Turning a Client Portal into a Cloud Analytics Platform for Renewable Energy

Context

Our client portal was a transactional tool — clients logged in to check the status of their solar installations, view basic account information, and submit support requests. It served its purpose, but it wasn't delivering strategic value.

Meanwhile, the business was paying significant licensing costs for Sigma to generate and display analytics reports to clients. The workflow was painful:

- Data lived in Databricks, siloed from the systems clients interacted with
- Creating a new report required coordination between data engineering, the Sigma platform team, and the client success team — often taking weeks
- Customizing reports per client was nearly impossible at scale
- Clients were asking for more granular, real-time insights about their renewable energy portfolios
- Leadership and sales wanted analytics capabilities as a differentiator, not just a cost center

The opportunity was clear: transform the client portal from a status-checking tool into a cloud analytics platform for renewable energy — and eliminate our dependency on Sigma in the process.

Action

This was a cross-organizational effort. I identified early that the technical challenge was straightforward compared to the alignment challenge. The project touched data engineering, application engineering, information security, data management, leadership, marketing, and sales. Everyone had different priorities and constraints.

Honing requirements across teams:

I partnered with **data engineers and data managers** to understand what data existed in Databricks, how it was structured, and what transformations were already in place. I worked with **marketing and sales** to understand what analytics capabilities would resonate with prospects and existing clients. I collaborated with **management and leadership** to frame the project as a platform investment — not just a feature.

With **information security**, I established the data access patterns and ensured that client data isolation was baked into the architecture from day one, not bolted on after.

Designing the data pipeline:

The core technical challenge was getting data out of Databricks and into a form that could power fast, flexible, client-facing reports. I designed a system with several key components:

- **ETL pipelines** to extract data from Databricks into a dedicated reporting database, decoupling analytics from the production data warehouse
- **Materialized views** that pre-compute common aggregations and metrics, serving as the foundation for both internal dashboards and external client reports
- **A flexible report schema** that allows new report types to be defined through configuration rather than code — enabling the creation of new reports in a day or less

The architecture was intentionally designed so that the same materialized views could power both internal analytics (for operations, sales, and leadership) and external client-facing reports, eliminating duplicate data pipelines.

Building for customization at scale:

One of the key requirements from sales was per-client report customization. Rather than building bespoke reports for each client, I designed a system where reports are composed from reusable data components and layout templates. Client-specific customizations are configuration, not code.

This meant a single engineering team could support hundreds of client configurations without the linear scaling problem that Sigma had imposed.

Building alignment and delivering iteratively:

I organized regular syncs with **engineering** to coordinate implementation, **data engineering** to validate pipeline reliability, and **leadership** to demonstrate progress. I worked with **sales** to ensure the platform narrative matched what we were actually building, and with **marketing** to position the analytics capabilities for prospects.

The project was delivered incrementally — internal reports first (proving the pipeline), then client-facing reports (proving the platform), then customization capabilities (proving the scalability).

Result

The impact was both immediate and structural:

- **Eliminated our reliance on Sigma** for client-facing analytics, removing a significant recurring licensing cost
- **New reports can be created in a day or less** — down from weeks under the previous workflow
- **Client report customization became a configuration task**, not an engineering project — enabling the client success team to tailor reports without developer involvement
- **The client portal evolved from a transactional tool to an analytics platform**, becoming a key differentiator in sales conversations
- **Internal teams gained access to the same data foundation**, reducing the "two versions of the truth" problem between internal and external reporting
- **Data pipeline reliability improved** — materialized views provide predictable performance regardless of underlying data volume changes in Databricks

The platform architecture also positioned the company to add new data sources and report types without rearchitecting — the foundation scales horizontally.

Learning

Alignment is the bottleneck, not architecture. The ETL pipelines and materialized views were well-understood patterns. Getting engineering, data engineering, infosec, leadership, marketing, and sales to converge on a shared vision — and stay converged through delivery — was the actual hard problem.

Design for the report you haven't built yet. The biggest architectural win was making report creation a configuration task. Solving for today's reports would have been faster initially but would have created a linear scaling problem. Investing in flexibility early paid off within months.

Eliminate intermediaries, not just tools. Replacing Sigma wasn't the goal — eliminating the workflow bottleneck was. The value wasn't in removing a vendor; it was in giving the right people direct access to create and customize reports without waiting in a queue.

Materialized views are an underrated architectural pattern. They served as the clean boundary between messy source data and fast, reliable reporting. They also made it possible to serve both internal and external consumers from the same data without building parallel pipelines.

Cross-functional delivery requires cross-functional communication. Partnering with marketing and sales wasn't optional — it ensured that what engineering built matched what the business was promising. That alignment prevented the "we built the wrong thing" failure mode that kills platform projects.