TECHNICAL PORTFOLIO

Automation and artificial intelligence. Cases in production.

INDEX OF CASES

01 **Unloading and silo management in an industrial plant** AGRI · LOGISTICS

Drivers and plant staff talk to the AI agent on Telegram; the engine recalculates the sequence of around 80 trucks a week and sends the alerts itself.

02 **WhatsApp appointment agent on the clinic's CRM** HEALTHCARE

Serves patients 24/7 on the real diary inside their CRM (Nubimed): CRUD operations on appointments, reminders and handover to a human when needed.

03 **Tenant lead capture and qualification over WhatsApp** REAL ESTATE

Qualifies candidates and their financial standing without phone calls, and automatically matches every candidate against the agency's property catalogue.

04 **Management platform for property managers** REAL ESTATE

The full incident cycle —report, classify, notify the trade and follow up— is handled by AI agents over WhatsApp.

05 **Email orders → ERP (Odoo)** DISTRIBUTION

Reads free-format orders written in natural language, matches them against the real catalogue and leaves them created in the ERP, ready to approve with one click.

06 **Food traceability automation** MEAT INDUSTRY

From supplier documents in wildly different formats to data ready for the ERP, batch by batch, exactly as the regulation demands.

07 **RAG assistant over internal documentation** INDUSTRY

Answers staff citing only internal documentation as its source; if the documentation doesn't cover it, it says so.

08 **Time Parser — dates in natural language** INFRASTRUCTURE

In-house microservice that turns "tomorrow at 7" into exact, always consistent dates, for every conversational agent.

Every project described here is my own work: process analysis, architecture, development, deployment and maintenance.

Unloading and silo management in an industrial plant

Planning, simulation and coordination of the weekly raw-material truck unloads — with an AI agent on Telegram that talks directly to the drivers.

What it is

Web application to plan, manage and control the weekly unloading of trucks into the silos of an industrial plant. It replaces the client's previous system —an Excel sheet with macros where one person noted around 80 trips a week by hand and recalculated everything with every driver's phone call— with a digital environment accessible from any browser.

Who it’s for

Logistics managers who need to know which trucks are scheduled, at what time and with how many tonnes. **Plant managers** who want to get ahead of production stoppages caused by an empty silo. **Drivers**, who confirm trips or report issues from their phone, with no phone calls.

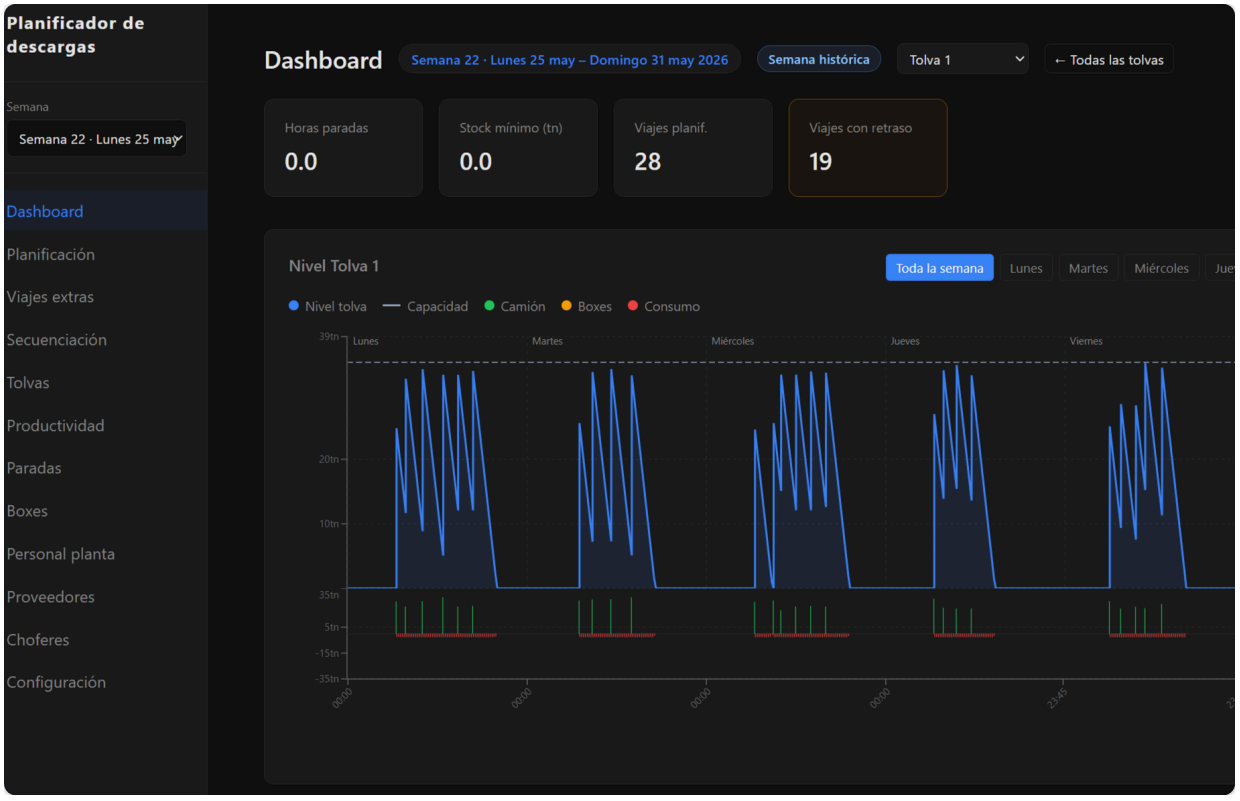
Main features

- **Weekly dashboard:** hopper capacity and level, estimated stoppage hours, forecast minimum stock, planned trips and delays, with an hour-by-hour level chart. Configurable consumption and manual entry of the real level when needed.
- **Import from Excel/CSV:** the client keeps planning exactly as before; the system imports the whole week, allows every trip to be edited and keeps the history in read-only mode.
- **Simulation and sequencing engine:** simulates the operating week (configurable schedules for unloading and consumption): truck arrivals against production consumption, resulting hopper level, finish time of each unload and delays caused by saturation.
- **Priority for critical trips:** look-ahead logic — if an upcoming critical trip won't fit in the silo, the system holds back the non-critical ones to make room for it.
- **Real plant conditions:** it tolerates variability when adherence to the plan isn't exact: unplanned extra trips, variable consumption bands, production stoppages and gradual intake through boxes.
- **Telegram bot with AI:** the driver is shown their scheduled trip with buttons — confirm, different weight, changed time or other issues — and it interprets free-text replies. Drivers register themselves the first time they write.
- **Chain recalculation and alerts:** every reply updates the application, triggers the recalculation and automatically notifies the affected drivers if new delays appear.
- **Integration with other systems:** configurable webhooks (recalculation finished, delay detected, trip added...) to connect with an ERP, Power BI or email, plus a documented API.
- **Simulation mode:** lets changes be tested before approving them, and holds back "minor" operations that would only be noise, without altering the truck sequence.

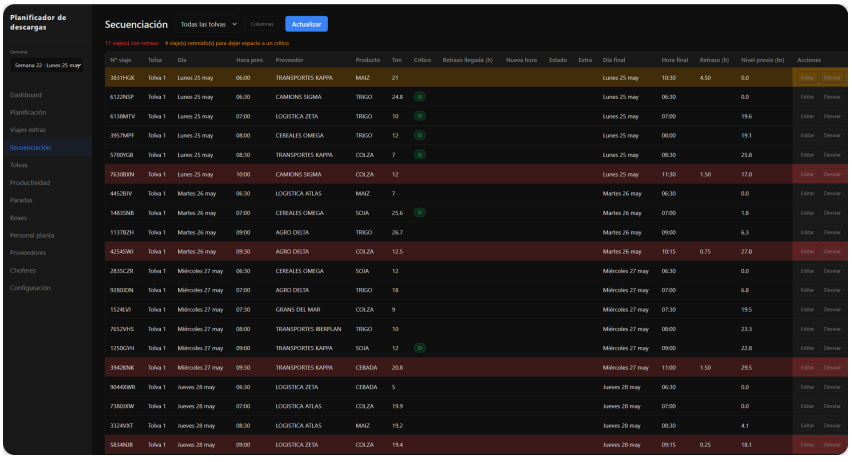
STACK

`React` `Node.js / Express` `PostgreSQL` `n8n` `Telegram Bot API` `LLM API` `Docker` `Own VPS`

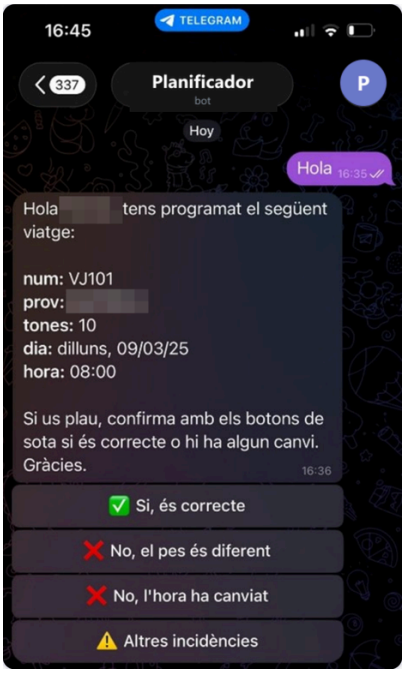
Screenshots



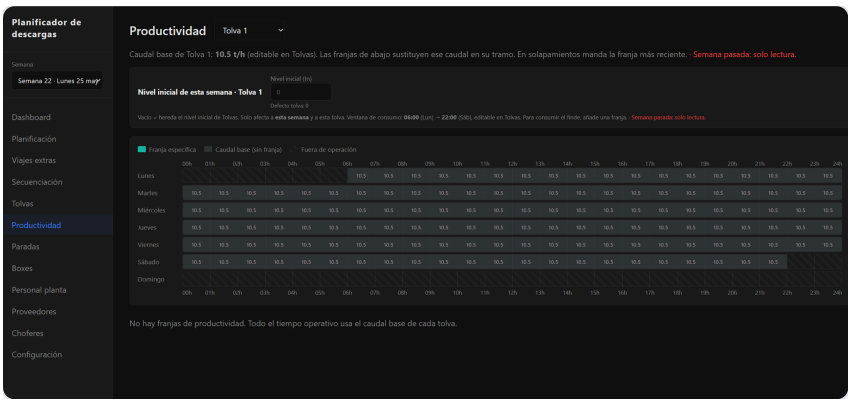
Weekly dashboard: simulated hopper level hour by hour against capacity, with KPIs for planned trips and delays.



Sequencing: critical trips prioritised, calculated delays and holds to make room for a critical one.



Telegram bot: trip confirmation with buttons or natural language.



Productivity: base flow rate per hopper and configurable consumption time bands.

WhatsApp appointment agent integrated with the clinic's CRM

AI on top of the CRM the clinic already uses (Nubimed): the diary and the patients stay in their usual system — the bot simply makes part of the phone calls disappear.

What it is

Conversational WhatsApp assistant that handles the full appointment cycle of a dental clinic —booking, checking, changing and cancelling— on top of the CRM the clinic already uses (Nubimed), without replacing it: availability always comes from its official API, in real time. The patient writes in natural language ("can I come next Thursday afternoon?") and an AI agent with tools (tool-calling) resolves the conversation: it identifies the patient, registers new ones and always works on real gaps in the diary. Reception works exactly as before — part of the calls simply disappear.

Who it's for

Clinics and health centres working with Nubimed that lose hours of reception time on the phone; each clinic gets its own isolated deployment.

Main features

- **A real conversation, not a menu:** it captures several pieces of information in a single message ("first visit with Dr. Roure, Friday at 16.00, I'm Joan Torres") without asking again for what has already been said, and maps symptoms to services without inventing treatments.
- **As flexible as a receptionist:** it allows switching professional or service mid-conversation and resolves relative dates ("Tuesday in two weeks") without calendar errors.
- **Clinic business rules:** continuity of care — a returning patient is offered their usual practitioner (preferred or mandatory mode, configurable) — and fair distribution when the patient says "anyone".
- **No lead lost:** appointments for third parties ("it's for my son") with their own record; if no slot fits, it logs a request for reception to call back.
- **Operational safety:** explicit confirmation before booking, cancelling or changing; if the patient backs out, nothing is touched; if a change finds no slot, the original appointment stays intact.
- **Kill-switch and handover to a human:** the bot goes quiet but keeps storing context, and when reactivated it picks up where it left off, taking into account what the human discussed through the CRM.

Version for businesses without a CRM: scheduler on Google Calendar

The same concept for small businesses with no CRM (hair salons, private practices, services): the same assistant on Google Calendar, one calendar per professional. It calculates gaps respecting opening hours and closures, sends automatic reminders, and synchronisation is two-way: if the owner moves an appointment by hand, the system detects it.

STACK

`CRM version: Node.js / TypeScript` `tool-calling agent with Claude (Anthropic API)` `Chatwoot` `WhatsApp Cloud (Meta)` `PostgreSQL` `Docker` `Own VPS` `GCal version: n8n` `OpenAI agents` `Redis`

Automatic tenant capture and qualification over WhatsApp

The prospect writes on WhatsApp, the AI qualifies them and collects their full record, and the agency is alerted when a matching candidate appears — with automatic catalogue matching.

What it is

Complete system for a rental estate agency that automates the candidate funnel end to end: the prospect writes on WhatsApp, an AI assistant serves them and collects their details in structured form, the agency receives alerts when a candidate meets its requirements, and the same system matches every candidate against the properties in the catalogue. It replaces the previous process —repeating the same questions on the phone and writing them down on a paper form— by digitising exactly the rental form the agency was already using.

Who it's for

Estate agencies and rental management firms that get a lot of enquiries and need to filter financial standing without spending hours on the phone. Deployed for a real agency, serving customers in Catalan, Spanish and English.

Main features

- **Multi-agent WhatsApp assistant:** specialised conversational agents that hand over to each other — one greets and receives, another qualifies (rental or purchase, and which specific property: address, reference or Idealista/Fotocasa listing) and the flow branches: if it's a purchase, a human agent is alerted.
- **Guided data capture inside the chat:** for rentals with an identified property, a third agent collects the record step by step, answers questions along the way ("what does employed status mean?") and normalises ambiguous answers. Prompts and questions are editable from the settings, with no code changes.
- **Public form in three languages:** it collects contact details, housing preferences (type, maximum price, rooms, area, parking, furnished, pets...) and the financial profile: employment situation, contract type, seniority and income bands. When finished, it returns the user to the WhatsApp conversation.
- **Admin panel:** capture metrics (total leads, today, this week, by area and employment situation), candidate management with filters and CSV export, and a property catalogue with bulk import from Excel/CSV.
- **Configurable alerts:** the agency defines its requirements ("permanent contract and income above €2,000") from the panel; when a candidate meets them, it gets an immediate alert by WhatsApp and email, with a historical log.
- **AI candidate–property matching:** two layers — a classic filter (price, area, rooms, pets) and a ranking by semantic similarity between the candidate's preferences and the property descriptions, with a configurable threshold. It runs when a property is created or edited and in a nightly process; every match is logged and the agency notified.

STACK

`React + Tailwind` `Node.js` `PostgreSQL` `OpenAI agents over WhatsApp` `Chatwoot CRM` `Google Gemini + Qdrant` `n8n` `Docker` `Own VPS`

Management platform for property managers

Centralises buildings, residents, suppliers and documentation — and the full incident cycle (report, classify, notify and follow up) is handled by AI agents over WhatsApp.

What it is

Web application that centralises the day-to-day work of a property management office: buildings, properties, residents, suppliers, incidents and documentation. The conversational layer is the heart of the system: when a resident reports a fault on WhatsApp, an AI agent logs and classifies it, the system evaluates the alert rules and another agent contacts the supplier from the relevant trade and follows it up — with no manual intervention.

Who it’s for

Property managers running several buildings who receive incidents at all hours through scattered channels. **Offices with several managers**, thanks to a super-admin role that handles each manager's account and access.

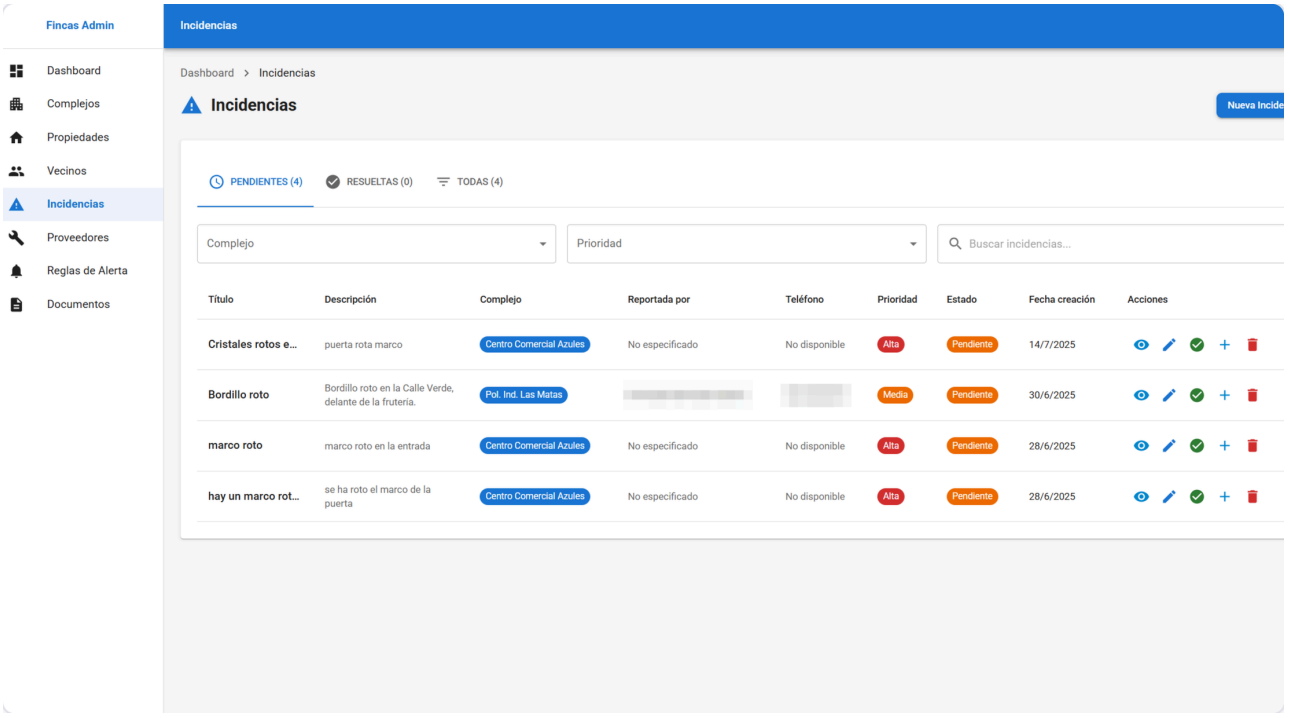
Main features

- **WhatsApp agent for residents:** the resident writes in natural language ("the entrance door is broken"); the agent asks for the missing details, identifies the building, proposes an urgency level, confirms it with the resident and logs the incident.
- **Answers from each building's documentation:** the same agent answers residents' frequent questions based on the uploaded documentation (rules, opening hours, service contracts...).
- **Conversation with suppliers and follow-up:** an AI agent contacts the supplier from the relevant trade on WhatsApp ("this building's plumber"), passes on the details and photos of the fault, and follows it up until resolution, keeping the status up to date on the platform.
- **Incident management and classification:** every incident has an urgency level, the resident who reported it, the building, internal follow-up notes and a gallery of images of the fault. The dashboard shows global statistics and the most recent ones at a glance.
- **Complete business structure:** a Complexes → Properties → Residents hierarchy: developments or buildings, homes and commercial units, and residents linked to each property. Every level with its own record, creation and editing.
- **Configurable alert rules:** the manager defines, with no code changes, when and whom to notify: by urgency level and keywords, with recipients, channel and expected response time.
- **Real-time notification:** every change to an incident evaluates the rules and fires the alert instantly, with no duplicate alerts to the same recipient.
- **Document management on Google Drive:** each building's documents are stored through a service account — the same documentation that feeds the agent's answers — with automatic alerts when obsolete documents are deleted.

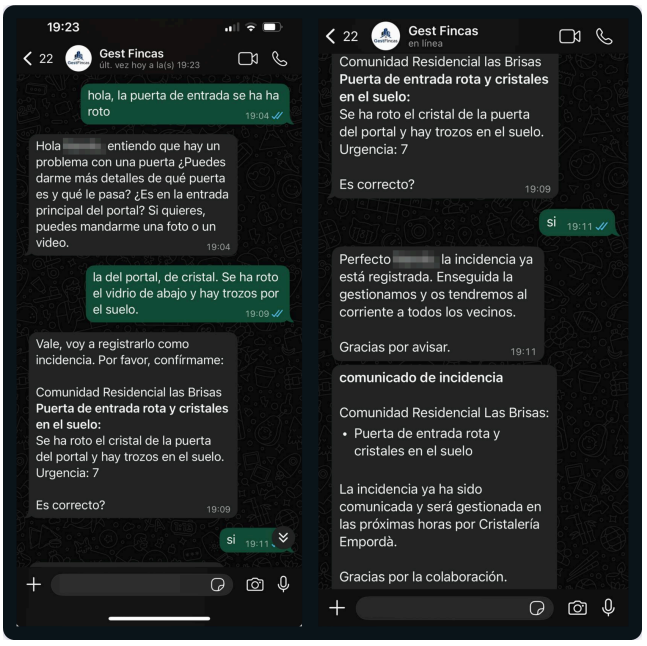
STACK

`React + Material UI` `Node.js` `PostgreSQL (triggers + LISTEN/NOTIFY)` `LLM API` `WhatsApp` `n8n` `Google Drive API` `Docker` `Own VPS`

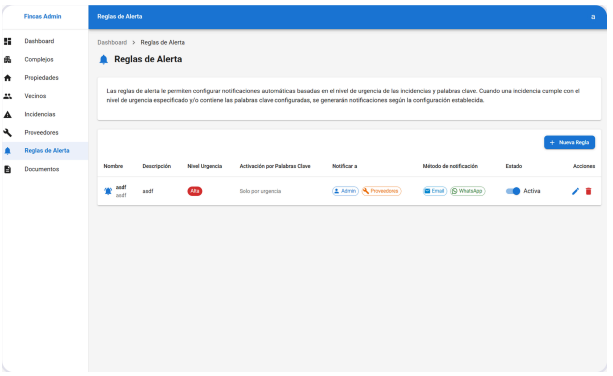
Screenshots



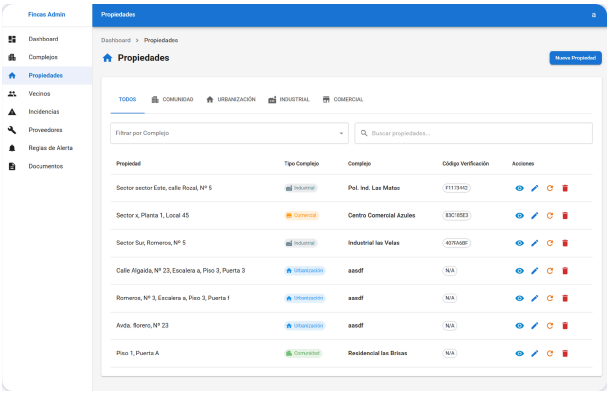
Incident management: priority, status, reporting resident and follow-up actions.



A resident reports a fault on WhatsApp: the AI agent logs it as an incident, confirms it and communicates the assigned trade.



Alert rules: urgency, keywords, recipients and notification channel.



Property structure: residential, development, industrial and commercial.

Email orders → ERP (Odoo), interpreted with AI

Customers keep sending orders by email as always; the system understands them, structures them and leaves them created in Odoo, ready to approve with one click.

What it is

Automation pipeline for a distributor that receives orders by email in free format and natural language: every customer writes their own way, with their own product codes and no template. The system reads each email, extracts the order data with AI, matches it against the real catalogue and creates the sales order in Odoo — removing manual data entry and the errors that come with it.

How it works

- **Mailbox monitoring:** the order inbox is watched; every incoming email triggers the flow.
- **AI interpretation:** an LLM extracts the structured data from the free text — customer, products, quantities and notes — in a validatable format.
- **Product matching:** the customer's codes and descriptions are matched against the distributor's database; if the customer uses their own codes, they are resolved against the history of equivalences. The AI picks the most likely candidate when a description is ambiguous.
- **Creation in Odoo:** a sales order with its lines through Odoo's external API, created as a draft: the sales rep only reviews and confirms.
- **Exception handling:** if a code isn't recognised, the system replies to the customer reporting the issue and notifies the internal owner. Nothing is lost in silence.

STACK

`n8n` `LLM API (structured JSON output)` `Odoo (JSON-RPC)` `PostgreSQL` `Gmail / IMAP` `Docker` `Own VPS`

Food traceability automation

From delivery notes and supplier documents in wildly different formats to structured traceability data, ready for the ERP.

What it is

Automated flow for a fresh-produce company that has to maintain the traceability of its raw meat (Regulation EC 178/2002). Previously, one person manually read the documentation for each intake — product descriptions in Catalan, different formats depending on the supplier— and transcribed it to import into the ERP line by line. The system performs that reading, interpretation and structuring automatically.

How it works

- **Document capture:** intake of the traceability documentation for raw-material deliveries (delivery notes, labels, supplier documents) in heterogeneous formats.
- **Extraction and interpretation:** a combination of text extraction and an LLM that understands product descriptions exactly as they are written —in Catalan, with each supplier's abbreviations and variations— and normalises them.
- **Mapping against the master data:** each description is mapped against the references in the client's system, keeping batch-by-batch correspondence between what is received and what is recorded.
- **Output ready to import:** the structured data is delivered in the format the client's ERP expects, removing manual transcription and its errors in a piece of data where a mistake isn't an inconvenience: it's a compliance problem.

STACK

`n8n` `LLM API` `document extraction / OCR` `client's ERP` `Docker` `Own VPS`

RAG assistant over internal documentation

"Like ChatGPT, but with your company's documents": a chat that answers citing work instructions and technical datasheets as its only source — without making things up.

What it is

Conversational assistant that answers staff questions using exclusively the company's internal documentation —work instructions, technical datasheets, procedures— as its source of truth. Instead of digging through folders or asking whoever has been there longest, the employee asks in natural language and gets the answer with a reference to the document it came from.

How it works

- **Documentation ingestion:** the company's documents are chunked and indexed so that the system can retrieve the exact fragment, always keeping the reference to the source document and section.
- **Retrieval + generation (RAG):** for each question, the system retrieves the most relevant fragments and the model writes the answer relying only on them. If the documentation doesn't cover the question, it says so — it doesn't fill the gap with invention.
- **Answers with a source:** every answer states which document it came from, so the employee can verify it and the company keeps the work instruction as the authority, not the chat.

STACK

n8n | Qdrant (vector database) | embeddings + LLM API | PostgreSQL / pgvector | Docker | Own VPS

Time Parser — deterministic interpretation of dates in natural language

"Tomorrow", "Friday at 7", "next week" turned into exact and always consistent dates.

What it is

Internal microservice that turns time expressions written the way people speak —"tomorrow", "Friday at 7", "the day after tomorrow"— into exact, structured and always consistent dates and times (in a standard format, with a time zone). It was born out of a real problem: AI models and generic parsers fail precisely on dates —they confuse "Friday" with "next Friday", return days that have already passed or impossible dates— and that is the kind of error a business cannot afford. This service solves that one piece once, properly, and makes it available to every other project.

Who it's for

It is internal infrastructure: a central service consumed by the automation flows and conversational agents of the different projects, with a single documented input/output contract.

Main features

- **Simple, stable API:** a single endpoint receives the user's phrase, a reference date ("today") and the time zone, and returns the resolved date, the day of the week, the time and two flags that matter to the business: whether it falls on a weekend and whether it is in the past.
- **Bilingual Spanish and Catalan:** it detects the language and understands days, months and relative expressions in both ("Friday", "next week", "the day after tomorrow").
- **Deterministic by design:** the same phrase always produces the same result. Explicit rules for ambiguous cases: it distinguishes "Friday" (this week) from "next Friday"; "next week" with no day resolves to Monday; "at 7" in an afternoon context is read as 19.00.
- **Protection against calendar errors:** it never returns a date in the past (it jumps to the next valid day) and it detects contradictions ("Thursday the 18th" when the 18th is not a Thursday), rejecting them with an explicit error instead of guessing.
- **Configurable excluded days:** it accepts a list of the business's closing days and automatically jumps to the next valid day.
- **AI as support, not as the base:** the main analysis is deterministic and works without depending on any model; only when the classic parser does not understand the phrase does an AI model step in as optional support. Minimal cost and predictable behaviour.
- **Traceable:** every response includes a step-by-step account of how the date was resolved (key for debugging conversations), with health monitoring and access-key protection.
- **Combined with the URTA agent (last active time reference):** so you can move forward or back in the conversation without mentioning the day again: the last specified reference is kept in memory, beyond the immediate conversation context.

STACK

`Node.js` `chrono-node` `Luxon` `Docker` `Own VPS`