PRODUCTQUANT.DEV | JAKE MCMAHON

Event Taxonomy Builder

Turn tracking clutter into a system product, engineering, and analytics can trust.

EXECUTIVE SUMMARY

The problem is not missing events. It is missing agreement.

Most teams already collect data. They do not have a shared system for deciding what each event means, which properties belong on it, or how the taxonomy stays clean as the product grows.

The result is familiar: analytics can count activity, but it cannot answer product questions with confidence. Engineering ships new instrumentation without a naming standard. Product asks for a funnel view that does not exist. Leadership gets charts that are technically accurate and operationally useless.

ProductQuant's view is simple. A useful taxonomy is a decision system, not a spreadsheet. If the team can not read the same event language, then the dashboard is decoration, not infrastructure.

> The goal is not more tracking. The goal is fewer arguments about what the data means and faster decisions about what to build next.

FRAMEWORK

The 5-part taxonomy system

The report organizes the work into five layers because event naming alone never solves the full problem.

LAYER	WHAT IT DOES	WHY IT MATTERS
1. Naming rules	Defines how events and properties are named.	Stops drift and keeps the taxonomy readable across teams.
2. Functional coverage	Maps product areas and workflows.	Shows where event coverage is thin or missing.
3. Property design	Specifies the context each event must carry.	Makes analysis decision-ready instead of shallow.
4. Tracking plan	Turns the taxonomy into a build-ready spec.	Lets engineering implement without guessing.
5. Governance	Creates review and maintenance rules.	Prevents the taxonomy from decaying after launch.

> A tracking layer without governance is temporary. The taxonomy only matters if it stays coherent after the first implementation sprint.

WORKED EXAMPLE

One product question, two different failure modes.

The same dashboard can hide completely different problems if the taxonomy is too shallow.

Before taxonomy

Product asks: "Why did enterprise churn rise last quarter?" Analytics can show logins, page views, and a few generic clicks. It cannot tell whether the issue was onboarding failure, support failure, a missing workflow event, or a champion leaving.

After taxonomy

The team can see which workflow broke, which property changed, and which segment drifted. Two accounts that look identical at a high level no longer get the same response just because the dashboard was vague.

Question

Which behavior changed before churn?

Bad answer

Login dropped and a few events moved around.

Useful answer

Workflow completion stalled after configuration. The account needed onboarding recovery, not a generic save motion.

Outcome

The team stops treating every churned account as the same problem.

SAMPLE ARTIFACT

What a useful taxonomy looks like in practice.

The excerpt below is the kind of structure the full kit gives the team.

EVENT	PROPERTIES	WHY IT MATTERS
workflow_completed	workflow_name, account_tier, time_to_complete, completion_source	Shows whether the user reached the core value moment.
workflow_abandoned	workflow_name, step_index, error_type, last_action	Shows where the product is losing the user before value appears.
billing_risk_flagged	risk_type, account_age, renewal_window, owner	Separates billing pressure from product or support issues.
champion_changed	old_owner, new_owner, segment, reason_code	Explains churn risk that has nothing to do with usage volume.

A good taxonomy gives every event a job. If an event cannot help product, engineering, or analytics make a decision, it does not belong.

INCLUDED ASSETS

Eight implementation tools, organized around one system.

The full kit is a working package, not a concept document.

Diagnosis layer

Methodology guide and taxonomy audit template to decide what should be tracked.

Naming layer

Naming conventions and property schema guidance to keep the event language consistent.

Coverage layer

Functional area event map to show where the product is under-instrumented.

Build layer

Tracking plan template so engineering gets a clear implementation spec.

Quality layer

Review checklist and governance rules so the taxonomy stays clean after launch.

Rollout layer

Implementation playbook and quick start checklist to get the system into production.

The ProductQuant view

What good looks like once the taxonomy is installed.

STEP 1

Read the framework and align on the event language before adding more instrumentation.

STEP 2

Use the tracking plan to implement the missing events and properties that answer actual product questions.

STEP 3

Run governance so the taxonomy stays usable after the first launch sprint.

STEP 4

Use the full kit to make product, engineering, and analytics share one decision system.

The full kit is a separate **$97** purchase. The report excerpt gives you the logic; the kit gives you the tools.