes. Records keep no «contents» of code · input · output. It is a REST module called only by the app, with no MCP surface.

`POST /api/v1/run`	Run — code · input · limits
`/api/v1/runs`	Run records (kept 30 days, configurable)
`/api/v1/runtime`	Shows the limits currently in force

Layered isolation

Container	Starts only on a user-space kernel (gVisor). **No fallback and no switch to disable it** — otherwise startup fails
Per run	New namespaces (user · network · process · mount), new temporary filesystem, zero capabilities, seccomp blocking exec · internet sockets · mounts
Startup self-test	Checks unprivileged uid · zero capabilities · loopback-only network · write-protected locations at startup and periodically — stops on any mismatch
Release gate	A suite of **27 escape attempts** runs in a real isolated container; release requires «0 failures»

Sources: plantpulse-automation-sandbox-api MANUAL § 1 · § 2 · § 4 · § 5 · plantpulse-ai-docker MANUAL § 5.11 · § 6.1 (`check-sandbox-escape`)

Engineering calc limits (configurable)

Expression length	1,000 chars
Nodes / depth	400 / 40
Evaluation time	0.5 s
Variables / function args	64 / 32
Significant digits	10

Sources: engcalc-api MANUAL § 4 · § 5.1

Sandbox run limits (configurable)

Time	default 10 s · 1–60 s
Memory	default 512 MB · 128–2,048 MB
Code / input	256 KiB / 16 MiB
Output	10,000 table rows · 8 figures (1 MiB each)
Concurrent / queue	2 / 16 — 30 s wait

Sources: automation-sandbox-api MANUAL § 4 · CONTRACT § 5 (`limits.py`)

Writes are not opened here either. The sandbox's only input is **data the automation obtained with read tools**, and with no network there is no way to write outward. Changing floor systems still goes through approval cards.

12 · VOICE · FILE STORAGE · LONG-TERM MEMORY

Three support modules — called by the app, each writing only to its own partition

These three are **parts of the app**, not model tools, so they expose REST only, no MCP. In return each owns things **the operator must answer for** — personal data · retention · erasure.

FILE STORAGE **ON BY DEFAULT**

Single owner of original bytes

Keeps originals of photos · video · documents · attachments with type detection and hash dedup. Separates the original bucket (versioning · object lock) from the derived bucket (thumbnails — expiry rules only here).

VOICE OUTPUT **OFF BY DEFAULT**

Screen text into «speakable text»

Turns numbers · units · times · codes into spoken words (Korean · English rules + a per-install pronunciation dictionary) and synthesizes sentence by sentence. Runs on CPU, with a sentence cache and a concurrency cap.

LONG-TERM MEMORY

Context beyond one conversation

Extracts facts from conversations and recalls them by semantic search in later ones. At least one scope (user · automation · run) is mandatory, and even admins cannot read others' memories.

API

`/api/v1/uploads`	Upload (resumable)
`/api/v1/objects`	Read (Range) · thumbnails · search
`/api/v1/admin/erase`	Personal-data erasure
`.../hold` · `/release`	Legal hold

Data · safeguards

- PG `pi.storage` + original · derived buckets. Backups take both at **the same point in time**
- Erasure turns every row of the same bytes into a tombstone and **blocks re-storing that hash**. If interrupted, it resumes and finishes
- Formats that can carry scripts (SVG · HTML) are never served inline
- Outside the user's asset scope → 404; audit records cannot be deleted

API

`/api/v1/speech`	Text → speech
`/api/v1/preview/normalize`	Preview speakable text
`/api/v1/settings`	Settings · pronunciation dict.

Data · safeguards

- PG `pi.voice` + a disposable sentence cache
- Cache keys include the normalization version · model name, so old audio never plays after rules change
- Weights are pinned by hash; the build stops if license text changes
- A full queue returns 429 — no silent slowdowns

API

`/memories`	Store · list · scoped delete
`/search`	Semantic search
`/memories/{id}/history`	Change history

Data · safeguards

- Own vector collection + change history. Unscoped list · search · delete return 400
- No outbound calls; telemetry is pinned off — turning it on makes startup fail
- Per-user deletion is offered via API

We state each module's maturity as it is. File storage ships in the compose by default, but its own documentation calls it an **MVP**; voice output is an **optional install** that is off by default (one line in the host settings turns it on). Neither has measured figures in its documentation yet, so none are given here.

Sources: plantpulse-storage-api MANUAL preamble · §1–§3 · §5 · §6 · §8 · plantpulse-voice-api MANUAL preamble · §1–§3 · §5 · §6 · §8 · plantpulse-mem0 MANUAL §1 · §3 · §5 · copilot MANUAL §5.33 · ai-docker compose (`profiles: [voice]`)

Where data about people remains — a map for operators

What	Where	How to erase · what is not kept
Chat photos · attachments	File storage (original bucket)	Erasure API — all identical bytes, re-store blocked
Conversations	App `pi.co`	Conversation delete (history text · messages · attachments included)
Memories	Long-term memory	Users edit · delete · pause on the «Memory» screen; per-user deletion API
Tool calls	App audit ledger · gateway log	Gateway keeps no arguments · results
Automation compute	Sandbox records	No code · input · output contents kept, 30-day retention

Sources: storage-api MANUAL §6 · copilot docs/BENCHMARKS «what the bench creates, the bench deletes» (conversation delete API) · mem0 MANUAL §5 · copilot MANUAL §5.33 (memory screen) · gateway MANUAL §6.4 · sandbox MANUAL §1 · §4

13 · INFERENCE STACK · ADMIN CONSOLE · EDGE PROXY

Three operational components — models · administration · entry

These three are not business functions but **the backbone of operations**. Models run on in-house GPUs, admins see the services on one screen, and there is a single way in from outside.

Inference stack

Serves three models on in-house GPUs — **language (chat and vision) · embedding · reranker** — plus **speech recognition** when video ingestion is on. The inference gateway in front wraps them in a standard chat-completions API and manages virtual keys · usage. The document conversion engine is in the same stack.

`/v1/chat/completions`	Chat completions
`/v1/embeddings` · `/rerank`	Embeddings · reranking
`/v1/models`	Only three served names exposed — no aliases

- Model server ports are not opened on the host. Apps go through the gateway
- A **GPU share budget** is computed before startup — if the sum exceeds the hardware cap, nothing starts
- Model archives are verified by hash, and are added **only under new names**, never overwriting an existing one
- Conversation bodies are not put in trace logs by default
- Health is not «connected»; it is healthy only when **the model list read with our key shows the served names**

Sources: plantpulse-ai-inference-server MANUAL § 1 · § 2 · § 5 · § 6.1a · § 6.2 · ai-docker MANUAL § 6.1 (healthcheck)

Admin console

Provides a service catalog, health rollup and identically shaped screens per service (MCP info · API · logs · settings · key state). The admin API behind it relays to each module at `/api/v1/{service}/...` (RAG · insight · doc understanding · vision · thread · gateway · sandbox · voice · files).

`/api/v1/health`	Catalog health rollup
`/api/v1/llm/*` · `/gpu/*`	Model · GPU metrics
`/api/v1/keys/probe`	Key gate check (401 without key · 200 with key)

- Health reads the **body**, not the status code, and calls once more with the console key — **a rejected key shows as degraded**
- Every write goes to the audit log, and the server fills in the admin identity
- The console **cannot call digital thread action tools**; sandbox runs · file writes are also 403
- Repeated login failures lock the account. Path tricks (dot segments · encoding) are refused

Sources: plantpulse-ai-admin-web MANUAL § 1 · § 5.1 · § 5.2 · § 5.4 · § 5.7–§ 5.14 · admin-api routers

Edge proxy

Terminates TLS and splits traffic **by port and path only** — admin console, inference API (allow list), MCP paths, and everything else to the decision app. It does no authentication. **Each backend checks keys itself**, so a proxy misconfiguration cannot become an authentication hole.

- Configuration is checked while the image is built (`nginx -t`)
- Plain HTTP gets 308 to HTTPS; security headers · trusted proxy list
- Streaming paths disable buffering; it runs as non-root

Sources: plantpulse-ai-proxy MANUAL § 1 · § 2 · § 4 · § 5.3 · § 7

14 · SAFEGUARDS ①

Rules written in a prompt are not kept unless code enforces them

This sentence is not a slogan but a development record. The same kind of incident happened **twelve times**, and stopped only after all twelve were blocked in code. So we moved «rules we ask the model to follow» into «code that checks at the output entrance».

Source: copilot MANUAL § 5.8 «Prompts»

Evidence guard — numbers come only from this turn's tool results

Case	What the guard does
Claims data while no tool succeeded	Asks once more; if still so, replaces the answer with «could not query, so cannot answer with figures»
Contains numbers absent from tool results	Forces one rewrite; any that remain get an «unverified number» warning line
Uses an earlier answer as evidence	Not accepted — the only evidence is the tool results of **this turn**
Misstates day counts · relations	Number guard (0.1-day tolerance on days) · relation guard (checks «A supplies B» claims against thread results)
Numbers in charts	The model writes only a chart placeholder; **the server builds the chart from tool results**

Sources: v1-chat-on-behalf § 9.2 (answer evidence guard) · copilot MANUAL § 5.18 (number · relation guards) · § 5.27 · ECOSYSTEM § 12.4-2

Guards at the output entrance — wherever model output comes in

State «no data»	So a fallback 0 is not read as a measurement and becomes a «crisis that isn't»
Do not invent	So the role of a non-existent measuring point never becomes diagnostic evidence
Required action fields	So no button ships that fails when pressed
Hide raw tool errors	So internal addresses · error HTML never appear in answers
Block prompt leaks	Cut mid-stream — not «after» the user has seen it
Strip reasoning traces	So reasoning markers never remain on screen

Guards sit at **one entrance**, not per caller — so they are not missed when a new caller appears. What is discarded is logged, and «must remove» and «must keep» cases are tested in pairs to also prevent **false positives that erase sound answers**.

Every module distinguishes «0» from «unknown»

Time-series insight	Emits different codes for «no data» and «failure», and puts «unusable, not empty» in MCP errors so the model never reports a failure as «no data». If the measurement is older than 7 days, «no measurement» is shown instead of accuracy
Forecast ledger	If actuals cannot be obtained, it is `unresolvable`, not filled with 0
Hypothesis test	A symptom that cannot be measured is `unmeasurable` — neither «supported» nor «refuted»
Digital thread	«Unknown» is not Go — it does not count as passed in readiness verdicts
Vision · doc understanding	No consensus is `readable=false`; an empty success is `ok:false`

Sources: timeseries-insight-api MANUAL § 1 · § 5 · § 6 · INTEL «forecast ledger» · «fault hypothesis test» · digital-thread-api MANUAL § 2 · § 6.14 · vision-api MANUAL § 1 · document-understanding-api MANUAL § 5

15 · SAFEGUARDS ②

Scope stays closed, writes go through approval cards, every call goes to the ledger

«What can be seen» and «what can be changed» are **server verdicts**, not buttons on a screen. Hiding a button is not protection.

Scope — if it cannot be decided, «not visible»

- Out-of-scope objects return 404 — denial and «not found» look the same, so **not even existence leaks**
- Results that cannot be decided are closed (`scope_unverifiable`). **An empty scope sees nothing**
- The app **signs** user context and passes it to backends (5-minute lifetime, configurable); backends verify the signature and filter by the same scope
- If the inbound key is empty, **every module refuses to start** — there is no no-auth switch
- If permission inputs cannot be read: 503 — never passed through «left open» (RAG · insight)
- Unknown settings mean startup failure or warning + default. Configuration errors are never swallowed by fallbacks

Sources: copilot MANUAL § 5.2b · rag-api MANUAL § 5.1 · timeseries-insight-api MANUAL § 4 · § 5 · gateway CONTRACT § 2 · mem0 MANUAL § 3 · ECOSYSTEM § 11.4-e · § 12.6 (principles)

Writes — from proposal card to human approval



The model has no path to execute directly · equipment control is out of scope · people confirm inside customer systems

Two-person approval RELEASED 2026.10	An action card in chat cannot be approved by the person in that conversation. High-impact actions (default: supplier expedite request drafts · work order drafts, set via `access.two_person.tools`) wait for a **second approval** after the first. Proposal cards work the same way. Shipped in public tag 2026.10 (10-19) (`TwoPersonRule`)
Default for external tools	Tools of external servers with no declaration are **treated as write tools**. The write-tool guard cannot be turned off
Automations	Write tools are never executed by automations — any action an automation proposes goes through human approval
Human-facing labels	«Accept» is only a record; pressing it changes no order · schedule · equipment

Sources: copilot MANUAL § 5.2b «two-person approval» (`TwoPersonRule` — public tag 2026.10) · § 5.7 · § 5.13 · § 5.20 · § 5.25 · § 5.30 · v1-chat-on-behalf § 0 · digital-thread-api MANUAL § 5 · § 6.4 · ECOSYSTEM § 11.4-k

Audit — what is kept and what is not

Where	Kept	Deliberately not kept
Decision app	Every tool call (audit ledger, no expiry) · setting changes · document views · approver · approval time	—
MCP gateway	Caller · server · tool · result code · time	Arguments · results · auth values
Digital thread	Connector action log · binding log	Payloads (values stay at the source)
Sandbox	Run audit · record rows	Code · input · output contents
Admin console · files	Writes · key views · deletions (undeletable audit records)	—

Sources: copilot MANUAL § 5.2 · § 5.2b · gateway MANUAL § 6.4 · digital-thread-api MANUAL § 5 · sandbox MANUAL § 1 · admin-web MANUAL § 5.7– § 5.14

16 · DEPLOYMENT

On-premises — models and data stay inside

The whole stack stands up on one customer machine from **a single compose file**. Models run on in-house GPUs and there are no outbound calls, so the same configuration runs on **air-gapped networks**. The Kubernetes chart is a port of that compose; the compose file is the source of truth.

Two install paths

Online	One-line install — OS detection → runtime · drivers · host tuning → registry check → settings file preparation. The platform connection is tested once before startup, and the install stops right there if it fails
Air-gapped	Bring in the signed **air-gap bundle** (images · installer · offline packages · verification tools), then verify → load → install. Models are brought in separately on media

The host is not changed carelessly

On a host already running containers outside our compose, the default is **«print the plan only»** — it shows what would change and writes nothing. Existing settings files are not regenerated, values are not overwritten, and kernel parameters are never lowered below their current values.

Values in one place

Per-install values (hardware shape · site values · platform connection · secrets) live only in one host settings file, and a pre-start checker judges every line. Secrets change only via a dedicated command, and platform accounts · passwords are not written in the settings file (they are obtained from the platform at startup).

Bundle verification — the trust anchor sits «outside the bundle»

1. Receive the bundle body and its sidecars (checksum · signature · public key)
2. Take the **public key fingerprint** from the documentation and compare — a key from the same place as the bundle cannot be the anchor
3. Verify signature · hash, and compare **the whole unpacked tree** with the signed file list
4. Images are re-hashed on the bytes being loaded. Files not in the verified bundle are not loaded, and their names are warned about
5. The signing tool shipped in the bundle is not vouched for by the bundle signature itself, so users are told to check it against the **upstream checksum** separately

> **Skipping verification must be explicit.** If the bundle is missing or verification fails, the import stops. To skip, a person must write the «no verification» switch by hand.

Image supply chain

Both registry images and bundle blobs are signed. External images are pinned to their digest at build time and pulled through an in-house cache, never directly from public image hubs. All our images run as non-root users.

Air-gapped import — what is in the bundle and what comes separately

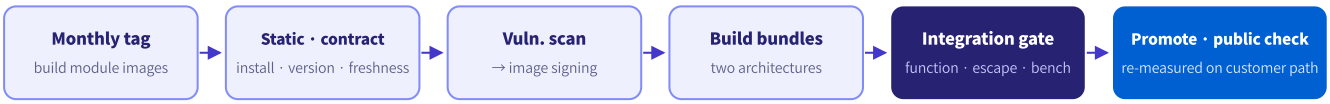
Category	Contents
In the bundle	All service images · installer · compose (i.e. image pins) · Docker offline packages · signature verification tools · sandbox runtime (version · hash pinned) · two source repositories · file-list checksums
Brought separately	Model weights (downloaded on the internet side, carried on media) — time-series models are inside the image. If video ingestion is on, the speech recognition model too
Before startup	If placeholders in the settings file are not filled, the start command refuses. The install checker verifies GPU shares · platform connection · runtime registration

Sources: plantpulse-ai-docker MANUAL § 1 · § 3 (online · air-gapped install) · § 4 (settings) · § 5.8 · ECOSYSTEM preamble (two signing points · non-root · proxy cache) · repos.tsv (Helm chart — compose is the source of truth)

17 · MONTHLY RELEASE · HARDWARE

Monthly auto-release — built, measured, signed and shipped with no one at the wheel

It is an unattended chain — at **02:00** each product module gets its monthly tag (`YYYY.MM`), and at **04:00** the deployment repository tags itself and carries on to publication. If any link in the chain «could not be measured», it counts as a failure — **we do not ship what we cannot measure.**



Integration gate	The only pre-release «function» gate. On the staging stack: ① auth · routing · MCP init · tool list ② **27 sandbox escape attempts** ③ the full chat bench in **strict mode** — a tool that is not healthy passes only if the «why» is written in a baseline file, and one that could not be called for lack of samples also fails. The bench takes about 130 minutes
«Failures are not masked»	No allow-failure flag is put on this gate. Masking a check that runs only right before release would show a «nobody measures this» state as green
Rollback	There is a procedure to roll a promotion back to the previous public version or a specific snapshot

Sources: plantpulse-ai-docker MANUAL § 6.1 (unattended monthly release chain — 02:00 service tagging → 04:00 self-tagging · `verify:integration`) · § 6.3 (rollback)

Hardware tiers

Tier	GPU	CPU	Memory	Disk	Status
Minimum	GB10 class (128 GB unified memory)	20 cores	128 GB unified	NVMe 1 TB+	**Measured shape**
Single card	NVIDIA RTX PRO 6000 Blackwell 96 GB × 1	32 cores+	64–128 GB	NVMe 2 TB	Profile exists — **not measured on that card**
Recommended	Same card × 2	32–64 cores	128 GB	NVMe 2 TB+	IN PREPARATION 2-card profile being built

≈ 77 GiB
GPU processes total — language · embedding · reranker · speech recognition · time-series server

113 / 119 GB
Host memory used (shared with the GPU)

30
Running containers

798 / 916 GB
Root disk used — models · two image versions · volumes

Source: plantpulse-ai-docker MANUAL § 3 «Hardware specification» — validation box (GB10, 128 GB unified memory) measured 2026-11-01. The single-card tier's GPU shares are the GB10 values carried over to 96 GB and were not measured on that card (same section). The system memory recommendation for card tiers (1 card 64–128 GB · 2 cards 128 GB) is based on the containers on the validation box using about **33 GiB** of memory in total.

GPU shares are counted before startup. The hardware profile sets the shares of the four model servers and the time-series server; if the sum exceeds the cap (GB10 0.75 · single card 0.95, configurable), nothing starts. The share of the speech recognition model, even when off, is included in the sum so that turning it on later cannot exceed the cap. For disk, **2 TB is recommended** because of models · two image versions during updates · backups.

Sources: ai-docker MANUAL § 3 · § 4 · § 5.10

18 · EVALUATION ①

People do not score — the same input gives the same score, whoever runs it

There is a set of questions that measures whether chat «answers properly», and **every verdict is machine-scored**. Model answers vary even for the same question, so no conclusion is drawn from a single score — at least 3 runs, some by the median of 5.

Bench	Items	What it measures	Axis
Tool call (KTB)	**25**	Does it call the right tool for the question — expected tool + arguments. Baseline **18 of 25**	**Gate**
Ontology (KOB)	16	For asset hierarchy · tag · relation questions, does it pick the right tool and avoid forbidden ones — median of 5	Record
what-if	8	For delay what-if questions, does it call the recovery tool and **carry its numbers into the answer** — checked against the calculator's numbers	Record
KTB-ext	116	Over «all» tools in the deployment, does it pick the right tool per category (measurement only)	Record
Answer accuracy	7	Does it carry tool-result numbers · names into the answer unchanged — checked against answers drawn from the tool at that moment	Record
Daily (groundedness)	8	Does it write only from the given context without inventing — «no data» marking · citations	Record
Floor tablet golden questions	17	Via the public chat v1 path — once per deployment, 3 times nightly. 0 failures on the defect axis	Record
Floor check · weight fingerprint · grader test	—	Is the deployed prompt the same as the repository · are dependencies alive · did the model change mid-run · **is the measurer right**	**Gate**

Sources: copilot docs/BENCHMARKS «run it all at once» table · ①–④ · MANUAL bench section (KTB-ext 116 items)

Why only KTB is a gate

Tool selection is an axis changed by **the app's prompts · tool descriptions · search**, so it can be fixed in code and can therefore block a release. Other model-quality axes vary even on the same question and cannot be fixed without changing the model, so shortfalls **are kept as alerts** but do not block promotion.

The questions are based on a demo plant

Questions are asked not about a real customer site but about a **heavy-industry demo plant** (gas turbines — 9 buildings · 16 lines · 53 assets (incl. 1 virtual power meter) · 801 tags) and objects in demo external systems. Moving values (alarm times, etc.) are scored by **tool · arguments**, not by the answer.

Sources: copilot docs/BENCHMARKS «run it all at once» (axis assignment — decided 2026-10-18) · MANUAL «basis of chat tests — heavy-industry demo»

What the bench creates, the bench deletes

The bench creates a new conversation per question, and finished answers also accumulate in long-term memory. Left alone, **bench memories would be recalled into later answers** and contaminate the measurement. So created conversations · memories are logged and deleted at the end, and deletion is **measured again**. If the memory store does not respond, the result is recorded as «not measured», not «0».

Source: copilot docs/BENCHMARKS «what the bench creates, the bench deletes»

Replays use the same functions as production. Evaluation · replay call production code as is; there is no separate copy implementation for replay. Results carry the versions of scorer · model · rules, and gates are registered in a file **before seeing results** — gates are never adjusted after seeing them.

Sources: ECOSYSTEM § 11.4-f · § 11.4-j · § 11.3 (autonomous gates — pre-registration)

19 · EVALUATION ②

What we measured — and what those numbers «do not mean»

Below are values each module's documentation records as **measured**. Conditions (items · date · data) are given alongside, and caveats the documents attach themselves are carried over as is.

Tool count and prompt length — smaller wins

Condition	Tool-call score
System prompt 5,488 chars	11 / 22
System prompt 1,189 chars	**18.5 / 22**

Long system prompts and many tools lower a small model's tool calling. So even as connectors grow, **the number of tools chat sees is fixed** (a few thread tools), and KTB · KOB are re-measured whenever a connector is enabled. Tool selection is the job of the gate's search (Ch. 6 — Recall@20 of 0.972 over 76 tools).

Sources: ECOSYSTEM § 11.4-m (measured chat tool count · prompt length) · gateway MANUAL § 5.2 (2026-10-18)

Analytics modules

What	Result	Conditions · caveats
Work order completion forecast	Mean absolute error (vs. target) on orders finished as planned: **3.9%** vs 6.9% for current-rate extrapolation (480 min, 12 orders) · **3.6%** vs 7.2% (20 min, 39 orders)	104 orders sampled from a simulated site (2026-09-27). Document caveats: **the same sample** used to choose the design; does not beat extrapolation on orders closed early
Fault type diagnosis (synthetic)	Top-1 hit 0.844 · flag precision 0.984 · false flags 0.003	Synthetic evaluation — the document states it is **an upper bound, not field accuracy**
Gauge reading	4 correct · 2 refused · **0 wrong**	2026-10-04 baseline. Wrong readings did occur in 12 repeats on borderline samples (document § 6.4.1)
Nameplate field reading	10 / 10	Same section. 7 / 10 with the document conversion engine alone

Sources: timeseries-insight-api docs/ORDER-COMPLETION-EVAL «results» · «base model» · «limitations» · INTEL «synthetic evaluation» · vision-api MANUAL § 6.4.1

Only anomaly scorers that pass the gate reach the floor. Time-series anomaly scores are measured on public industrial data and simulation data against a pre-registered gate (precision · false alarms · negative controls), and scorers that do not pass stay in **shadow**. The accuracy screen shows «no measurement» instead of a value when there is no result or it is older than 7 days. Re-measurement (recording reconstruction-error-only AUC) is on the roadmap.

Sources: timeseries-insight-api MANUAL § 6 · docs/ANOMALY-EVAL · ECOSYSTEM § 12.1-5

What goes into the next edition

What	Status
Public chat v1 · on-behalf questions	Implemented · on staging. SHIPS IN NEXT RE-RELEASE Voice question · voice answer not yet on staging
Console surface · approval cards (actions)	Ship together in **the same next re-release** as public chat v1 PLANNED
Platform data API migration	Shadow comparison per table under way. Bulk sensor time-series waits for the platform bulk API spec
Permissions — roles · scope · grades	Phases A · B (2026.11) · C (2026.12) · D (2027 Q1 — external auth · temporary rights) PLANNED. Two-person approval is separate from these phases and **already released** (2026.10, Ch. 15)
Field validation set	From customer-site alarms · maintenance records · gauge photos · procedure videos — waiting for a site
Procedure bench · external reference bench	A bench that goes through several tools to a verdict, and an adapter connecting public industrial bench scenarios to our tools — pending
Commercial connectors, live	Mock-server conformance complete — real product connections at pilot sites

Sources: v1-chat-on-behalf preamble · docs/platform-data-via-api preamble · ECOSYSTEM § 12.1-2 · 4 · 8 · 9 · § 12.6 · plantpulse-dt-connectors FEATURES

PlantPulse® **AI Platform**

TECHNICAL WHITEPAPER

Each module does one thing, and people make the decisions.

Every claim in this paper comes from the documentation and code in the module repositories. If you want to look deeper into a module during pre-installation review, we will provide its API specification together with its contract tests. You can also run the same benches yourself on the demo plant.

STRUCTURE

One owner per module

One repository · one image · one contract. Other modules' data is read through the APIs their owners publish.

GATE

Tools through one gateway

Permissions · scope · search · logging are decided in one place, and external tools arrive switched off.

ACTION

Writes via approval cards only

Drafts · requests only in customer systems, two-person approval for high-impact actions, equipment control out of scope.

EVIDENCE

Numbers from tool results

Evidence guards check at the output entrance, and «unknown» is never reported as «0».

DEPLOYMENT

In-house · air-gapped · signed

Models and data stay on customer machines; bundles are verified by signature · hash · file list.

EVALUATION

Machine scoring · release gate

The monthly auto-release ships only after passing the tool-call bench and the escape tests.

DOCS

docs.plantpulse.io

/ai — AI Platform manual

CONTACT · DEMO

kopens.com/contact

1588-5268 · webmaster@kopens.com

PRODUCTS

plantpulse.io

Platform · Edge · AI · Studio

Rev. 2026.11 · Draft

KOPENS, Inc. © 2026 KOPENS. PlantPulse® is a registered trademark of KOPENS, Inc.

kopens®