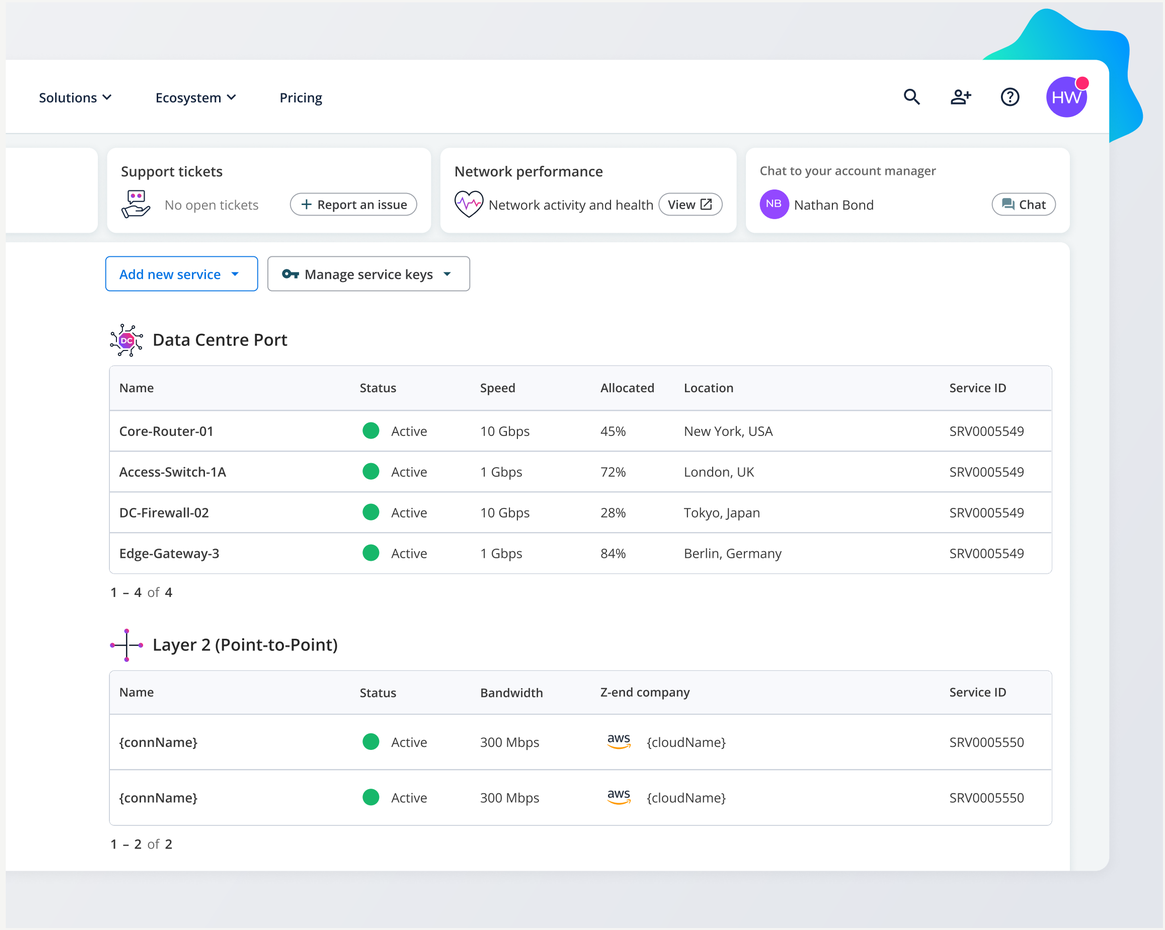
CONSOLE CONNECT

Designing a dashboard that could scale with the product

A scalable dashboard that gives customers a clear overview of their services, with useful signals and direct paths into the wider platform.

Henry Wahyu
Lead Designer
Product Design · Interaction Design · Systems Thinking



CATALYST

It still worked.
I saw it wouldn’t scale.

Legacy products were moving onto a new marketplace architecture. The dashboard could keep running, so rebuilding it was not initially a Product priority.

I made the case for redesigning it before more products brought more bespoke UX, frontend work and maintenance.

More products migrating



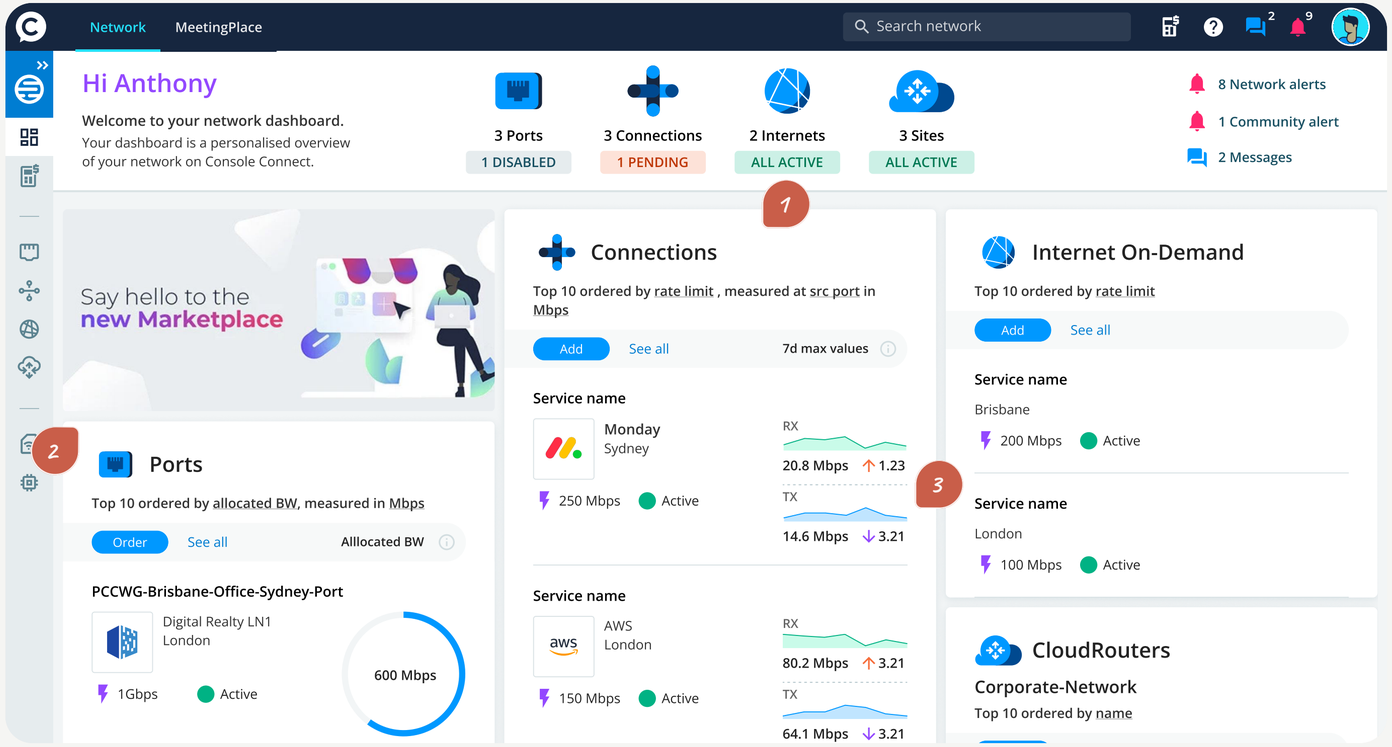
More bespoke widgets



More design, frontend and maintenance work

STARTING POINT

A dashboard built product by product



1. Bespoke widgets

Useful context, rebuilt for each major product.

2. Masonry layout

Reading order was hard to control.

3. Data-heavy visuals

Loading grew more complex as widgets accumulated.

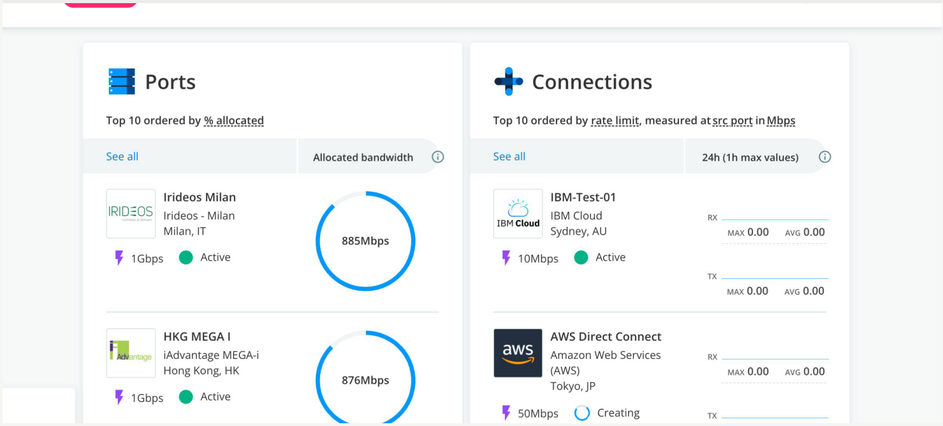
CORE TENSION

Useful context vs a model that could scale

Bespoke product widgets

Useful product-specific context

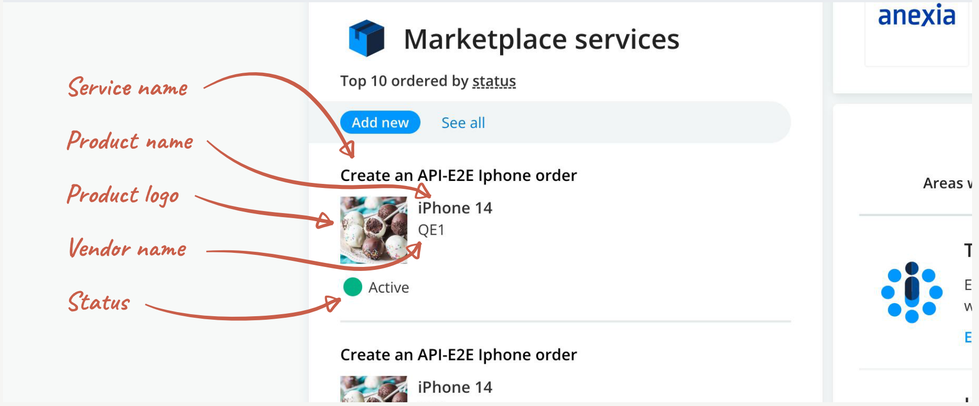
Recurring design + frontend work
More UI to maintain



Generic service views

One consistent pattern across products

Only information common to every product
Less useful product-specific detail



How could we preserve useful product-specific information without designing bespoke dashboard UI for every product?

DISCOVERY

The dashboard’s job was different from what we assumed

The existing UI showed what we had built. It could not tell us what customers needed.

What did we need to learn?

How customers monitored services,
what brought them to the portal,
and what they needed next.

What research showed

Many customers were API-first,
with their own monitoring
and automation.

How the brief changed

Orient: what services do I have?
Notice: what needs attention?
Continue: where do I go next?

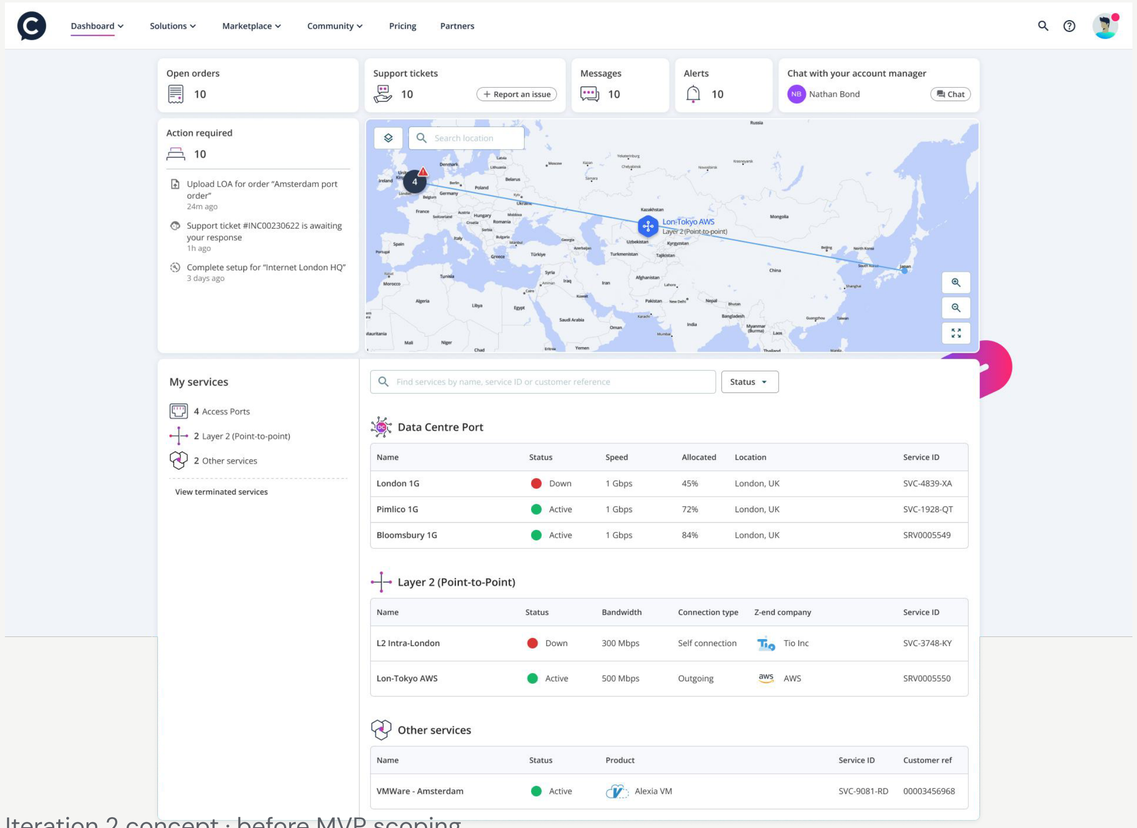
Overview + Attention + Launchpad

I shaped learning objectives and questions with our UX Researcher, joined interviews as a secondary researcher where possible, and used debriefs to shape the direction.

UX Researcher led recruitment, facilitation and detailed synthesis.

DESIGN DIRECTION

From product widgets to a reusable architecture



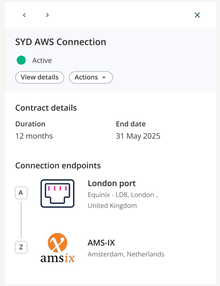
Iteration 2 concept · before MVP scoping

Overview + attention

A deliberate hierarchy and mini-widget zone, organised around user intent.

Detail by service

Shared tables with product-specific columns. Other Services as the fallback.



Quick View

Explored — deferred from first release

A second layer for utilisation, linked services and contracts.

STRUCTURE

A deliberate grid restored hierarchy

I replaced masonry with flexible rows and columns, so reading order could be intentional.

My HTML and CSS understanding informed how the layout could respond to different widths.

Design intent
Build hierarchy through deliberate placement of contents/info on the dashboard.

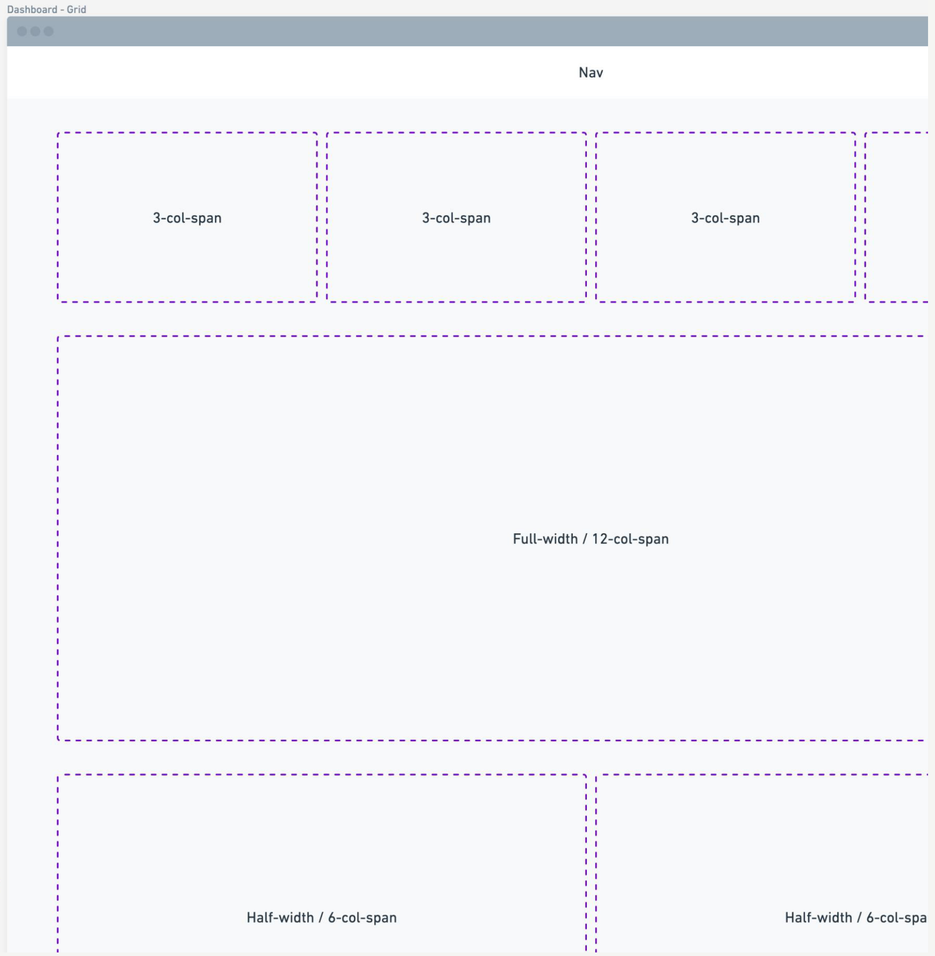
A row & column based grid allows for more control over the placement of contents.

Page contents are laid out in a vertical stack consisting of rows.

Each row can contain between 1 - 4 widgets.

Referring to the 12-columns grid, each widget's width can span:

- 3 columns
- 6 columns
- 9 columns
- 12 columns



Structure explored in wireframes before hi-fi iteration 1.

SERVICE ARCHITECTURE

One shared pattern. Different information by product.

Shared interaction and component → product-specific columns → generic fallback

Name	Status	Speed	Allocated	Location
Core-Router-01	Active	10 Gbps	45%	New York,

Name	Status	Bandwidth	Z-end company	Service ID
Sydney AWS Zone 1	Active	300 Mbps	AWS	S4821093

Name	Status	Network scope	No. of connections	Service ID
cr-us-east-prod-01	Active	Global	1	S1058374

Name	Status	Product	Service ID	Customer ref
{serviceName}	Active	{product Name}	SRV0005552	00003456968

Ports

Speed + allocation

Connections

Bandwidth + endpoint

CloudRouters

Network scope + connections

Other Services

Generic fallback when a specialised view is not ready

Specialised dashboard UI became an enhancement rather than a prerequisite for releasing a product.

EVALUATION

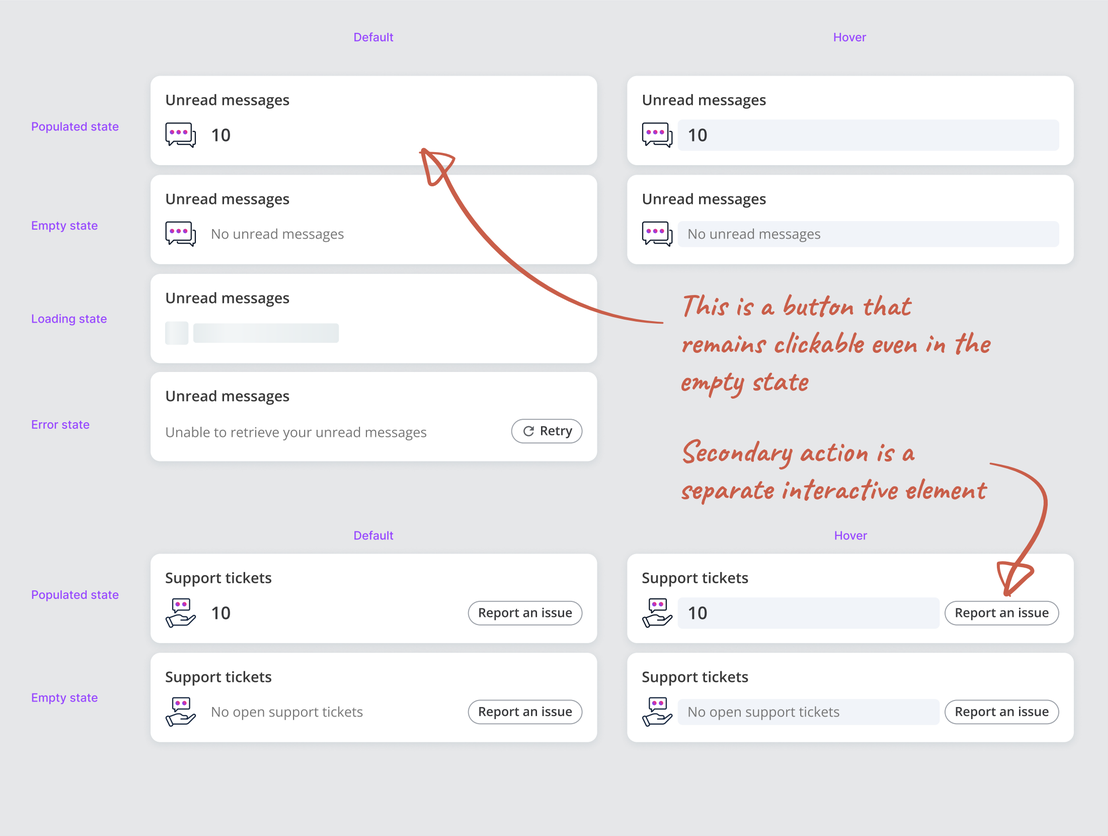
The launch point had to survive every state

With the direction established, I worked with our UX Researcher to test the interaction model.

A number-only link vanished at zero. Users also expected the whole mini-widget to be clickable.

A fully clickable card conflicted with accessible secondary actions. With developers and our Design Engineer, I revised the pattern.

Persistent launch point · larger hit area · separate secondary action



Wireframes → hi-fi 1 → unmoderated testing → iteration 2 → moderated usability test

EVALUATION

Testing exposed a deeper status-model problem

Platform “Active”

Provisioning complete
Commercial lifecycle active
Billing begun

≠

User interpretation of green

The network service is healthy and operational.

A commercially active service could still have an operational problem.

This needed broader product and technical work. Not a local label or colour fix.

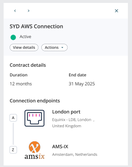
Unresolved in the first release

SCOPING

The core structure stayed. Not every concept did.

DEFER

Quick View

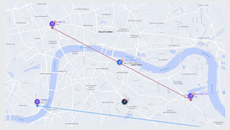


The most useful data and frontend/backend support were not ready; migration was incomplete. The existing detail journey remained.

With Product + Engineering

DEPRIORITISE

Geographic map

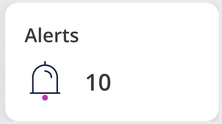


Research and network-domain feedback did not establish enough actionable value for the complexity.

With Product + Engineering

DROP

Alerts

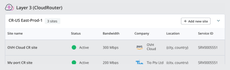


The underlying capability already needed broader work. A dashboard widget would not resolve that.

My recommendation

SIMPLIFY

CloudRouter nesting



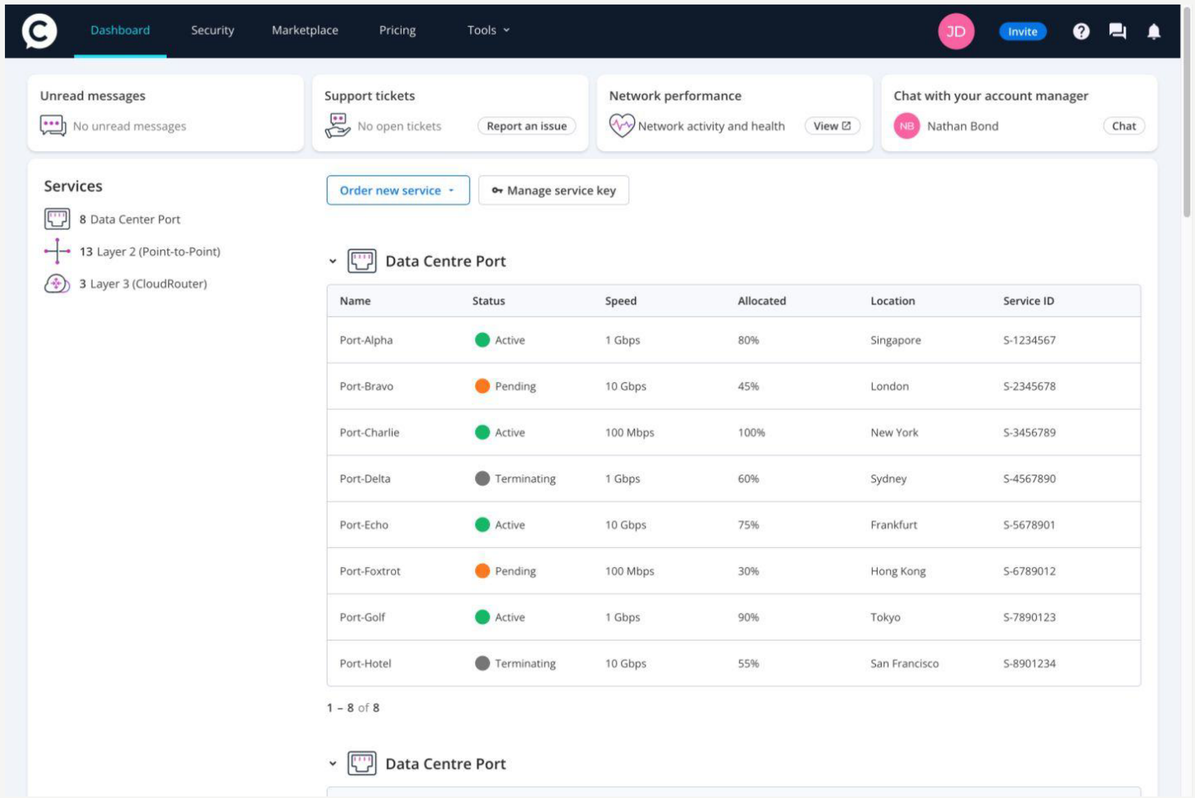
Usability testing showed the richer interaction still needed refinement.

My recommendation

The core structure stayed. The parts that weren’t ready were deferred, simplified or dropped.

TECHNICAL VALIDATION

Testing the responsive model in React



prototype.dashboard.wahyu.me ↗

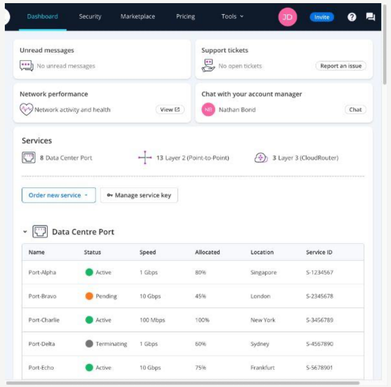
Existing system components

Representative mock service data

Responsive behaviour in-browser

Deployed on Netlify

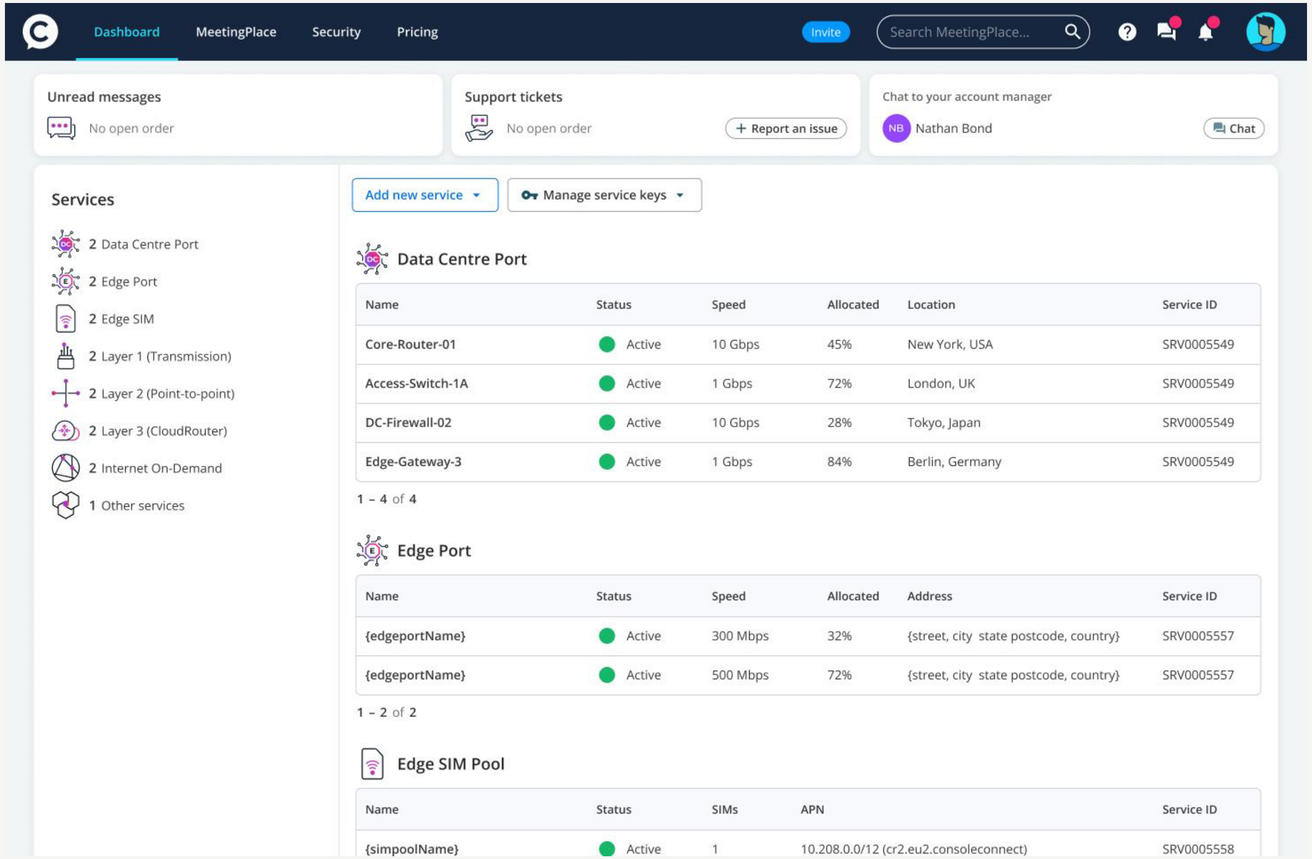
Implementation prototype · later product state



AI assistance helped me move faster; the useful output was a working layout I could inspect.

WHAT SHIPPED

Overview first. Detail by service.



SHIPPED

Deliberate hierarchy

Modular mini-widget area

Service summary

Shared service tables

Product-specific information

Other Services fallback

Quick View remained deferred.

OUTCOME

The question changed

BEFORE

How should this product appear on the dashboard?

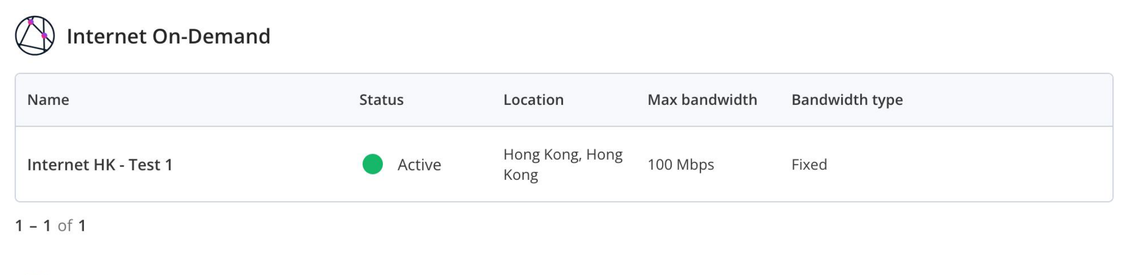
An open-ended design and implementation question.

AFTER

Dedicated table or Other Services? What should the table surface?

The structural pattern was already established.

INTERNET ON-DEMAND · EXISTING TABLE PATTERN



Internet On-Demand

When it later migrated, the interaction pattern and component already existed. The remaining design work was mainly deciding which product-specific attributes were useful to surface.

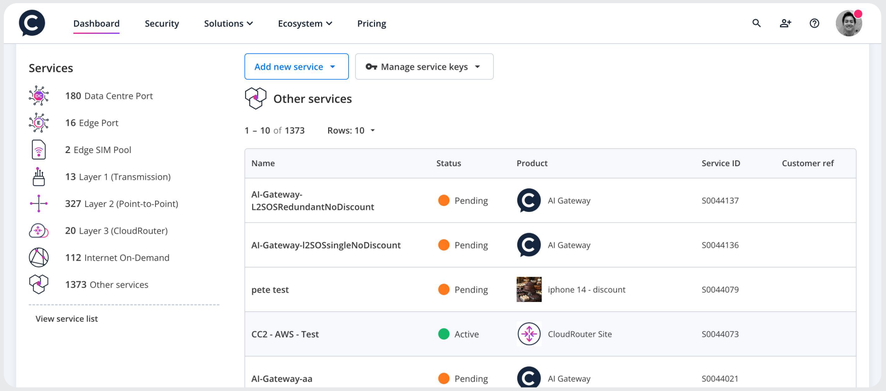
The dashboard no longer had to be redesigned product by product.

SUPPORTING PROOF

The pattern kept working

CloudRouter Sites could fall back

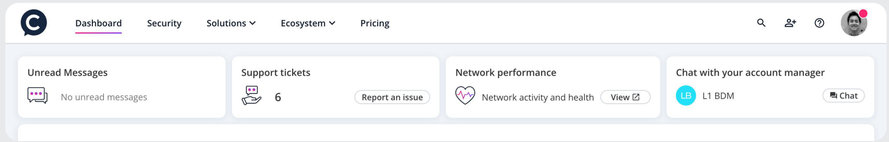
When the richer interaction was not ready, Sites could use Other Services without another temporary dashboard treatment.



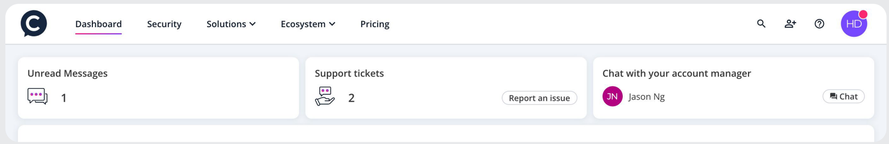
Another mini-widget could fit

A later conditional Network performance widget needed no restructure.

With access



Without access



Qualitative reception

Team-retro feedback was positive about the dashboard direction, loading states and polish.

The lasting value wasn’t any single widget or interaction.

It was creating a system that gave the product room to change.

