Global Container Tracking

The World leader in AI-driven supply chain transformation

FOURKITES

Role: UX Lead Designer / UX Manager

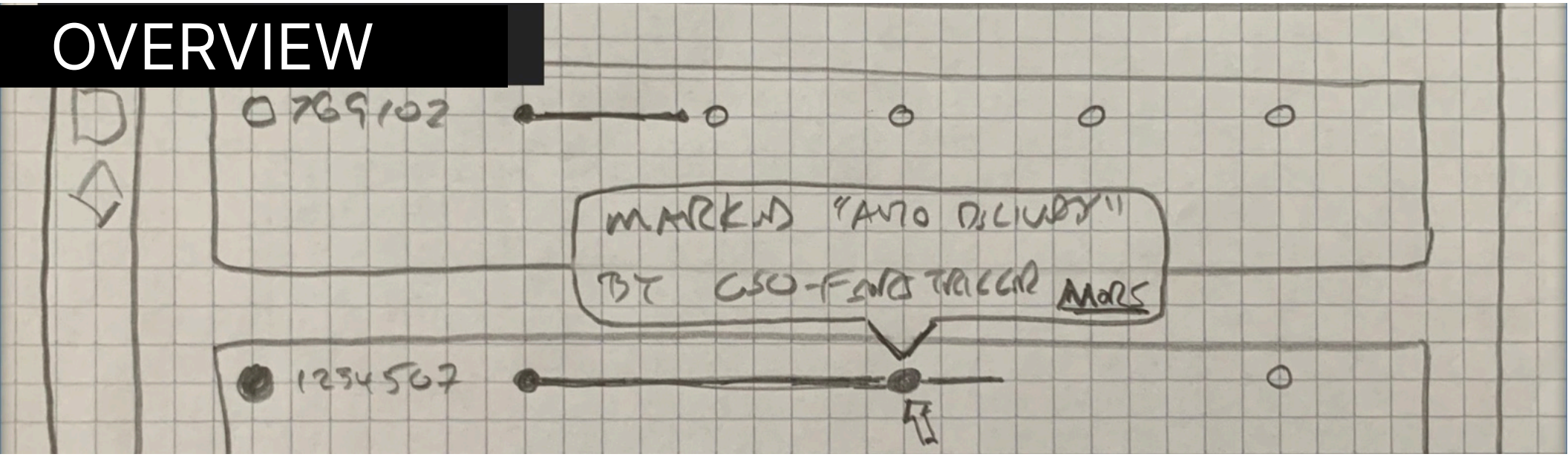
- Led UX strategy and design for the complete redesign
- Ran cross-functional discovery with data science, carrier integration, and maritime operations teams
- Designed the data confidence model and milestone taxonomy that became foundation for future features
- Created information architecture, user workflows, interactive prototypes, and final UI
- Collaborated closely with engineering through implementation and launch
- Conducted customer research and usability testing throughout development

Challenge: Redesign container tracking to handle the complexity of ocean freight workflows

Global Tracking

FourKites

OVERVIEW



FourKites is a global supply chain visibility platform used by enterprise shippers. As maritime freight volume grew, our customers increasingly struggled to track their containers across multiple carriers, ports, and partners.

The existing experience wasn't built for the complexity of ocean workflows. Data quality varied by carrier, visibility was inconsistent, and users had to reconcile information manually, often outside the platform.

I led the redesign of FourKites' Global Container Tracking experience to give customers a single, trustworthy view of their international freight.

PROBLEM

Customers couldn't confidently answer the most basic question:
"Where is my container, and is it on schedule?"

The challenges included:
Fragmented data ecosystem
Information came from carriers, ports, third-party sources, and internal predictions. Each source had different reliability, update frequency, and completeness. Users had no way to know which data to trust.

Inconsistent milestone updates and unreliable ETAs
One carrier reports departure, another reports arrival, a third has no update at all. Estimated times of arrival changed without explanation. Users couldn't tell if delays were real or just data lag.

No clear indicator of data confidence or source
Platform showed information without context. Was this a verified carrier event or an algorithmic prediction? Fresh data or days-old update? Users had no way to assess reliability.

Exceptions hidden deep in the interface
Delays, reroutes, and missed connections were buried in tables or scattered across different views. By the time users found problems, they'd already caused downstream disruptions.

High support volume because customers couldn't self-diagnose issues
When users couldn't understand what was happening, they escalated to support. Support teams spent hours reconciling conflicting data sources and explaining system behavior.

The result wasn't just confusion. It created erosion of trust in the platform.
Users started maintaining their own spreadsheets, calling carriers directly, and treating FourKites as one data source among many rather than the source of truth. This undermined the core value proposition.

Global Tracking

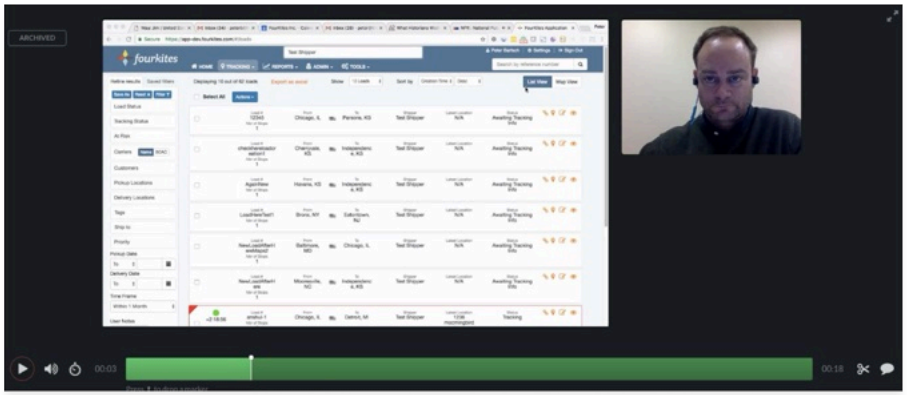
FourKites

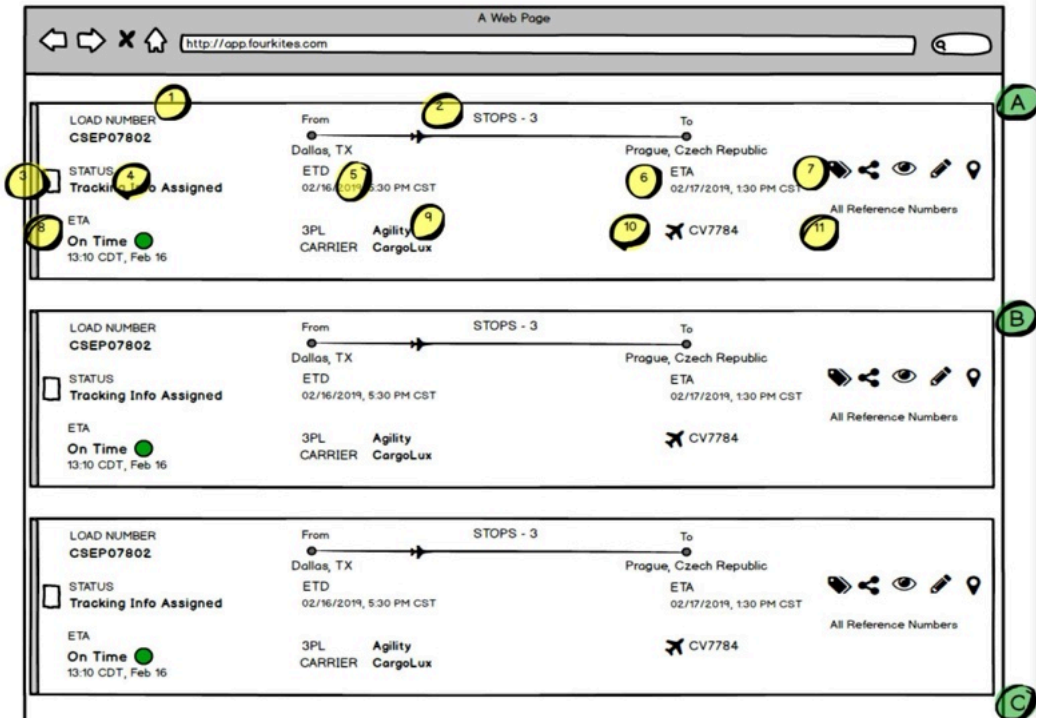
GOAL

Design a unified, intuitive interface that:

- Provides end-to-end milestone visibility across the entire container journey
- Makes data confidence and source transparency obvious at a glance
- Surfaces delays and exceptions at the moment they matter, not after the damage is done
- Reduces the need for manual tracking, spreadsheet reconciliation, or support escalations
- Rebuilds customer trust in the product's maritime data

This wasn't just about making things prettier. It was about making the system trustworthy enough that customers would rely on it for critical decisions.





APPROACH

1. Mapping the Data Ecosystem

I partnered with data science, carrier integrations, and maritime operations teams to understand how data flowed through the system.

What I learned:

Data sources varied wildly in quality
Major carriers (Maersk, MSC, CMA CGM) provided structured, real-time updates via API. Smaller carriers sent sporadic email updates. Some ports had live systems, others updated once daily. Third-party aggregators filled gaps but with varying accuracy.

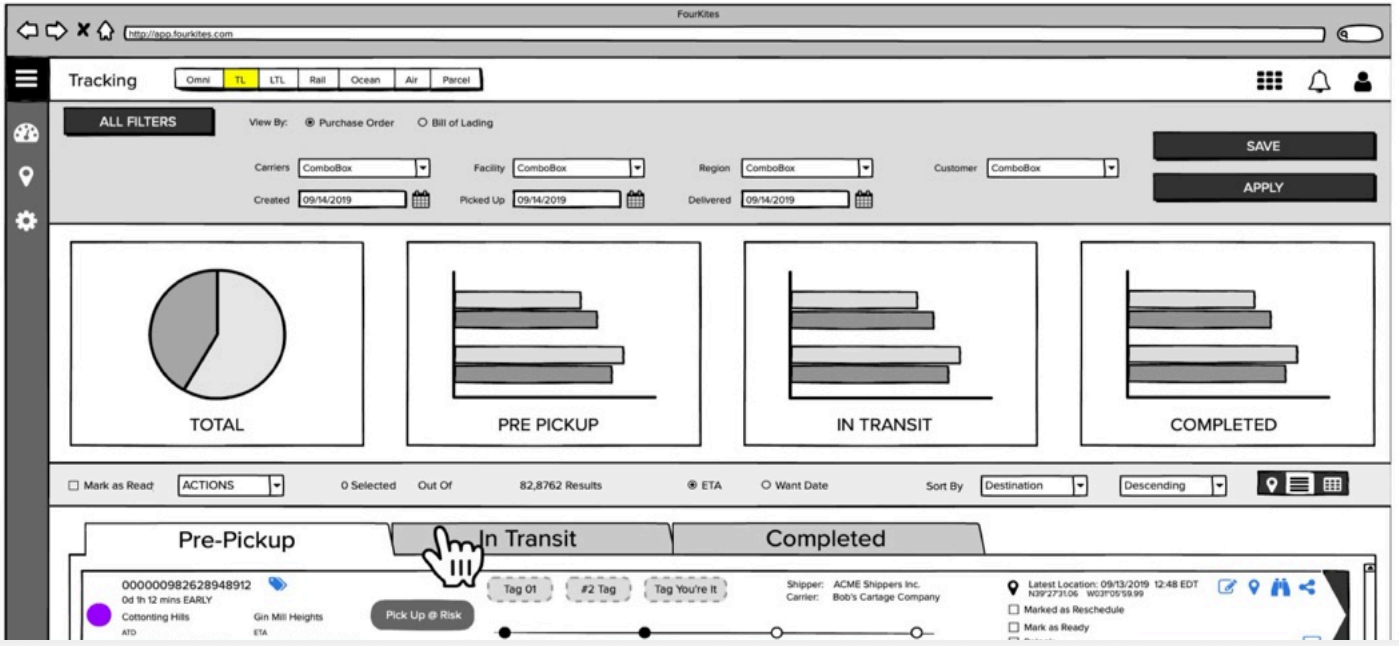
Predictive models added uncertainty
When carrier data was missing, FourKites used historical patterns and real-time signals to predict ETAs. These predictions were often accurate but sometimes wrong. Users couldn't tell prediction from fact.

Transshipments broke the journey
Containers often moved between vessels at intermediate ports. These handoffs created data gaps where neither origin nor destination carrier had visibility. Users lost track of their containers entirely during transfers

APPROACH

1. Mapping the Data Ecosystem (cont.)

Delays cascaded unpredictably
Port congestion in Long Beach affected estimated arrival in Chicago. Missed connections in Singapore rippled through the entire journey. The system knew about these issues but didn't connect them to specific containers.



Key insight:
The data ecosystem was inherently messy. Rather than trying to fix data quality (impossible given external dependencies), we needed to design around uncertainty. Show users what we know, how well we know it, and what we're guessing.

APPROACH

2. Listening to Users
Through interviews with logistics teams and support staff, I focused the design around the top user questions:

Where is my container right now?
Not "where was it yesterday" or "where should it be theoretically." Actual current status with timestamp.

What changed?
ETAs shift constantly. Users needed to know what changed, when, and why. Was this a real delay or just a data

Is it delayed?
Compared to what? Original schedule? Latest ETA? Customer's internal deadline?

How reliable is this update?
Is this verified carrier data or an algorithm guess? Hours old or real-time?

These questions became functional requirements for the new experience.

User archetypes:

The Fire Fighter (Customer Service)
- Gets alerted when containers are delayed. Needs to quickly assess impact and communicate with customers.
- Values speed and clarity over comprehensive detail.

The Planner (Supply Chain Manager)
- Monitoring dozens of containers simultaneously.
- Needs to spot problems before they escalate. Values overview
- and exception highlighting.

The Investigator (Logistics Analyst)
- Digging into specific container issues.
- Needs complete history, data sources, and explanation of system behavior.
- Values transparency and detail.

The interface had to serve all three without overwhelming any of them.

APPROACH

3. Creating a Unified Container Journey

The centerpiece of the redesign was a single journey view that provided:

A clean milestone timeline
Every container moves through predictable stages: booked, loaded, departed port, in transit, arrived port, unloaded, delivered. I designed a horizontal timeline showing these milestones with status indicators (completed, in progress, predicted, delayed).

Map visualization when positional data was available
For vessels with AIS (Automatic Identification System) tracking, show real-time position on map. For containers without position data, show last known location and route. Make it clear when position is live versus estimated.

Carrier versus FourKites-predicted ETAs
Show both side by side. If they differ significantly, flag it. This transparency helped users understand discrepancies and trust our predictions when carrier data was stale.

Port stop details and transshipment events
Each port interaction expanded to show arrival/departure times, dwell time, handling events, and connecting vessels. Transshipments, which previously disappeared into data black holes, now had explicit visibility.

APPROACH

3. Creating a Unified Container Journey (Cont.)

Design decisions:

Linear timeline over complex network diagram
Containers move linearly through time, even if routing is complex. Timeline view was more scannable than trying to show all possible paths.

Progressive disclosure
Default view showed high-level journey. Each milestone expanded for detail. This served both the Fire Fighter (quick scan) and the Investigator (deep dive).

Consistent visual language for status
Green (on track), yellow (watch), red (delayed), gray (predicted/unverified). Used across timeline, map, and alerts.

APPROACH

4. Introducing Data Confidence

To restore trust, I designed a data confidence model that visually differentiates:

Verified carrier events (high confidence)
Marked with solid indicators and carrier logo. These are facts reported directly by the carrier or port.

Predicted events (medium confidence)
Marked with dashed indicators and "predicted" label. Based on historical patterns, current conditions, and real-time signals. Show confidence percentage when available.

User-submitted or secondary-source data (variable confidence)
Marked with distinct styling and source attribution. Useful but not authoritative.

Key design principle:
Never hide uncertainty. If we're guessing, say so. If data is hours old, show timestamp. If sources conflict, show both and explain.

This transparency did something counterintuitive. By admitting what we didn't know, users trusted what we did know. The system became more credible by being honest about its limitations.

APPROACH

5. Exceptions Made Actionable

Delays, reroutes, and ETA changes are elevated into contextual alerts that guide the user instead of interrupting them.

Before: Exception table buried in sidebar, sorted chronologically, no context
After: Exceptions surfaced in three places:

1. Journey timeline
Delays and changes marked directly on affected milestones.
Hover or tap for explanation and recommended action.

2. Alert panel (when needed)
Critical issues that require attention appear in dedicated panel with clear severity (urgent, warning, info) and suggested next steps.

3. Dashboard rollup
For users monitoring multiple containers, exception summary with filtering and bulk actions.

APPROACH

5. Exceptions Made Actionable (Cont.)

Alert design principles:

Prioritize by impact, not recency
A 3-day delay on critical shipment outranks a 2-hour delay on routine delivery.

Provide context, not just facts
Don't just say "ETA changed." Say "ETA delayed 2 days due to port congestion in Shanghai. 3 customer orders affected."

Suggest actions
"Notify customer," "Reroute via air," "Request expedited customs." Give users a path forward, not just bad

Reduce noise
Minor ETA adjustments (under 6 hours) don't trigger alerts unless shipment is already critical. Users told us they wanted fewer, more meaningful notifications.

Global Tracking

FourKites

SOLUTION

SOLUTION

A redesigned Global Container Tracking view that:

Aligns all container events into a clear, sequential journey
Timeline view shows every milestone from booking to delivery. No more hunting through tables or reconciling conflicting views.

Integrates source transparency and confidence scoring directly into the UI
Users immediately see verified versus predicted data. Timestamps and source attribution build trust.

Surfaces meaningful exceptions
Delays, missed milestones, port congestion appear where users need them, with context and guidance. No more buried alerts or surprise disruptions.

Reduces cognitive load through consistent patterns and language
Same visual language across timeline, map, alerts, and dashboard. Learn once, apply everywhere.

Works seamlessly across carriers, ports, regions, and data sources
Whether tracking a Maersk container from Shanghai to Los Angeles or a smaller carrier from Rotterdam to New York, the experience is consistent. The system adapts to data availability without breaking the mental model.

SOLUTION

Reduced support escalations
Users could diagnose issues themselves, decreasing dependency on Customer Support. Support ticket volume for maritime shipments dropped 41% in first quarter after launch.

Higher trust in maritime data
Confidence indicators and transparent sourcing eliminated ambiguity. Customer satisfaction scores for maritime visibility increased from 6.2 to 8.4 (out of 10).

Increased adoption
Teams that previously relied on manual tracking shifted workflows to FourKites. Daily active users for ocean tracking increased 63%. Time spent in platform doubled (positive indicator, users finding value).

Foundation for predictive visibility
The new architecture supported future features like predictive ETAs and automated exception workflows. Data confidence model enabled machine learning improvements (system could learn from which predictions were accurate).

Stronger market differentiation
The clarity and transparency of this UI became a competitive advantage in the ocean visibility space. Sales team reported design as key differentiator in competitive evaluations.

REFLECTION

What Worked

Designing around uncertainty instead of hiding it
Admitting what we didn't know made users trust what we did know. Transparency built credibility.

User questions as functional requirements
"Where is my container? Is it delayed? What changed? How reliable is this?" These questions drove every design decision and kept us focused on actual user needs.

Progressive disclosure for multiple user types
Timeline view served quick scanners. Expandable details served investigators. Same interface, different depths.

Cross-functional partnership
Working closely with data science and carrier integration teams meant designs were grounded in technical
reality. We didn't design impossible features or ignore real constraints.

REFLECTION

What I'd Do Differently

Test with smaller shippers earlier
Initial research focused on enterprise customers. Smaller shippers had different needs (fewer containers but each
more critical). Earlier testing would have surfaced their requirements sooner.

More aggressive exception filtering
First version showed too many alerts. Users got alert fatigue. We tuned this post-launch but should have been
more ruthless in initial prioritization.

Better mobile experience
Designed desktop-first. Mobile view worked but felt cramped. Given that many logistics coordinators work from phones, should have designed mobile-first.

Documentation for support team
New confidence indicators and terminology created training needs for support. Better documentation andtraining materials upfront would have smoothed transition.

REFLECTION

What This Taught Me About Complex Data Products

Transparency beats perfection
Users preferred honest, imperfect data over polished, unreliable data. Show your work.

Exceptions are the product
In tracking, everything working as planned needs minimal interface. The product is really about surfacing and resolving problems.

Consistency across variable quality is hard
Designing one interface for wildly different data quality (real-time APIs versus daily email updates) required explicit confidence communication, not just graceful degradation.

Trust is infrastructure
For enterprise platforms, trust isn't a nice-to-have. It's foundational. Users need to believe the system before they'll rely on it for critical decisions.

CAPABILITIES DEMONSTRATED

1. **Complex data product design:**
   Designing interfaces for uncertain, multi-source, real-time data

2. **Trust and transparency:**
   Using design to build credibility in imperfect systems

3. **Enterprise UX:**
   Serving multiple user types with different needs in single interface

4. **Cross-functional collaboration:**
   Working with data science, engineering, and domain experts

5. **Exception design:**
   Surfacing problems at the right time with the right context

6. **Progressive disclosure:**
   Balancing overview and detail for different use cases

7. **Visual information design:**
   Making complex journeys scannable and comprehensible