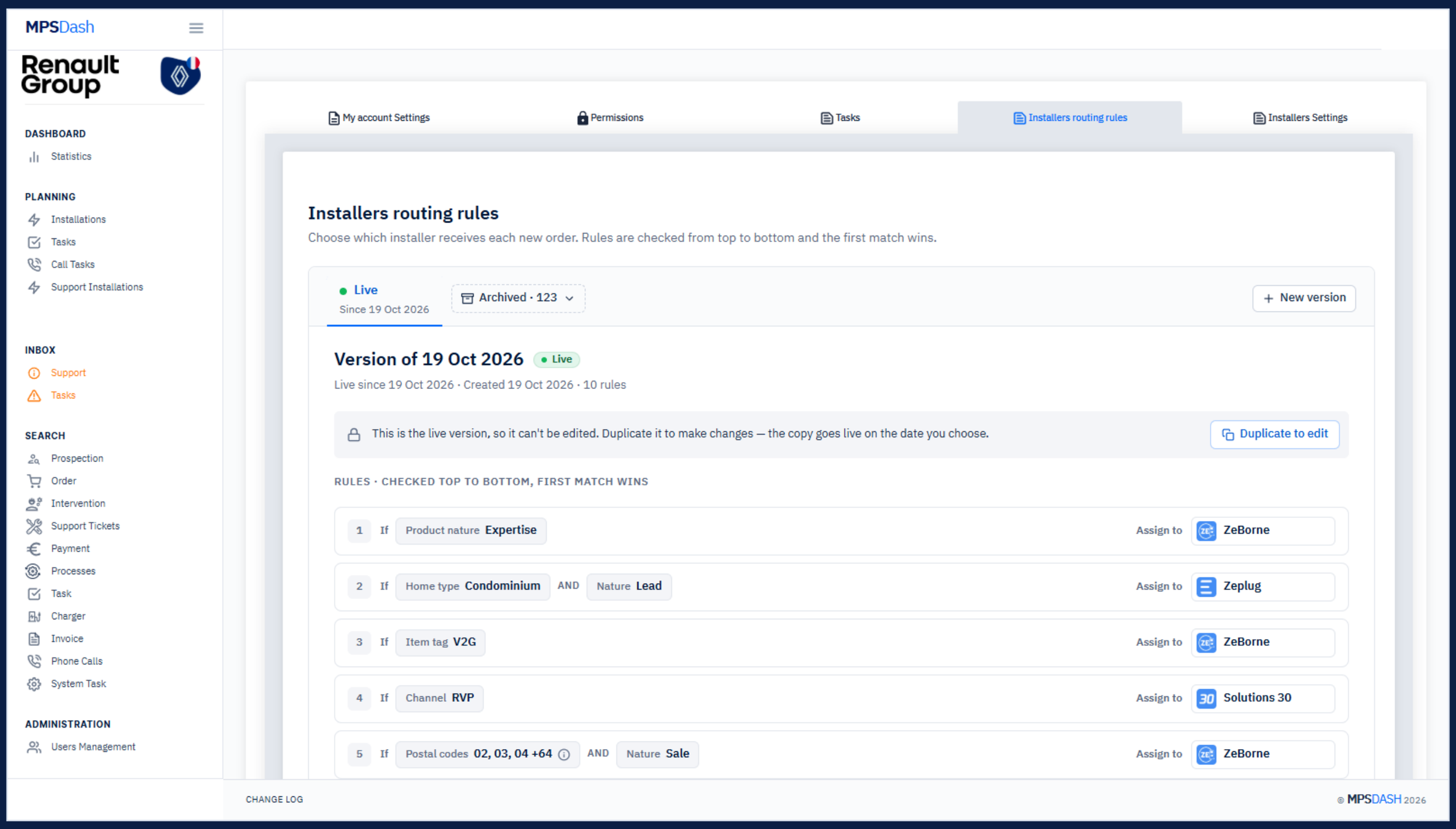
ELTO platform

The back office behind home EV charging.

UI/UX design and front-end engineering for a six-country operations platform.

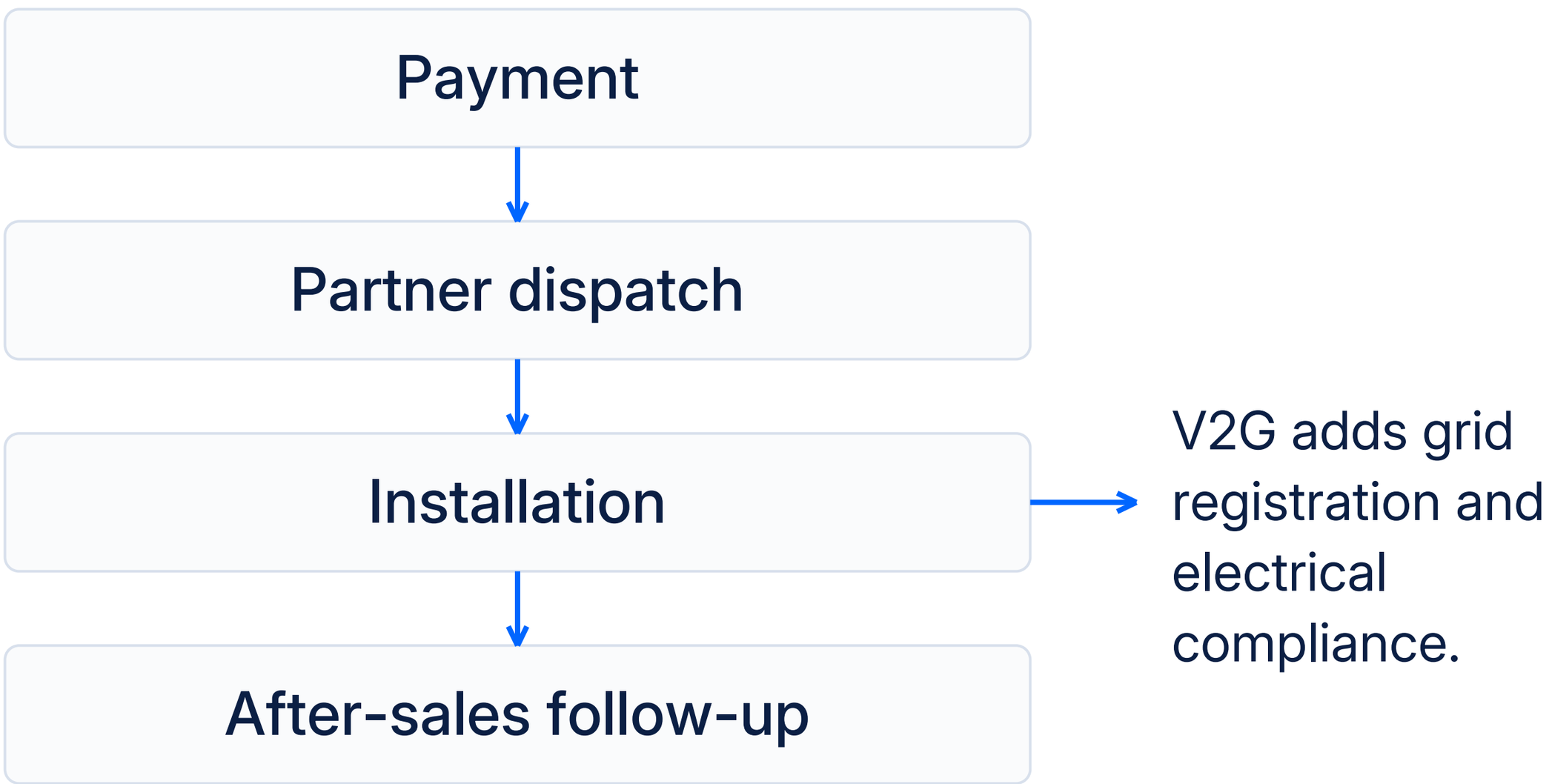


Taha Znazen

AI Product Engineer

One sale becomes an operational journey.

ELTO teams coordinate charger sales and installations for Renault and Mobilize customers. Each order crosses payments, partners and after-sales work.

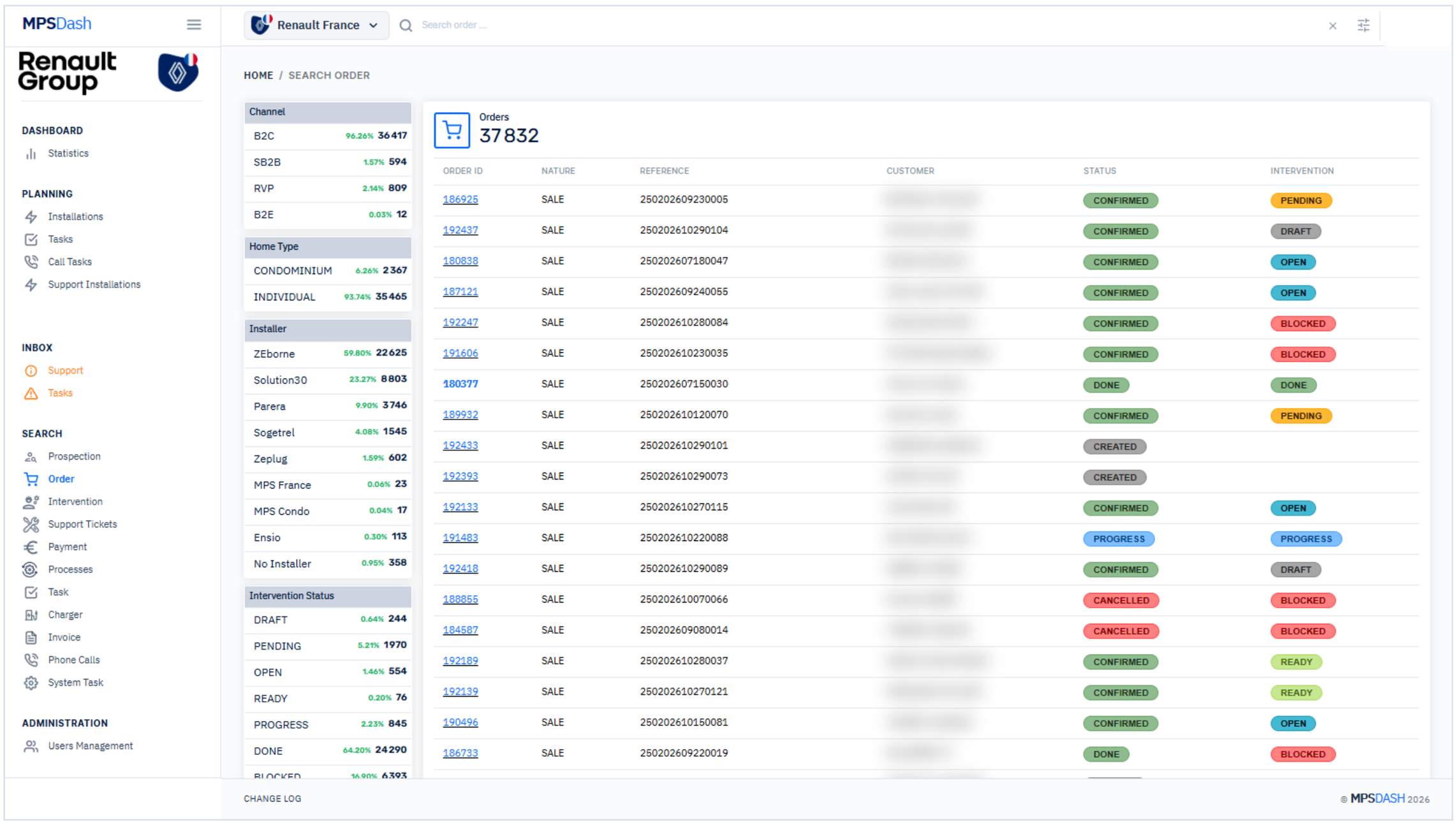


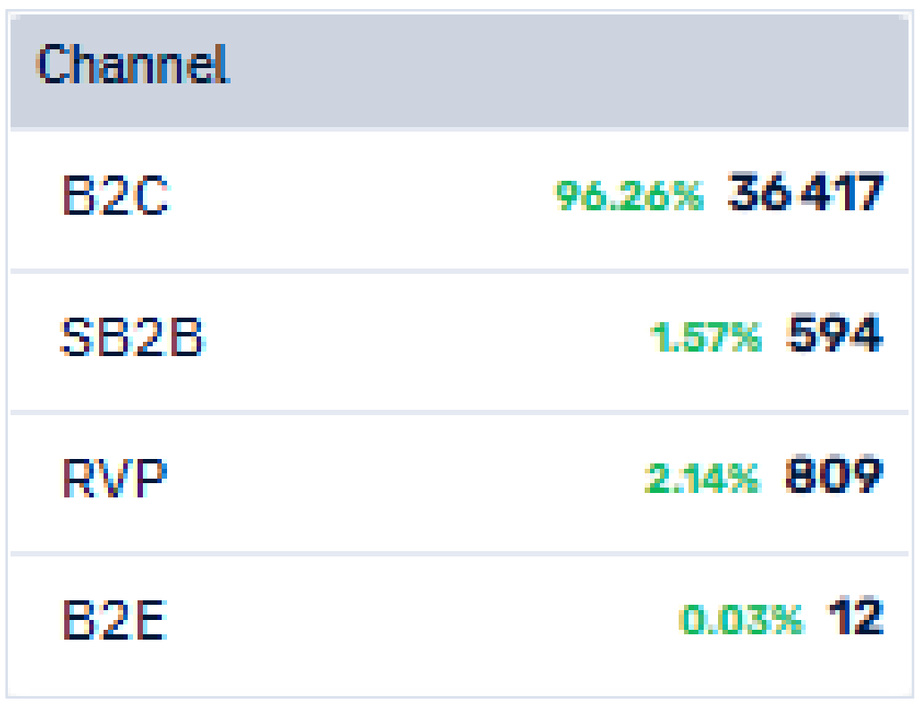
I designed all 43 screens and built front-end features with a teammate. A separate team built the backend.
Entity Services, 2023-present.

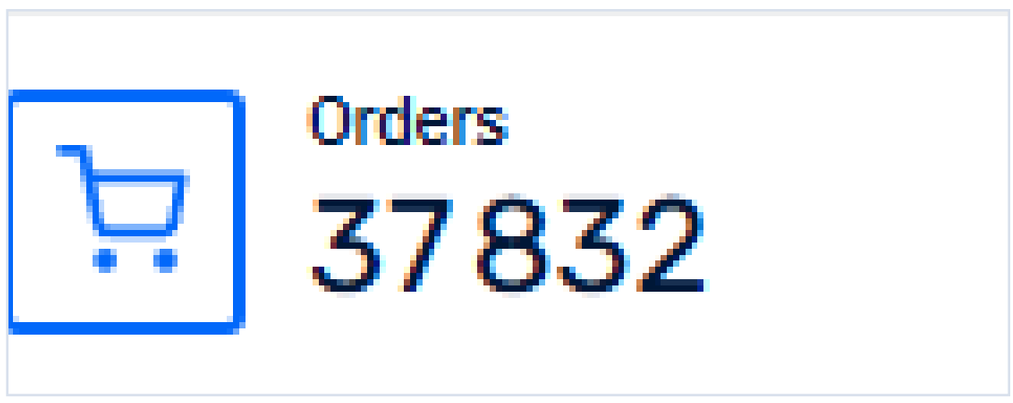
France, Germany, Spain, Italy, Belgium and the Netherlands.

One search model.
Many kinds of work.

Counts, filters and results stay together. Advanced filters add precision; saved queries preserve useful searches. CSV export follows the user’s permissions.



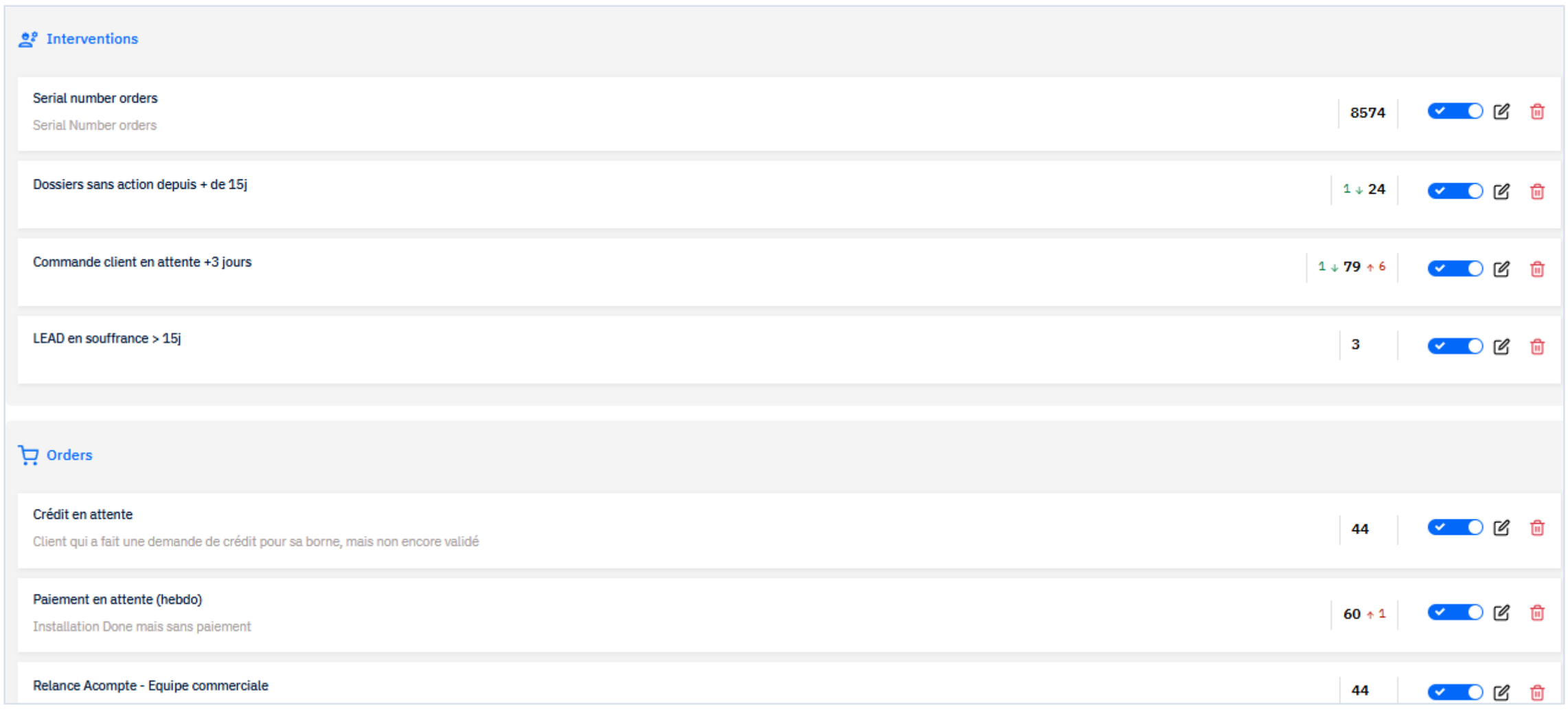


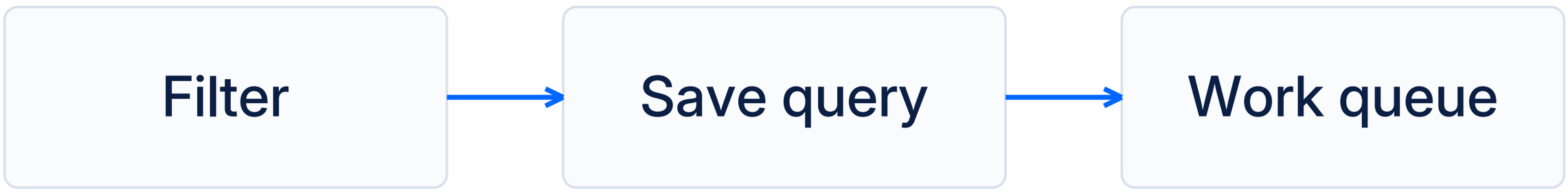


Server-side aggregations keep counts tied to the result set.

A search becomes a daily queue.

I turned saved searches into a daily support inbox. Agents see live queues; admins can enable, edit and reorder them.

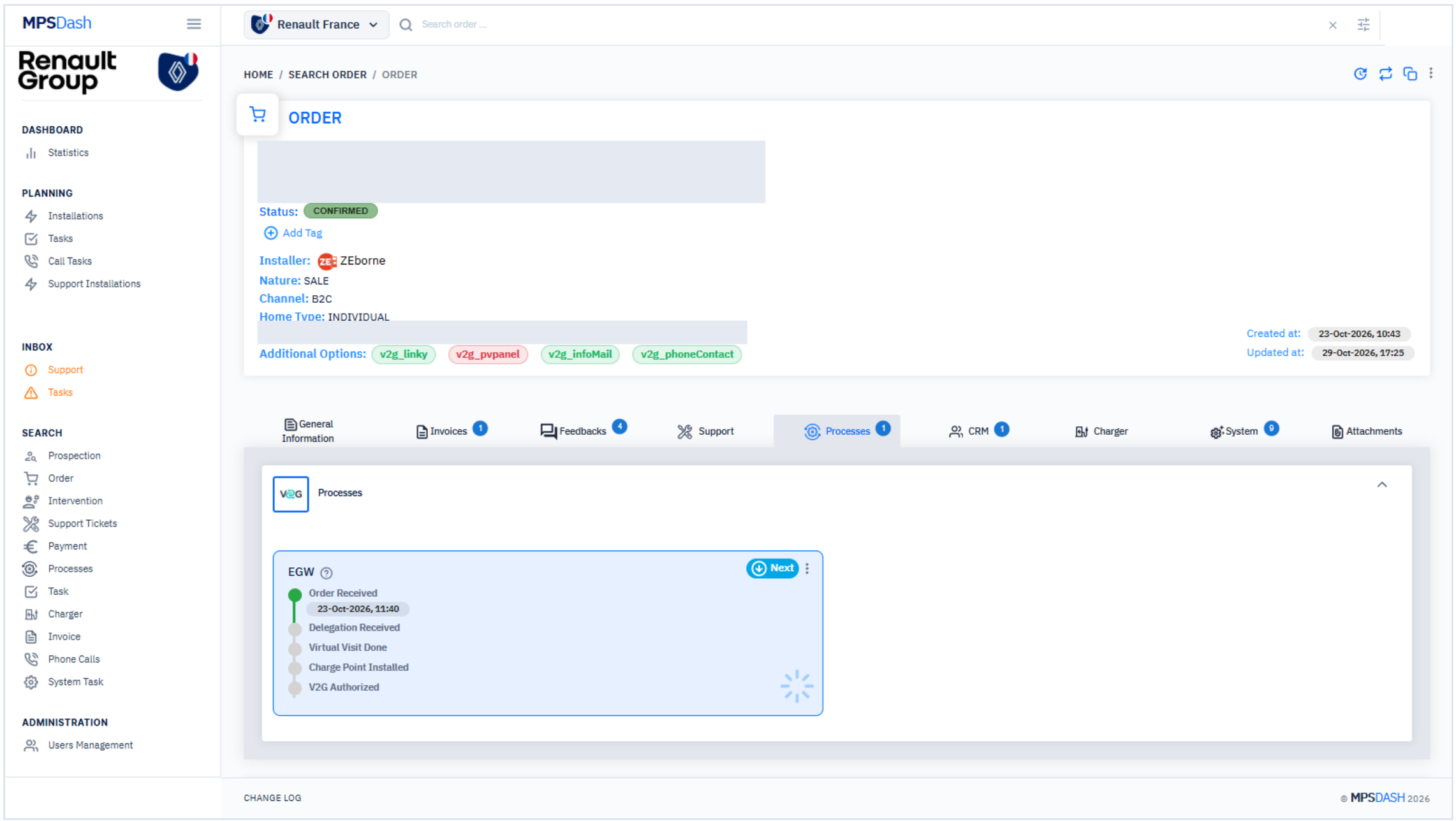


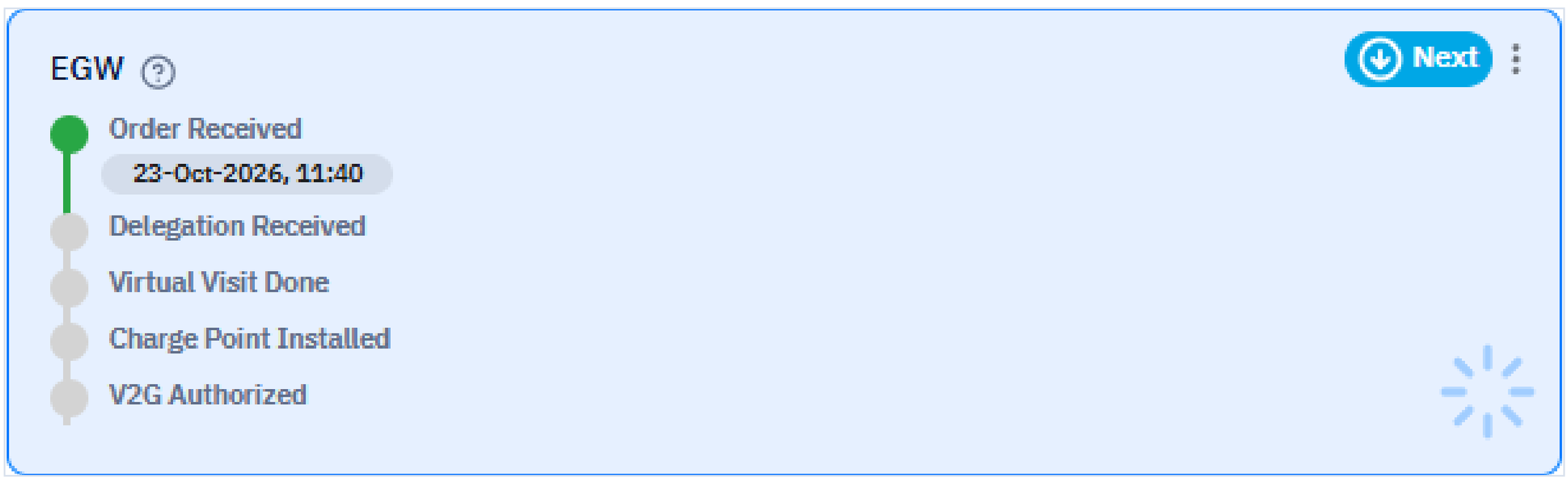


A query such as “orders waiting more than 3 days” becomes a recurring checkpoint, without a new screen for every exception.

The order keeps the story together.

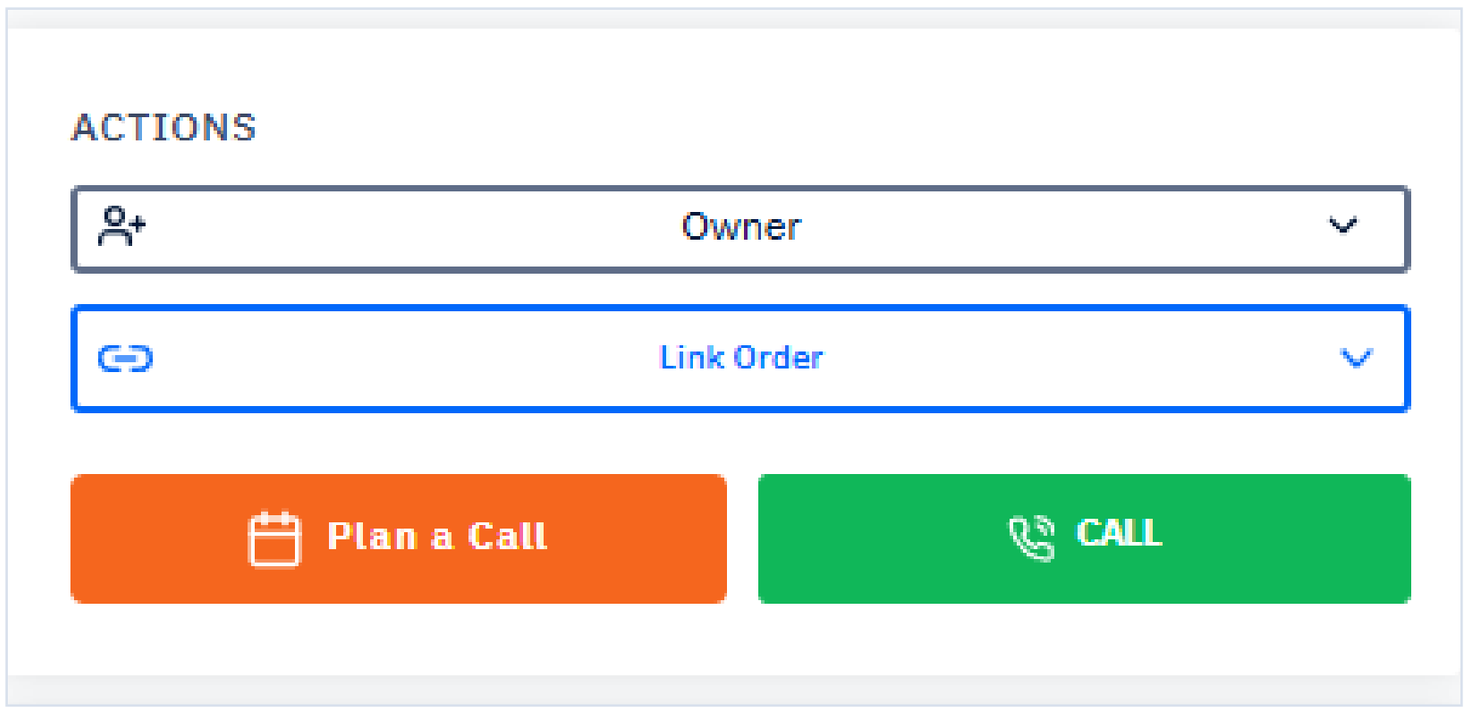
Billing, CRM, support, chargers and post-installation processes stay linked to the order. I kept that context visible across the detail pages.



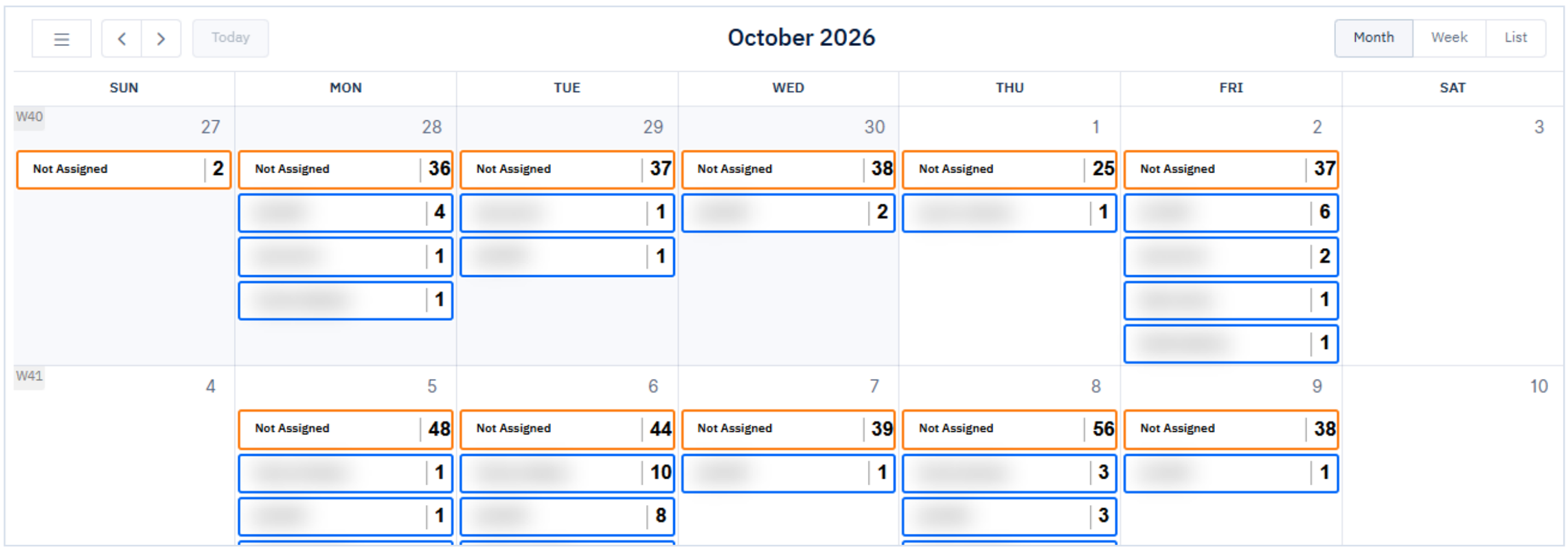


A desk for calls and follow-ups.

Leads, call tasks and support records stay linked to the order. Ringover actions let an agent call now or plan the next conversation.



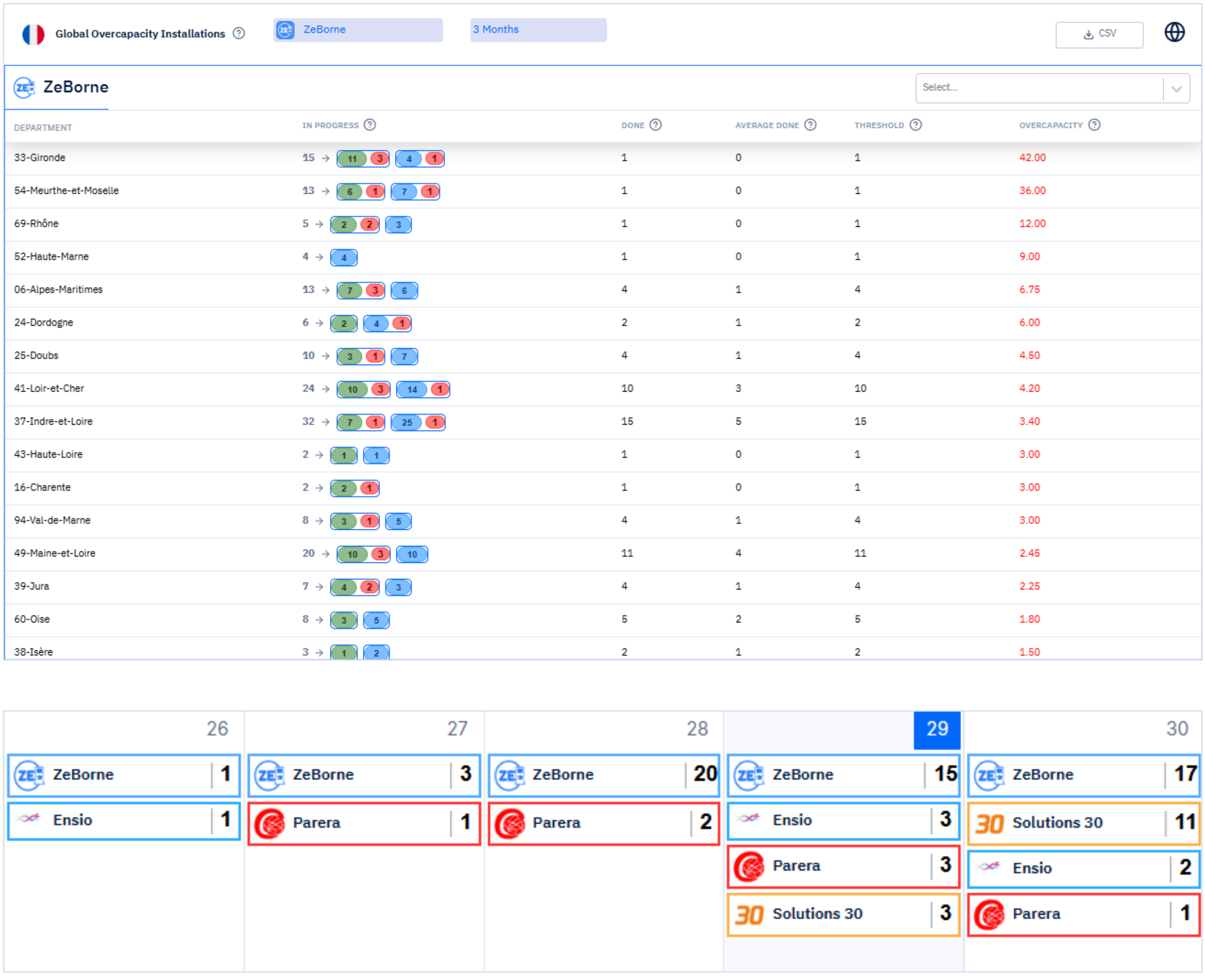
Call actions sit beside the lead and its linked order.



Tasks, planned calls and support tickets use connected records, with calendar views for daily follow-up.

Plan around the installer.

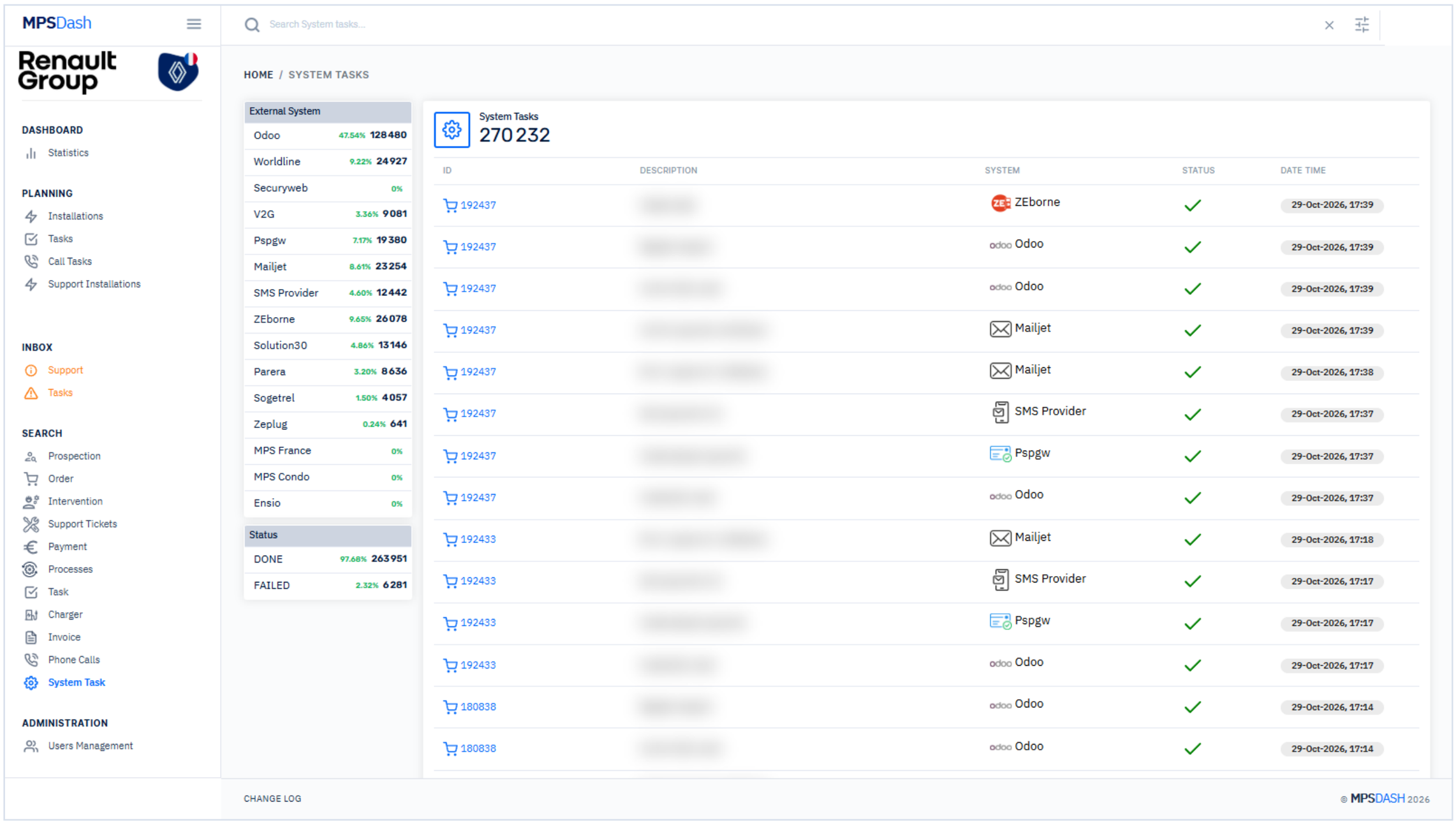
Work in progress, thresholds and average completed jobs sit side by side. Teams can inspect the installer and department before acting.

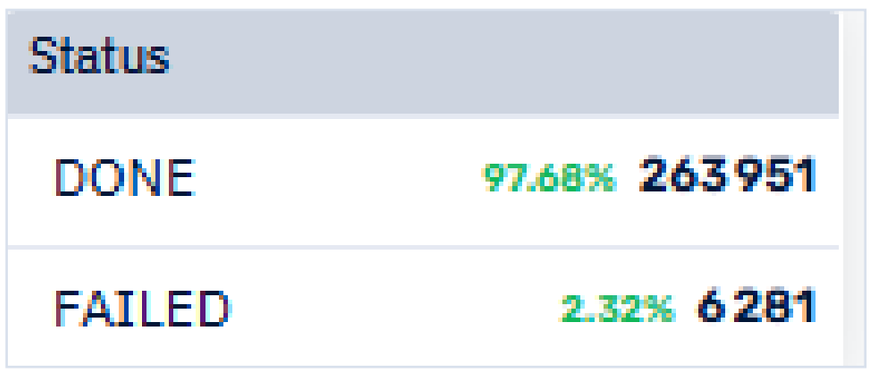


Capacity tables and installer calendars expose load and scheduling in different views.

Make the handoffs visible.

The system-task view connects an order to its external exchanges. Status, system and timestamp give support a concrete starting point for investigation.

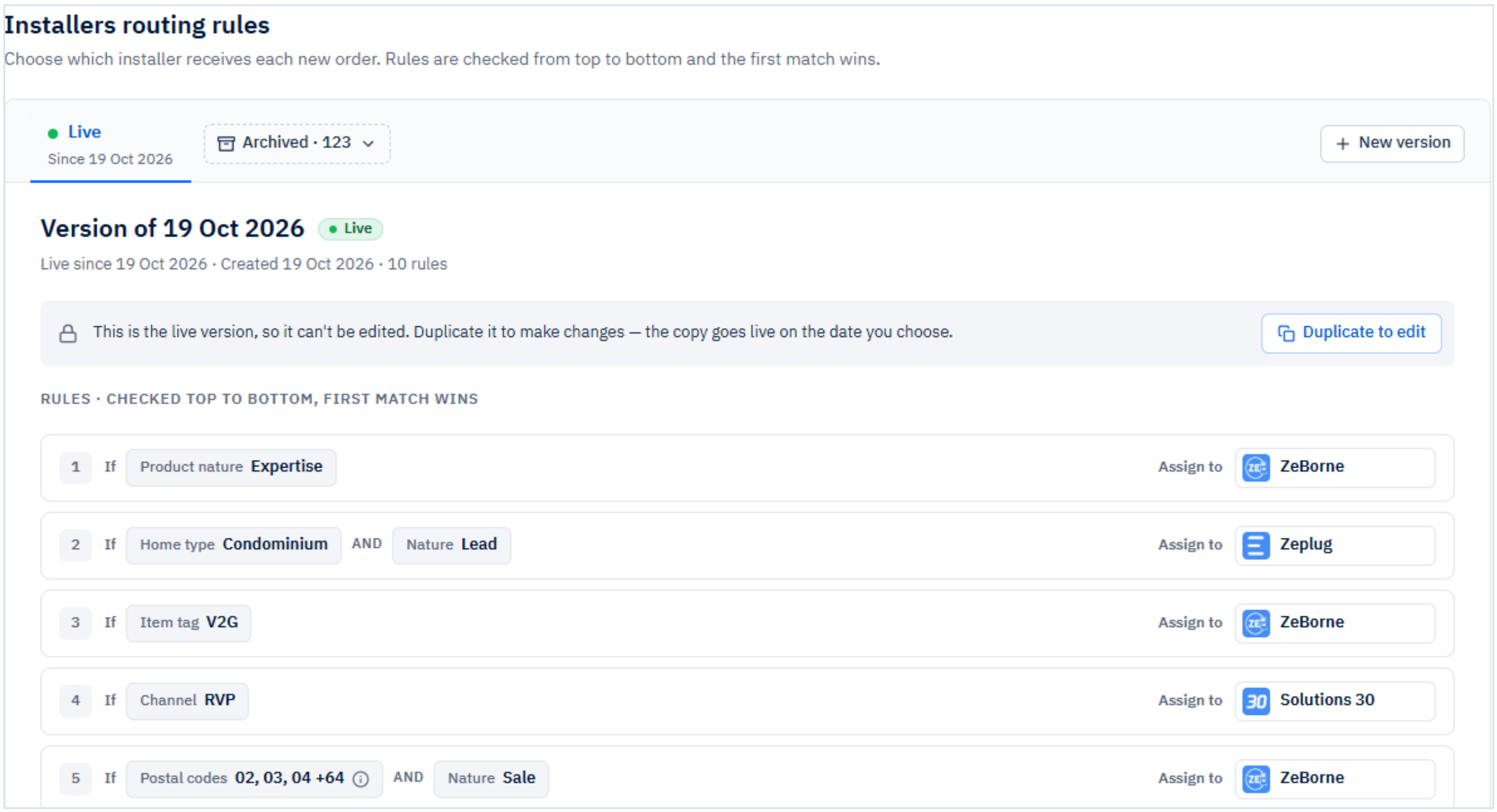




Payments and invoices track deposits, balances and validation. Exchange logs expose failures across ERP, payment, messaging and installer handoffs.

Rules that route real orders.

The first matching rule assigns the installer for each new order. I rebuilt drag-only controls and fragile state handling while keeping the save contract.



Versioned rules

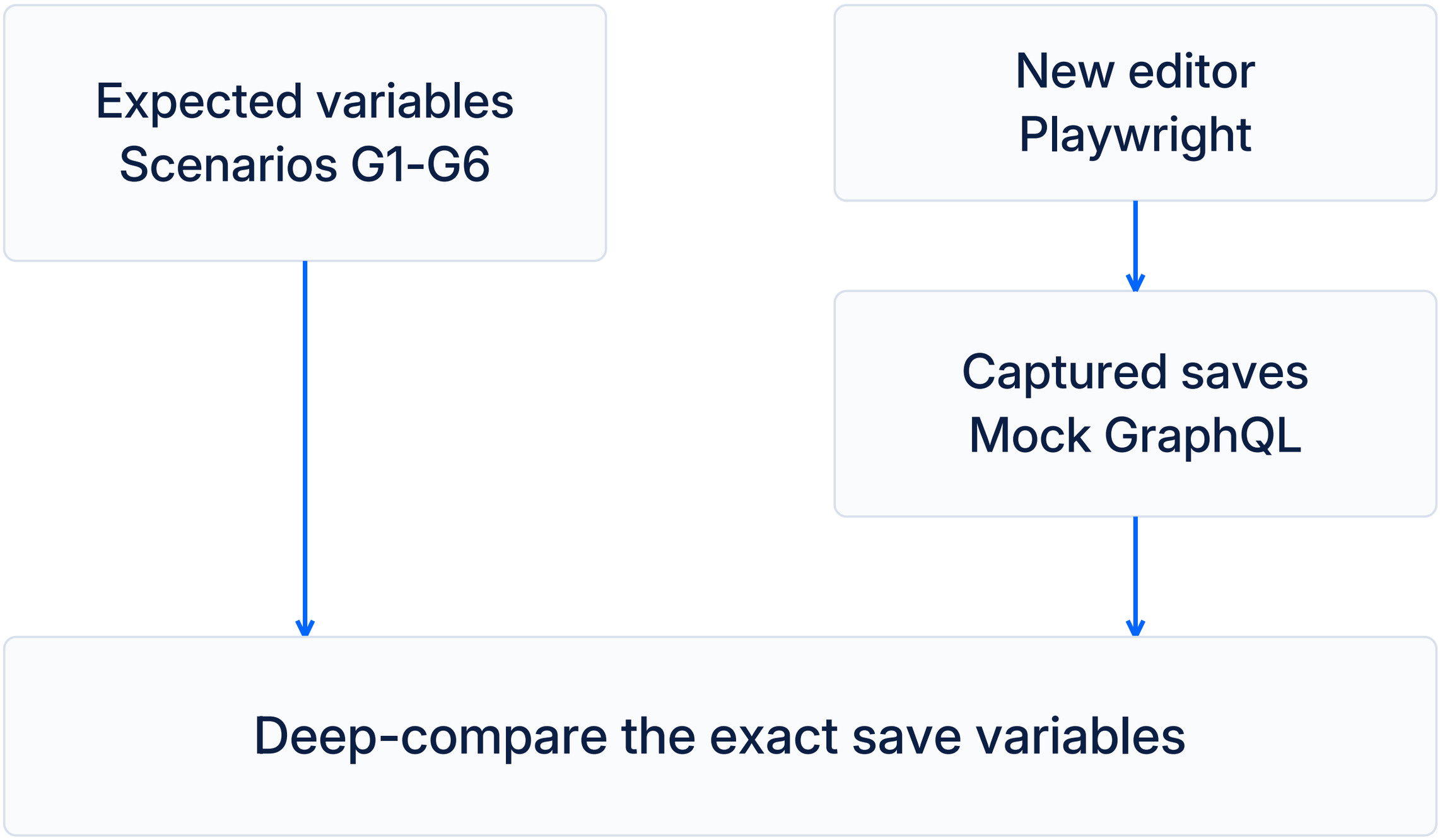
Live versions stay locked. Operators duplicate them and choose when the copy goes live.

Keyboard paths

Searchable conditions, focus return and Move up / Move down replace drag-only dependence.

Freeze the contract first.

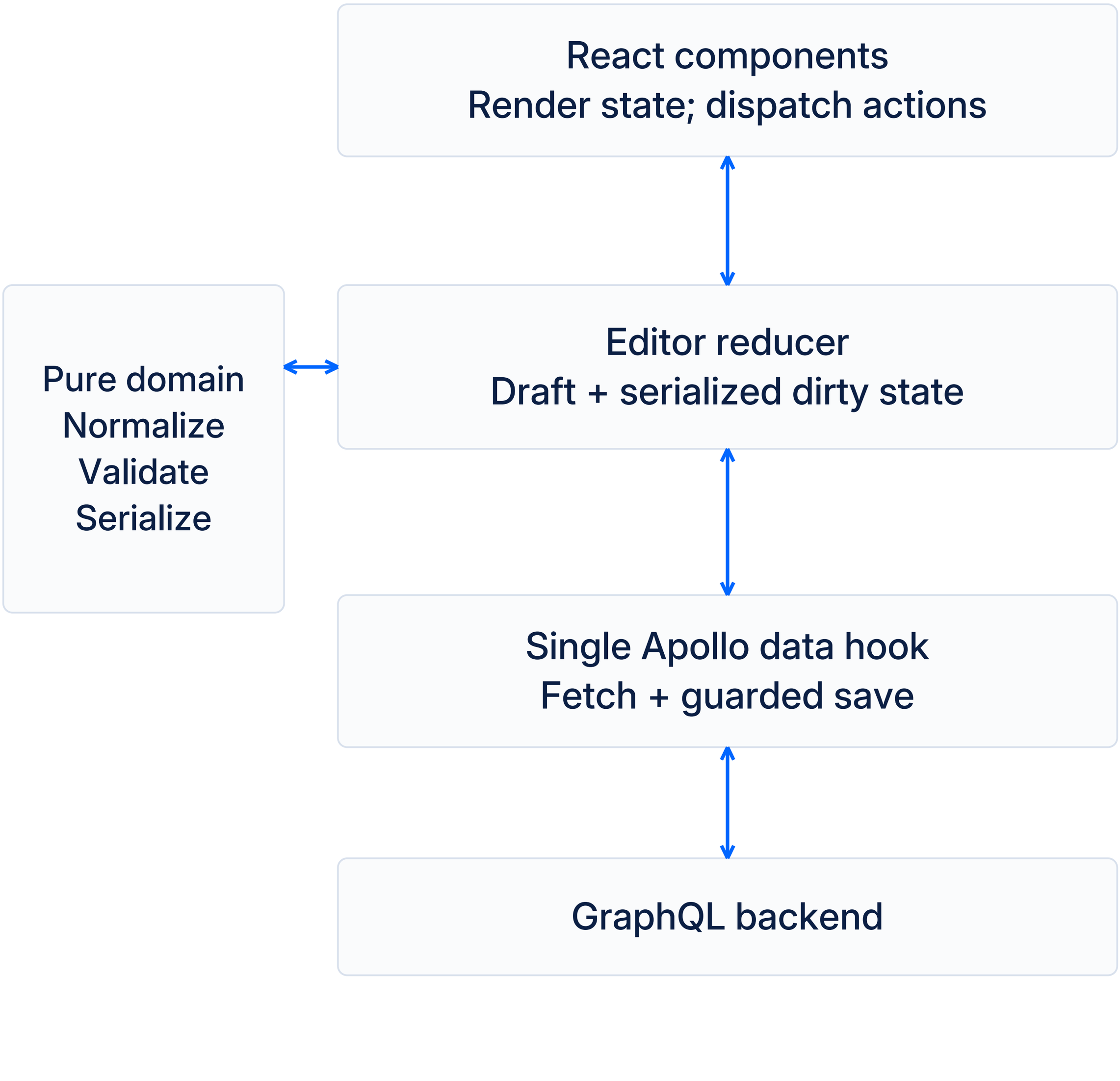
Before replacing the old editor, we recorded six real save scenarios. Playwright replays them in the new UI and compares the exact GraphQL variables.



Unknown filter types are preserved through serialization. These checks exercise the front end against a mocked backend.

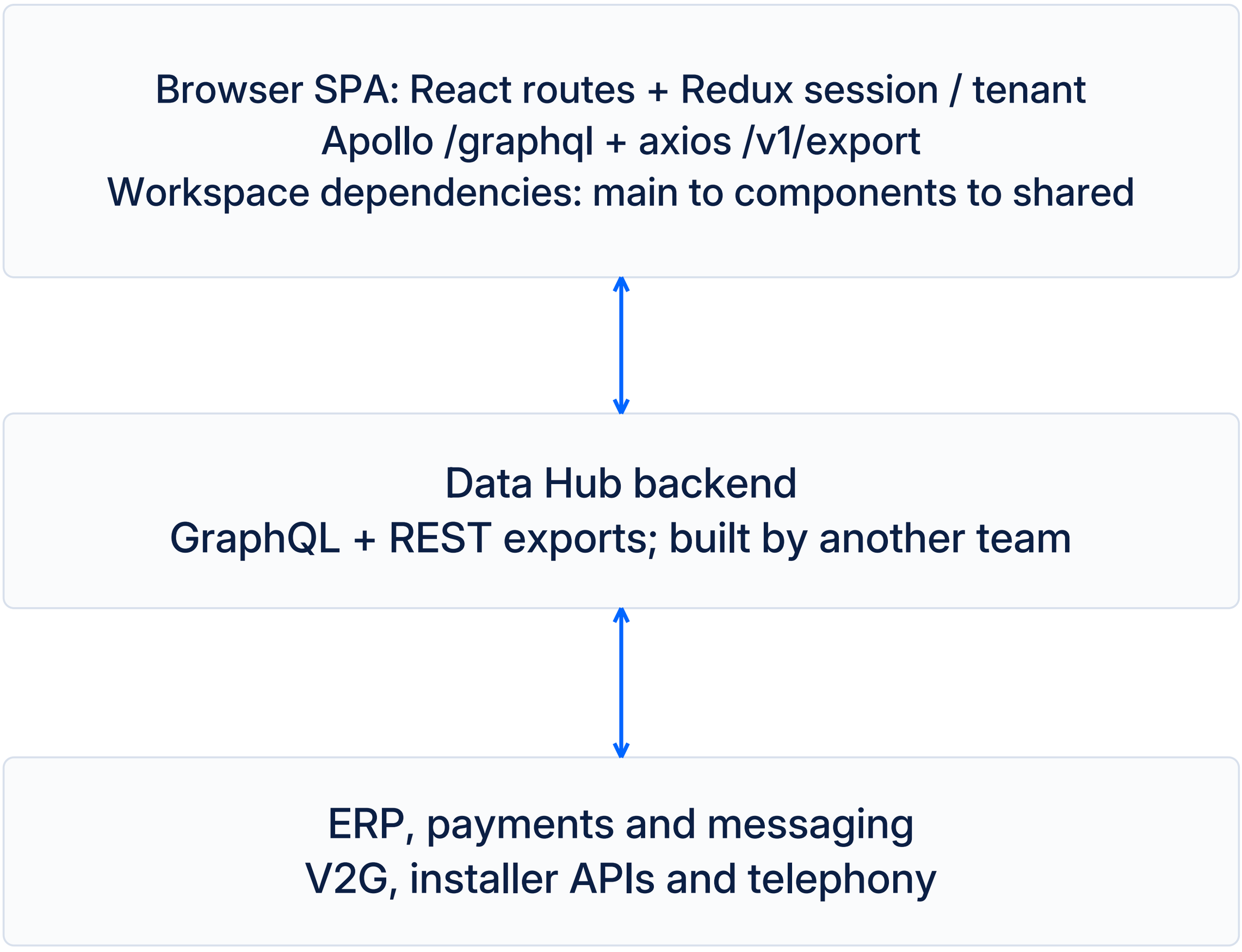
Put the hard logic outside React.

I separated registry-driven validation, draft state and network access. “Unsaved” means the serialized payload differs, so guards and the save bar agree.



One app. Clear system boundaries.

Routes check roles and permissions. Apollo supplies auth and tenant headers, with central handling for expired tokens, permission errors and server errors.



Docker + nginx serve the SPA; CI runs delivery checks.
Diagram reconstructed from the technical project description.

Modernize one step at a time.

I shipped the bundler and package-manager migrations separately, keeping the deployment contract stable and leaving each phase in production before the next.

4 August 2026

CRA to Vite 7

Keep build/ unchanged. Add browser shims and pre-bundle MUI/emotion to resolve legacy dependency issues.

---

5 August 2026

yarn/Lerna to pnpm 11

I cleared type errors across all three packages and added typecheck to CI.

Improve structure incrementally.

A lint baseline blocks new folder violations and removes entries when fixed, avoiding an all-at-once restructure.

Lint + structure	Typecheck	Tests	Build

I set the plan.
AI executes a bounded phase.

I wrote the scope, ground rules and quality gates. Claude Code carried out one phase per session; I reviewed each result before delivery.

I define a phase

Scope, constraints and acceptance gates.

↓

Claude Code implements

One bounded phase in one session.

↓

I review the result

Inspect changes against the written plan.

↓

Quality gates decide readiness

Lint, structure, types, tests and build.

AI-assisted development, directed and reviewed by me.

Built into daily operations.

France tenant, 29 October 2026. Operational record totals from the production captures.

37,832

Order records

34,930

Intervention records

191

User accounts, with role-aware routes and country tenant scopes.

179

Unit/component tests

25

Playwright tests

I introduced the first automated tests through the routing-editor redesign. The France snapshot counts order and intervention records at different stages.

An operational identity, kept intact.

MPSDash names the product. Renault Group and Mobilize identify the operating context. The reference keeps those marks intact.

Explicit	Name the action and show the state. Use direct verbs and concrete consequences.
Systematic	Keep navigation, filters and record details familiar across different kinds of work.
Contextual	Keep the tenant, linked order and operational status close to the decision.

Prefer “Duplicate to edit” and “Choose which installer receives each order”. Avoid promises such as “intelligent routing” when the workflow is rules-based.

Keep product and tenant identity distinct.

The product wordmark and the tenant lockup do different jobs. Keep them separate, aligned and on clear backgrounds.

MPSDash

Leave one M-cap-height of clear space around the wordmark.

Renault Group

Tenant identity stays beside the workspace, rather than becoming a new ELTO logo.

The marks shown here come directly from the product screens. The spacing rule is defined for this reference.

Make small-size limits visible.

For these screen-derived sources, use at least 32 px for the detailed tenant badge and 96 px for the wordmark. The tests below retain actual pixel sizes.



16 px 24 px 32 px 48 px 64 px

64 px

96 px

128 px

MPSDash

MPSDash

MPSDash

MPSDash

Do not stretch.

MPSDash

Do not use on dark.

Use the white field.

Retain source colors and proportions. Avoid stretching, clipping, recoloring, or placing a white-matte crop directly on a dark field.

Approved vector and reversed masters are needed for other background treatments.

Color carries a job.

The showcase palette follows the navy, blue and pale surfaces visible in MPSDash. These exact values define its editorial reference.

Structure

#0B1F44

RGB 11, 31, 68

Action

#0066FF

RGB 0, 102, 255

Secondary text

#4D6078

RGB 77, 96, 120

Surface

#FAFBFC

RGB 250, 251, 252

Canvas

#FFFFFF

RGB 255, 255, 255

Rules

#D8E0EC

RGB 216, 224, 236

Ink / white: 16.23:1

Blue / white: 4.83:1

Secondary / white: 6.44:1

Reference pairings meet the 4.5:1 text contrast target.

Type makes dense work readable.

Inter is the editorial typeface for this showcase, with four embedded weights and clear roles for headings, body copy and controls.

Clarity before density.

Know what needs action.

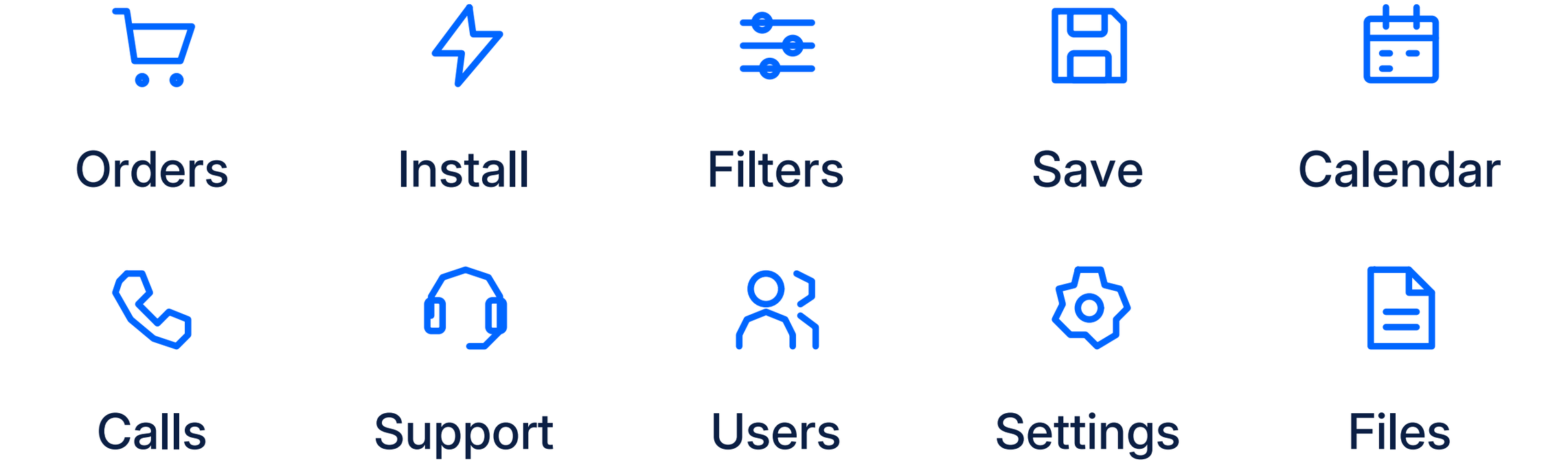
Find an order, inspect its history and decide the next step. Dense work becomes easier to scan when labels, values and actions have distinct roles.

72 / 78	Display, semibold
64 / 70	Headings, semibold
34 / 46	Body, regular
30 / 40	Reference text, regular

Inter 400, 500, 600 and 700 are embedded. Use 500 for controls and 700 for emphasis.

A familiar language for action.

Reconstructed controls capture selected interface patterns. The symbols represent real product functions; they are editorial reference assets.



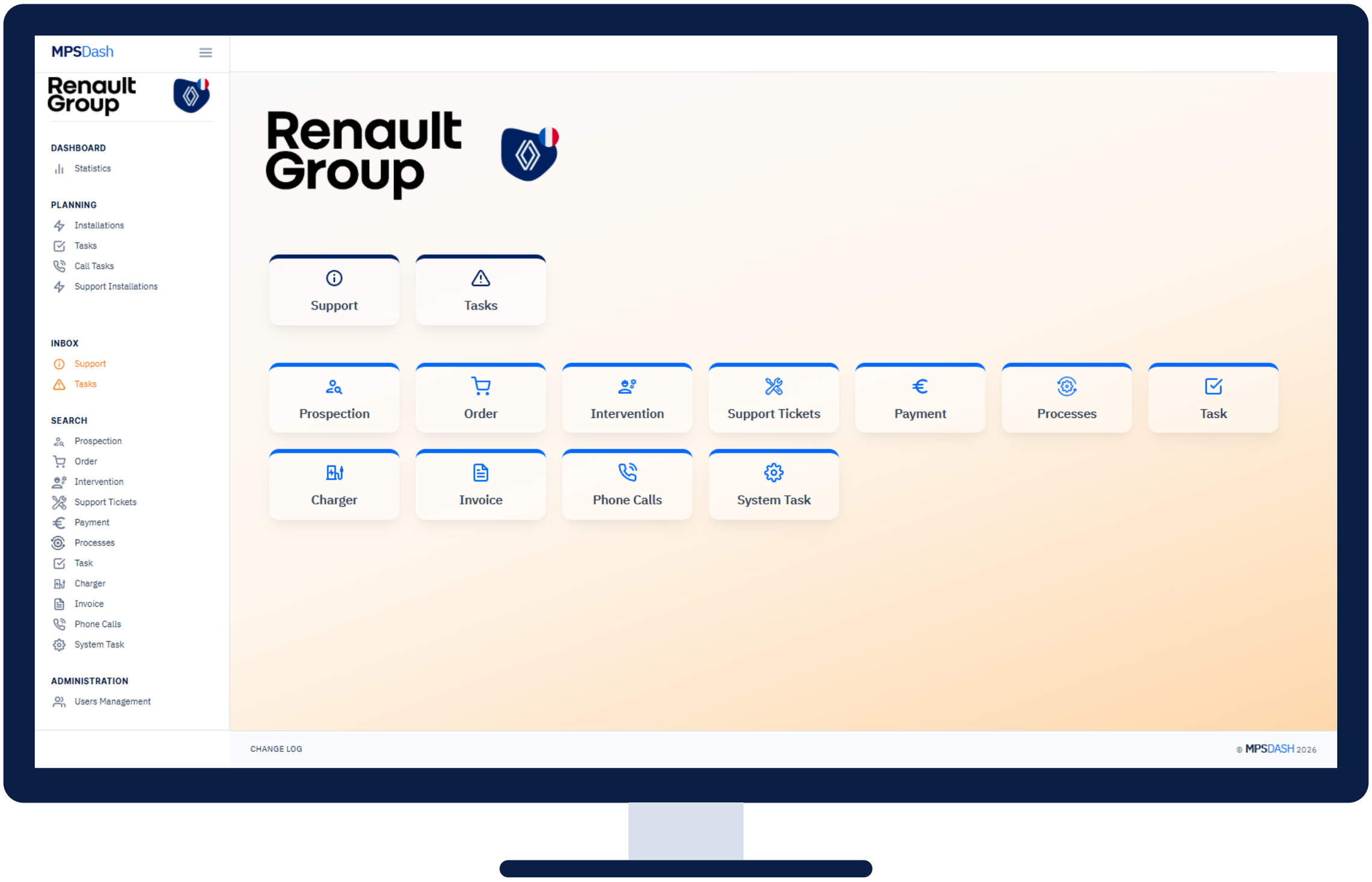
Resolve the rule before saving.

Live #087545 Attention #8B4B00 Error #B42335

Use 24-unit icons, a 1.7-unit stroke and rounded joins. Reference controls use 8-unit corners, 16-unit gaps and visible focus outlines.

The identity lives in the workspace.

The identity appears where teams work: the operational desktop and its in-app documentation. Product screens remain the main imagery.



Quick Start

The ELTO structure is based on ELTO Holding and various European operating entities.

To be present in the different European territories, ELTO Holding controls and relies on partners or subsidiaries established in the different countries.

ELTOs activity consists of organizing and selling charging solutions to 2 types of customers:

- B2B customers
- B2C customers.

Elto provides services directly to B2B or B2B2C customers, but also to B2C customers through automotive dealerships with the Mobilize Power Solution (MPS) brand.

Turn operational complexity into usable software.

I connect product decisions, interface design and front-end engineering, then use written plans and quality gates to direct AI-assisted delivery.

Taha Znazen

AI Product Engineer

Let’s talk about the product you’re building.

tahaznazen@gmail.com

LinkedIn · Taha Znazen