Fabrice Dubois

Digital Product Design

uxriver

Service Tag

App for LED lighting technicians in the field.
A digital tool to help with the installation and maintenance of street or office lights, reducing service disruption and factory returns.

When

2018 to 2024

My role

- Design lead
- Interaction design
- Information architecture
- Visual design contribution

Learnings

- Setting a way of working with a remote team
- Tech challenges (eg. NFC, GPS)
- Obstacles to app adoption
- Responsive design

DESIGN AWARD 2023

Signify is the new name of PHILIPS LIGHTING

Service Tag (1/4)

A pragmatic approach

As a general principle the app encourages repair and reuse, versus costly factory returns and component waste.

It's particularly useful for luminaires lacking Internet access and thus the capability to be remotely handled. Though cities around the world have accelerated their upgrade to connected technology, many older generation lights are still in service and need on-site maintenance.

After identifying the luminaire by scanning its QR code, the user gets a wealth of information and tools to help them execute tasks at all stages of the lifecycle, such as:

- Installing
- Late stage tuning
- Troubleshooting
- Planning a repair
- Replacing a broken part

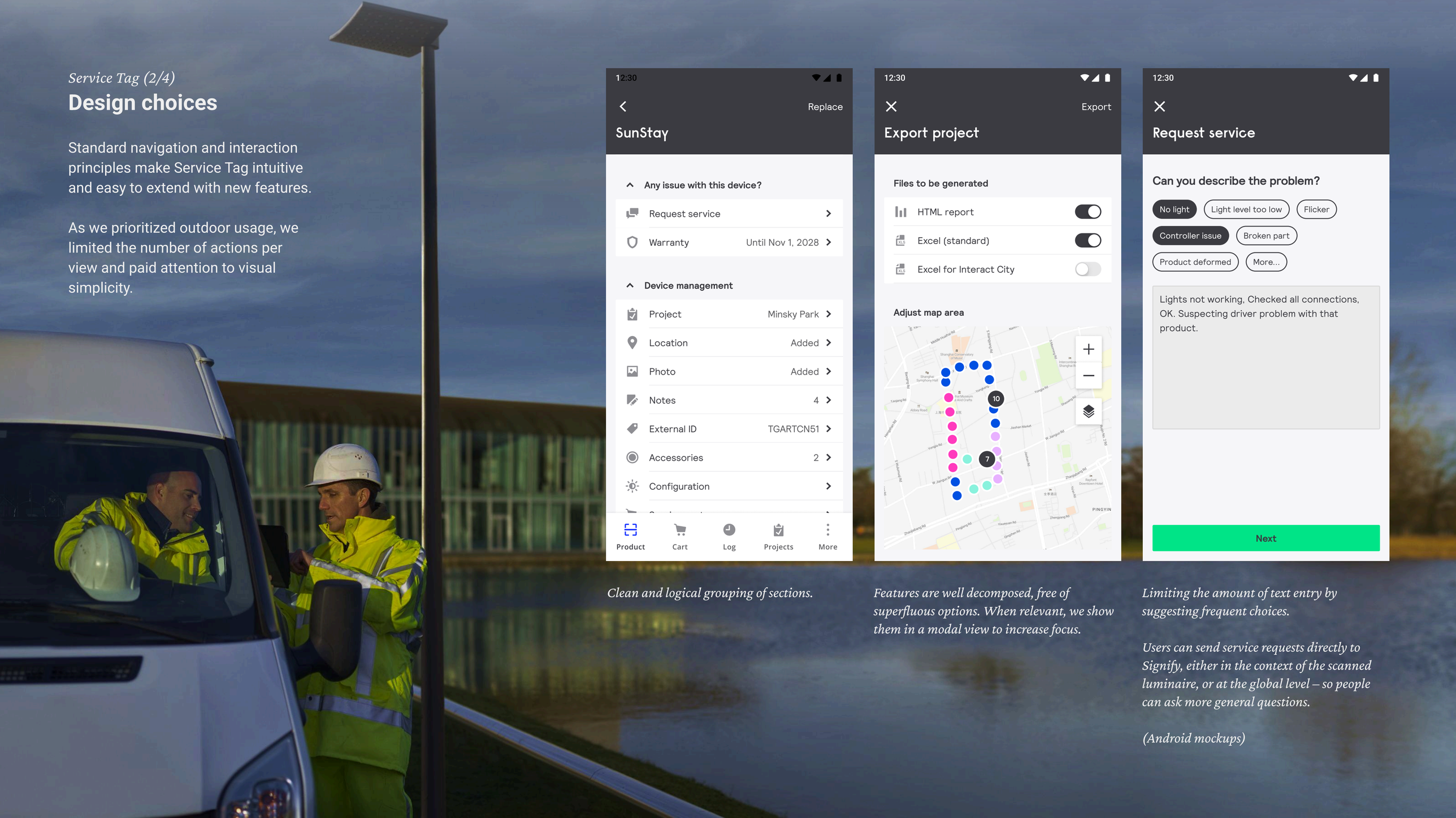
Because scanning is the starting point of most workflows, we made it very easy to find (landing view) and added alternative ways for when it can’t work.

Service Tag (2/4)

Design choices

Standard navigation and interaction principles make Service Tag intuitive and easy to extend with new features.

As we prioritized outdoor usage, we limited the number of actions per view and paid attention to visual simplicity.



Clean and logical grouping of sections.

Features are well decomposed, free of superfluous options. When relevant, we show them in a modal view to increase focus.

Limiting the amount of text entry by suggesting frequent choices.

Users can send service requests directly to Signify, either in the context of the scanned luminaire, or at the global level – so people can ask more general questions.

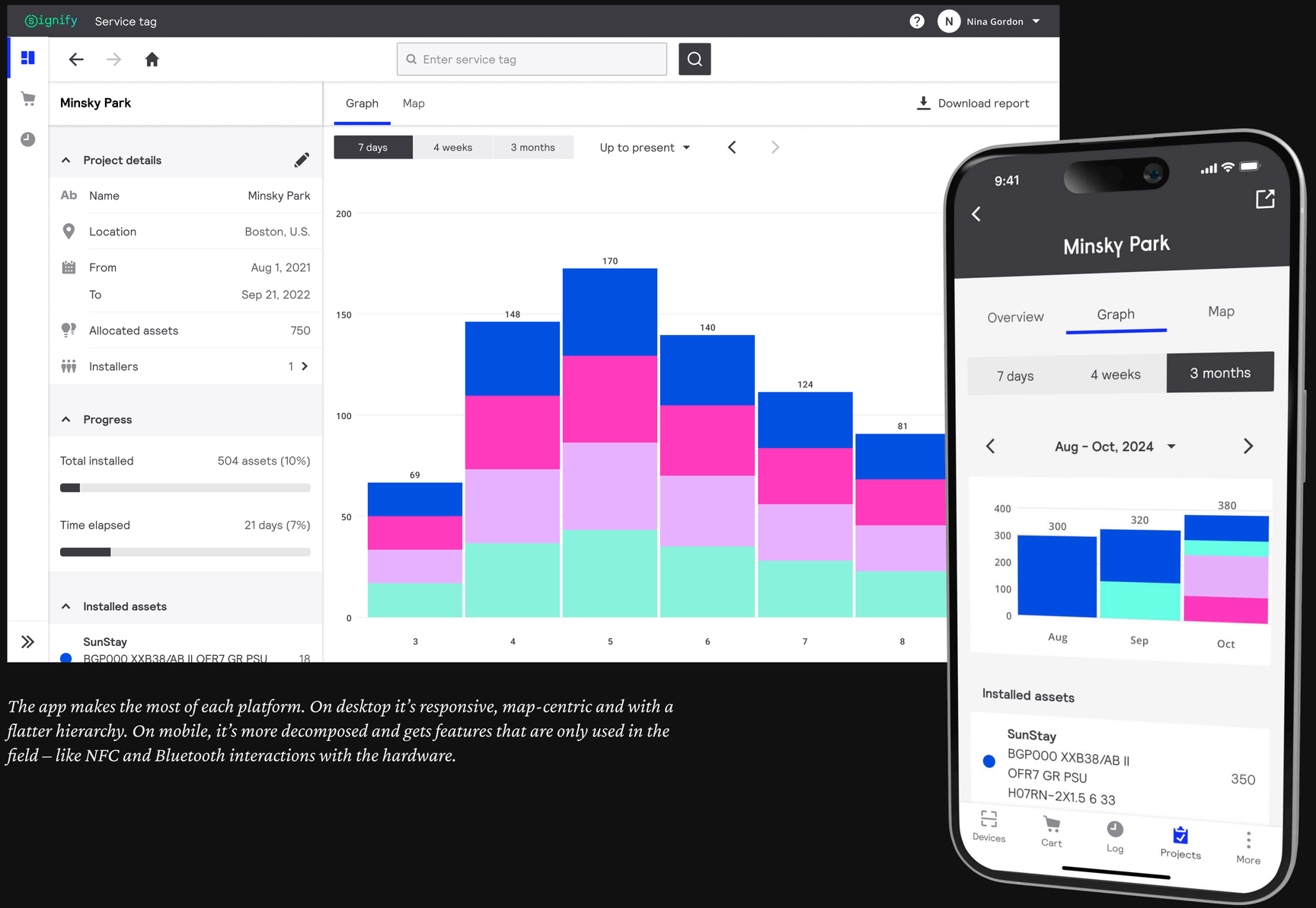
(Android mockups)

Service Tag (3/4)

Mobile and web

The Android and iOS apps have much in common, being both based on the same design system, Signify Photon. A few UI patterns remain native, such as certain navigation and menu conventions. Branding matters, but it's more important that the app feels at home on the device of the user.

I later worked on a web adaptation for a new category of users. Office-based managers, they wanted a better integration with their desktop workflow and a more comfortable way to visualize their installation team's progress.



The app makes the most of each platform. On desktop it's responsive, map-centric and with a flatter hierarchy. On mobile, it's more decomposed and gets features that are only used in the field – like NFC and Bluetooth interactions with the hardware.

Service Tag (4/4)

How we worked

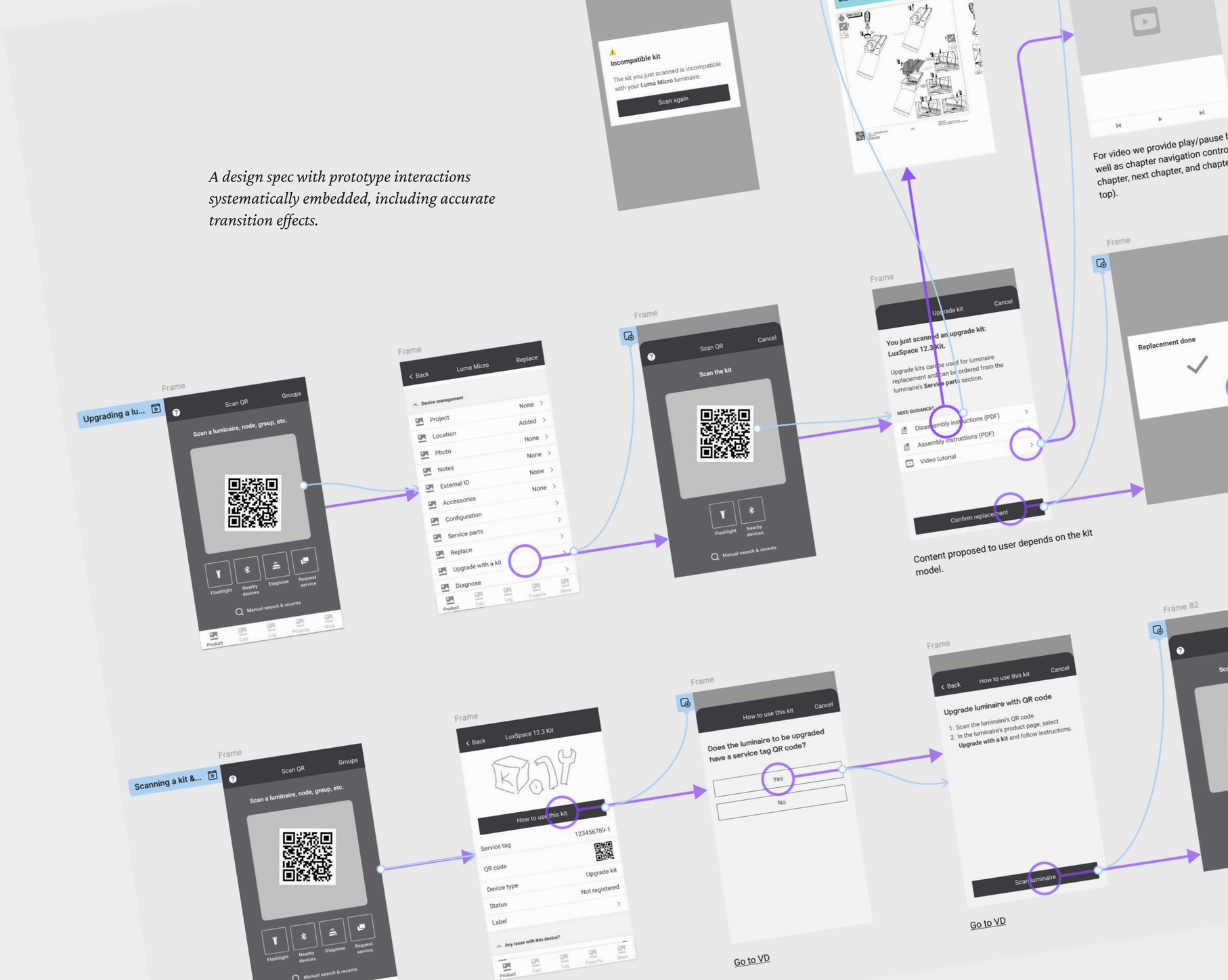
Communication with the PM and developers (all based in India) was very pragmatic, result-oriented. They would patiently brief me and explain their concerns, and in return, I strived to deliver clear designs that I commented in a weekly call.

The PM particularly valued the prototypes I kept at their disposal for customer meetings, as they helped them discuss the need. In my design process, I wire interactive prototypes as I go along, rather than as an afterthought.

Prototyping allows me to:

- Preview the user experience while I create the interface
- Quickly fix any major issues
- Show devs the subtle transition effects I envision to keep users oriented between views
- Accelerate stakeholders alignment
- Conduct lean user testing

A design spec with prototype interactions systematically embedded, including accurate transition effects.



Design principle:

Orthogonality

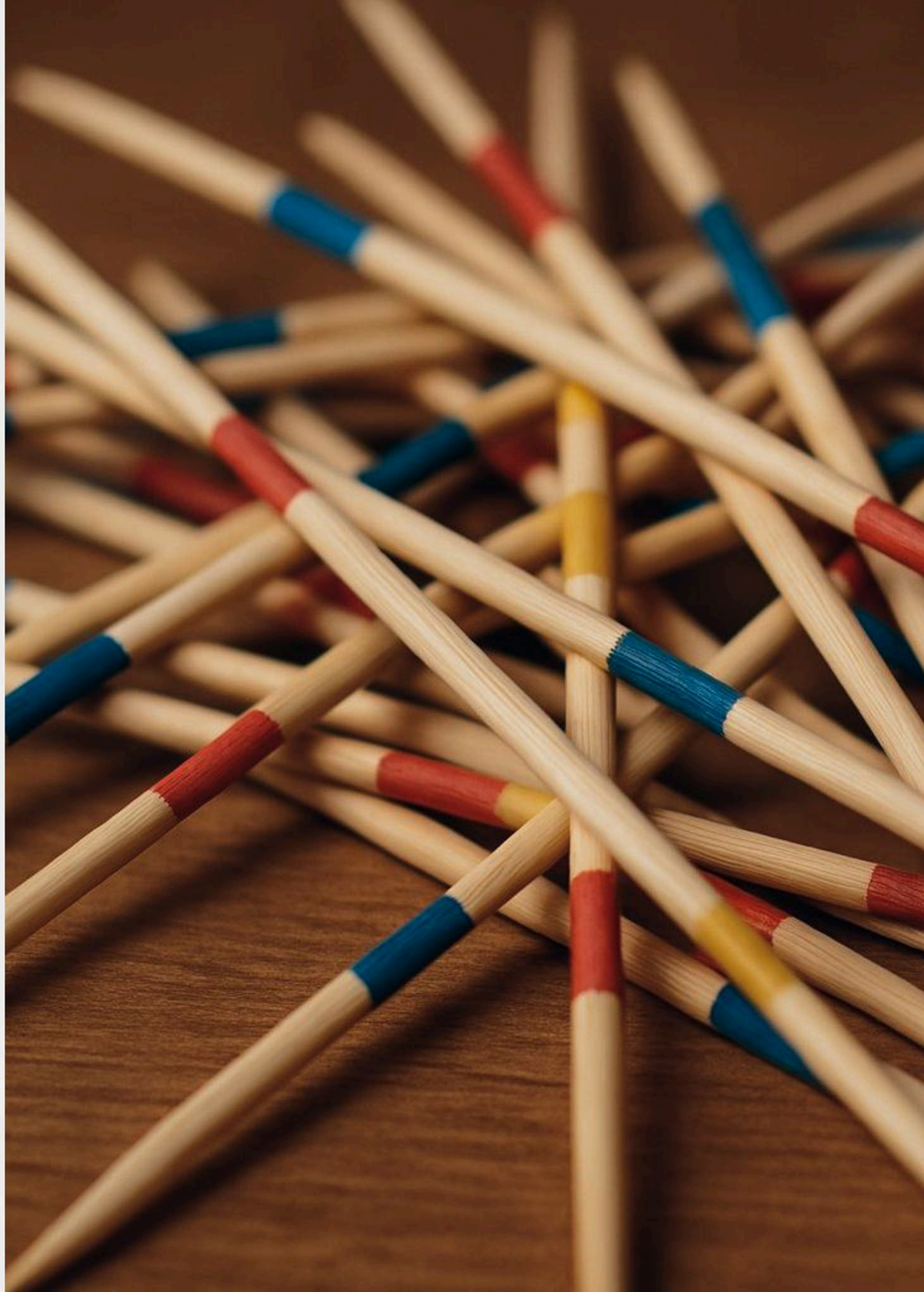
Orthogonality is about ensuring that the parts of a system don't depend on each other without a good reason.

In a well designed system, the parts tend to be orthogonal to each other. It means you can easily handle one part without worrying about the others, as each part is well-separated and understandable on its own.

The Mikado game, pictured here, is a metaphor for very poor orthogonality: it's hard to touch a stick without impacting the others. Even *looking* at an individual stick is tricky. Everyone finds products designed like that difficult to handle mentally. Hard to explain, hard to understand, painful to work with and work on.

Designers sometimes get briefs that look like a bowl of sticky noodles, a bit like the Mikado. Their first job is to disentangle the noodles, to clarify actual dependencies, eliminate unnecessary ones, restate definitions if needed. In general, prefer a long list of short requirements to a short list of long ones.

"Don't link what's independent."
– Fred Brooks



Interact City

Advanced web app for municipalities to remotely control and monitor streetlights. Various workflows for different user roles, from light level scheduling to energy monitoring, to fault investigation.

When

2018 to 2024

My role

- Design lead
- Interaction design
- User interviews
- Usability testing

Learnings

- Team planning & agility
- Application architecture
- Design systems
- Web app design
- Complexity management

DESIGN AWARD 2021

Signify interact

Interact is Signify's portfolio of professional connected lighting applications

Interact City (1/4)

Evolving a complex product

Over the years, Signify (previously known as Philips Lighting) has offered various city lighting solutions, each with unique software, hardware, and advantages.

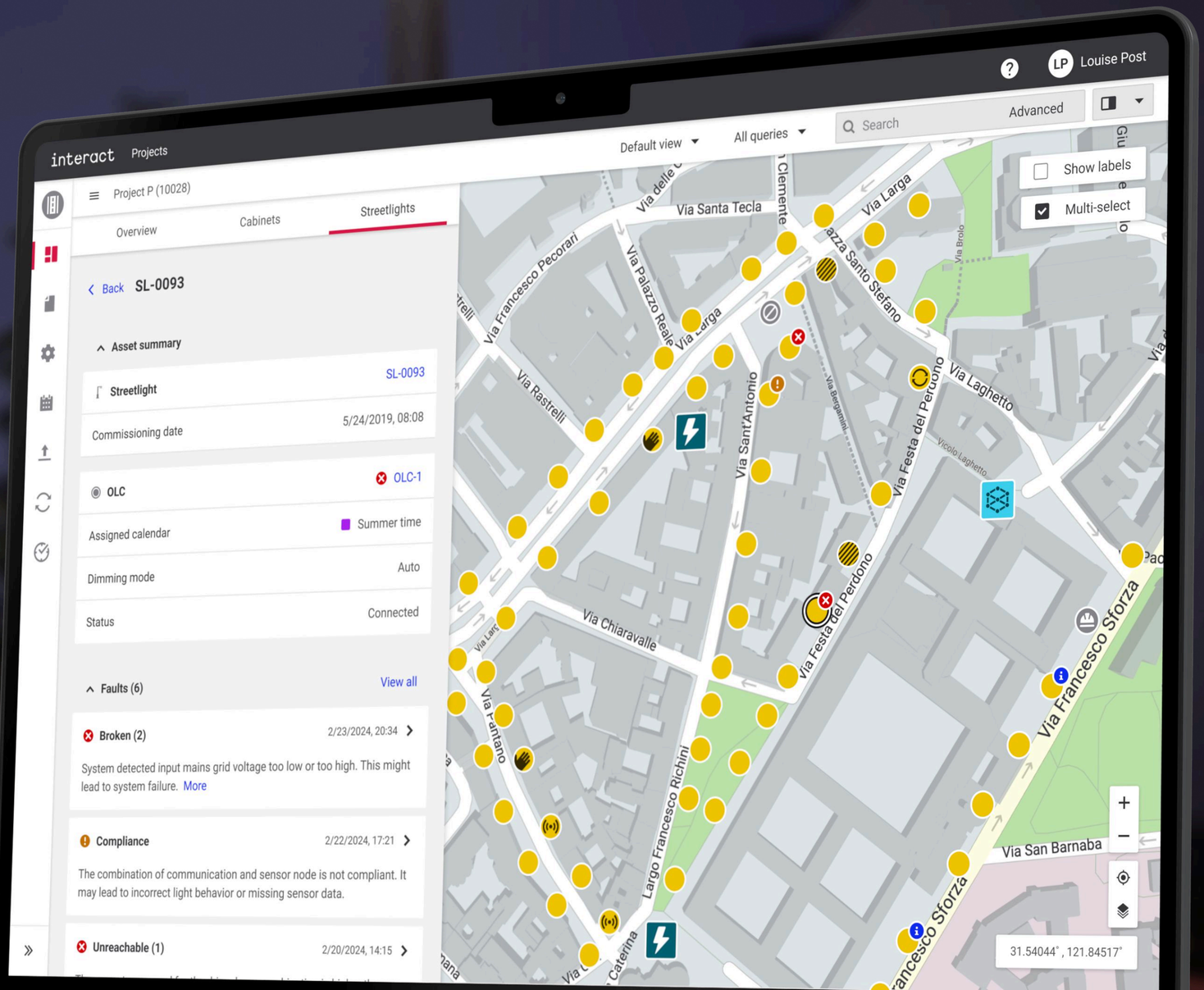
With Interact City, our goal was to create a unified and modern interface that could integrate all these systems gracefully. This required us to understand each system well, recognizing common elements while keeping their unique features.

Aligning things could be challenging. For instance, changing workflows too much could confuse users accustomed to the older products.

On top of that we gradually added features and support for new technologies, like sensors, solar-powered lights, or simplified commissioning.

In short, we needed to keep the design adaptable but complexity manageable.

Style wise, we learnt how to apply the Interact design system consistently and how to deviate thoughtfully.



People want to see a lot of data on the map, but it must remain intuitive. Displaying numerous objects, all with different types and states, required an icon system that prioritizes key information while letting users access details on demand.

Interact City (2/4)

Approaching deeper changes

Case study

A simplified example of process that we followed when working on a significant revamp.

At some point in the project I saw a need to rethink part of the UI. More an more issues reported by users were hard to address individually because they pointed to deeper limitations. Not able to answer those requests simply, I explained it was an architecture problem that would need to be addressed globally, not by accumulating local fixes.

One day I decided to make time for a proper study, to go to the bottom of the problem and come up with a strategy. Vague ideas were in the air but implications for users and on other features had to be understood.

Understanding →

Most complaints seemed caused directly or indirectly by the same design issues:

Unclear navigation logic. Sometimes hard for people to build a mental model and predict UI reactions.

No uniform way to handle objects. Mostly designed for streetlights and cabinets, the existing UI is too rigid to adapt to new types of objects.

First, we needed more insights...

Interviewing →

6 or 7 user interviews, open format as I wanted a discovery approach.
My goals:

- Understand how people think
- See what really matters to them
- Validate assumptions (or not)
- Clarify what I try to solve

Users seemed to work like this:

1. View a certain set of objects (eg. *Lights on Liberty Street*)
2. Apply actions
3. Repeat with a new view (or go back to previous view)

So they needed to quickly jump between arbitrary views. This called for a navigation logic that wouldn't depend on the views themselves. Also, we lacked a clear definition of the concept of view.

Shaping →

Hardest part. Turning intuitions into a set of key ideas, and ensuring they work, especially *when put together*.

- What do I introduce or change?
- What do I lose, how do I mitigate?
- What new questions arise?

Key idea examples:

A more intuitive way for users to access their favorite views, inspired by browser bookmarks.

A simpler view navigation, also inspired by browsers. The mechanism is orthogonal to views themselves.

A more flexible data visualization feature that user can toggle as a map option, independently of navigation.

Etc.

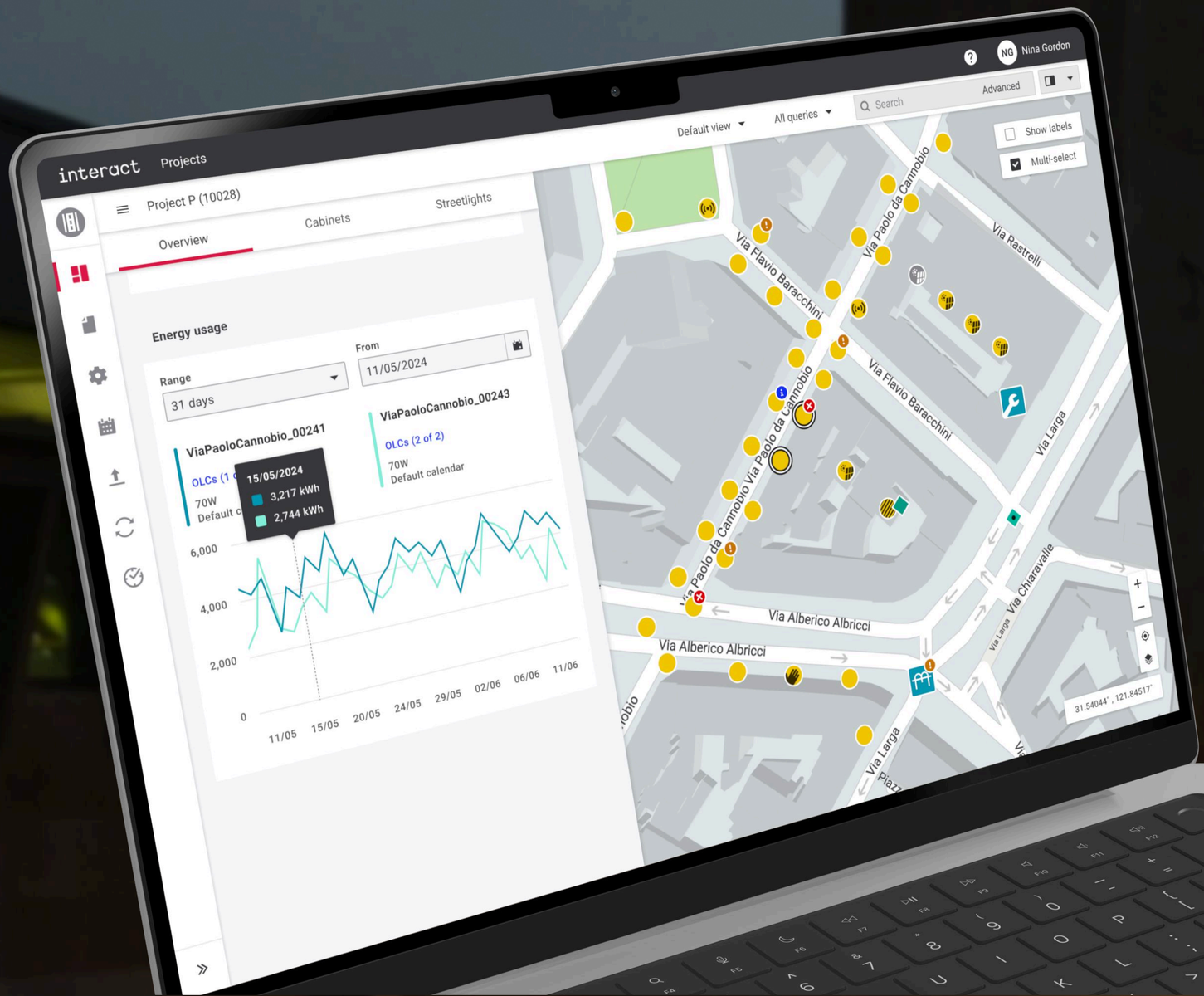
Testing, Planning, Detailed design

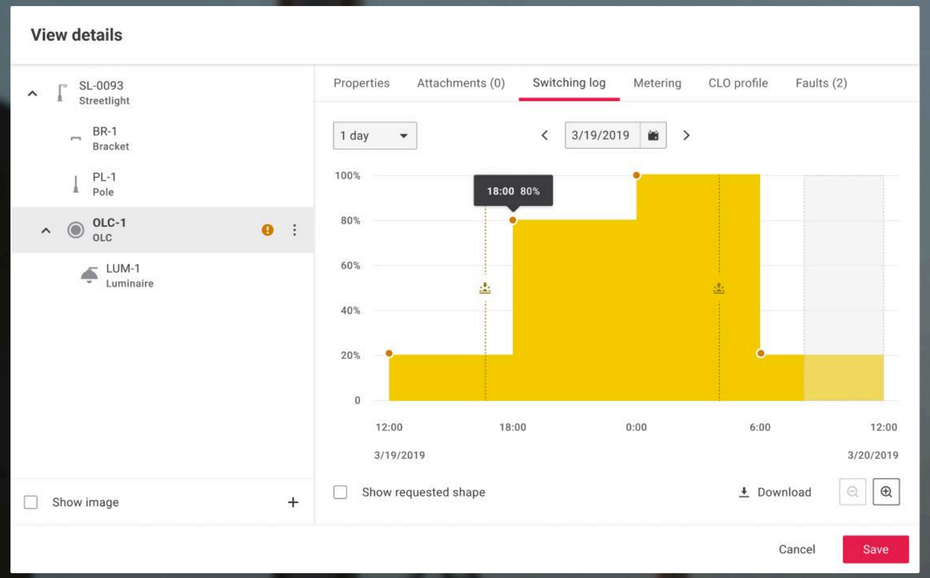
With low-fi prototypes to test each key idea, we met again with our interviewees. Integrating their feedback took a few rounds but the direction appeared solid.

Time to share more... Demo to local PM, then to the whole project team.

I then created a dependency graph to identify what we had to do first, which helped deliver changes incrementally and confidently.

Detailed design could start at last...





The View details UI. *Initially designed to inspect streetlight properties, this evolved as a more general helper tool that proved relevant to many workflows. It combines tree and tab views to organize function and information efficiently.*

The quick comparison tool. *When they troubleshoot a light, engineers like to start by checking its energy usage against another light. They can do this in just two clicks on the map. A usability improvement that came directly from chatting with a user.*

Interact City (4/4)

How we worked

Our main project contacts were PMs, software architects and customer-facing engineers. We were up to 3 contributing designers, plus a technical writer, while system experts (who are daily users of the product) informed our work with regular inputs and feedback.

We used the SAFe agile method. Every quarter I scheduled design work for the design team so as to stay ahead of R&D and make time for solid solutions. Over the years, I delivered most of the design evolutions – though the initial version came from an agency and we built largely on Signify's design system.

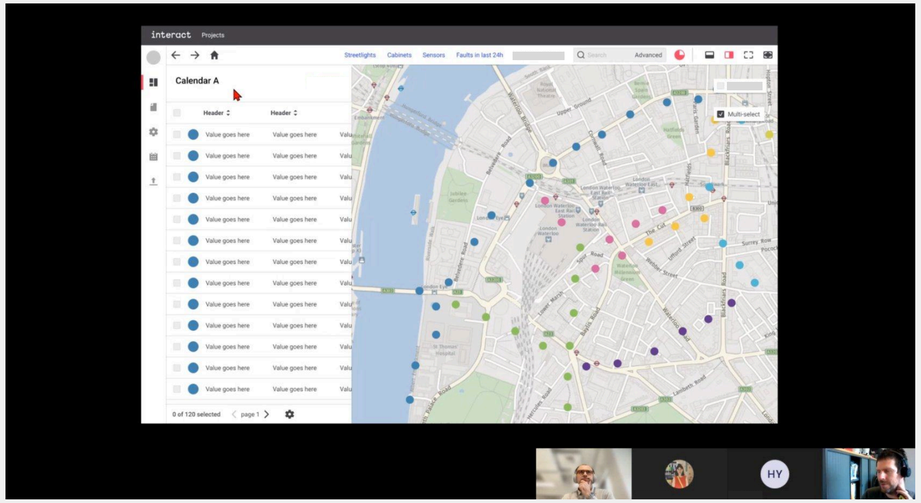
I encouraged the team to join global meetings such as the Change Control Board (CCB), to connect with project members in diverse roles. Good for socializing, motivation, and demonstrating that designers don't mind getting their hands dirty when needed.

Meeting in person. *Here with experts in Bangalore, boosting collaboration and trust. Hearing diverse voices helped understand problems faster and generate fresh ideas.*

To save everyone's time, prep work is needed – but it doesn't have to be complicated. A UX colleague once said to me: "Stay pragmatic. It's all about getting answers, in the end."

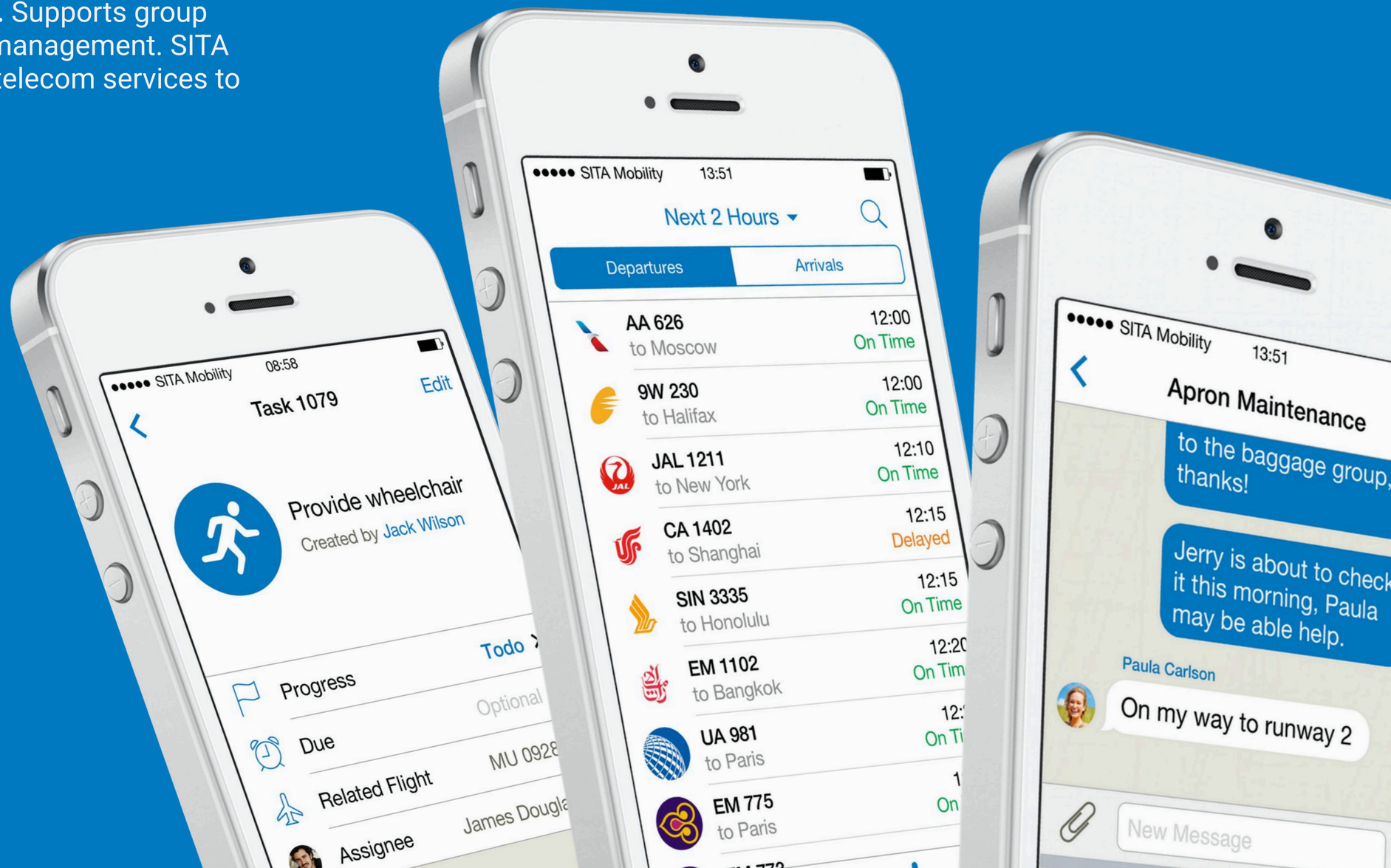
Remote user testing. *A user from the Netherlands evaluates a prototyped idea. With people's permission, I record these sessions to catch important details that might be missed during the live call, and to allow me to focus on the respondent without worrying about taking notes. I also like to invite other designers as observers.*

Experience taught me that watching people interact with a product reveals more than their words ever could. People often don't do what they think *they do.*



SITA Airside

Mobile tool suite for airport staff. Supports group collaboration and digitized form management. SITA is a global IT company providing telecom services to the air transport industry.



When

2015 to 2016

My role

- Information architecture
- Functional requirements spec
- Interaction + visual design
- Interaction design for admin website
- Design guidelines for SITA iOS apps

Learnings

- A new domain, air transportation, revealed its challenges and need for efficient tools
- Information architecture
- Design for tablet
- iOS expertise

Topics

Digitized forms
I designed a form library for iPad that offered a unified view of all kinds of forms that people routinely use. Agents could check a form history and browse templates relevant to their job.

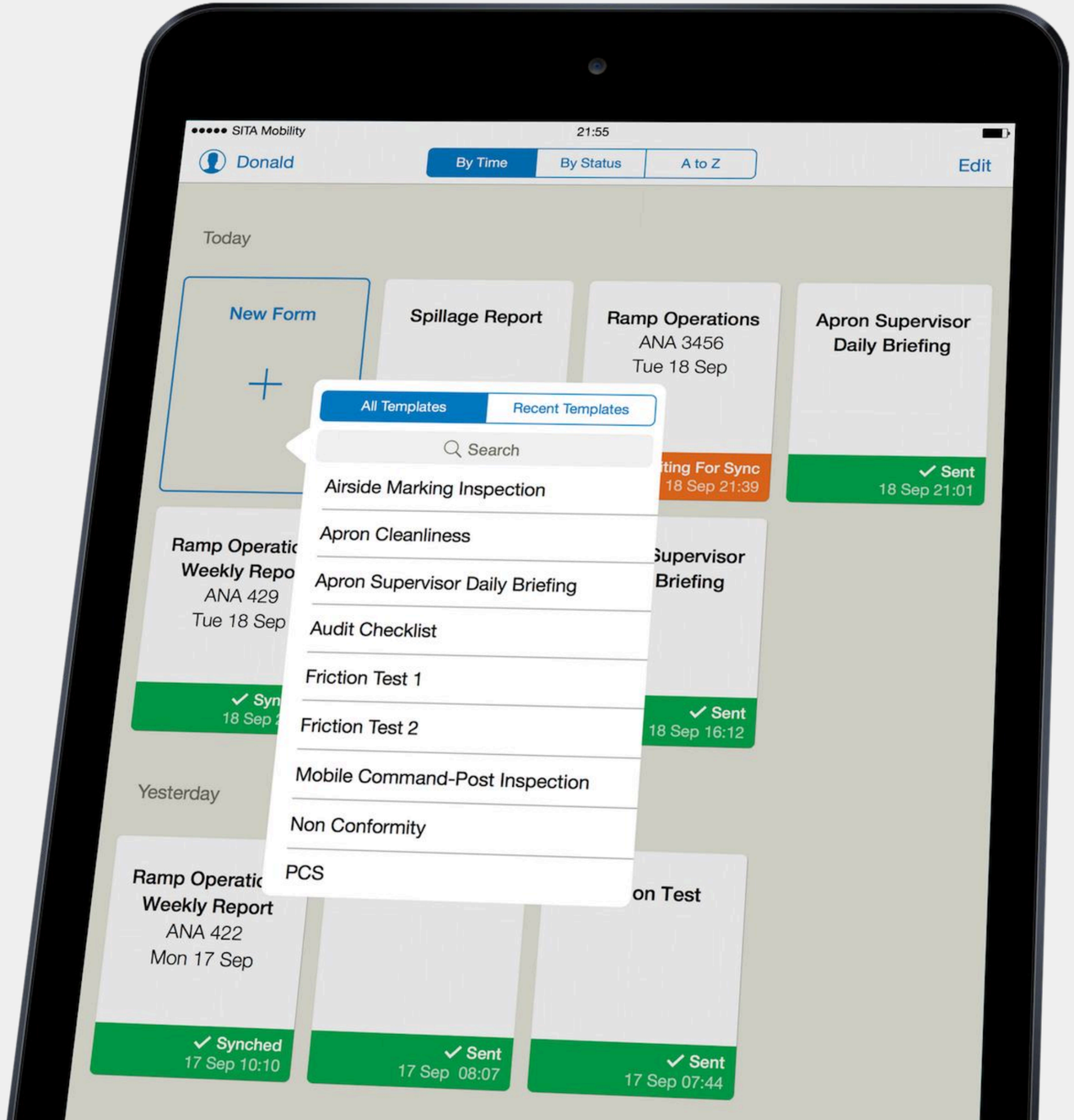
Two types of forms were covered, though they were presented in the same way in the library:

- For simple routines, like reporting on runways' condition, existing HTML templates were reused and I didn't have to design those
- For more complex and interactive procedures, such as turnaround timing and cargo loading, I created ad-hoc user interfaces

Group collaboration
Think WhatsApp, but specialized for airport agents. Designed for iPhone.

One challenge was to design a chat tool that would support both official and user-defined groups, as well as teamwork features such as task assignment and location sharing. Official groups, in particular, required considering admin aspects like who controls them, who can join them, etc.

Prior to any UI work I clarified the requirements, business logic and information architecture, creating a strong foundation and helping the team stay on the same page.



The form library helps people manage their form collection and fill in new forms.

SITA Airside (2/6)

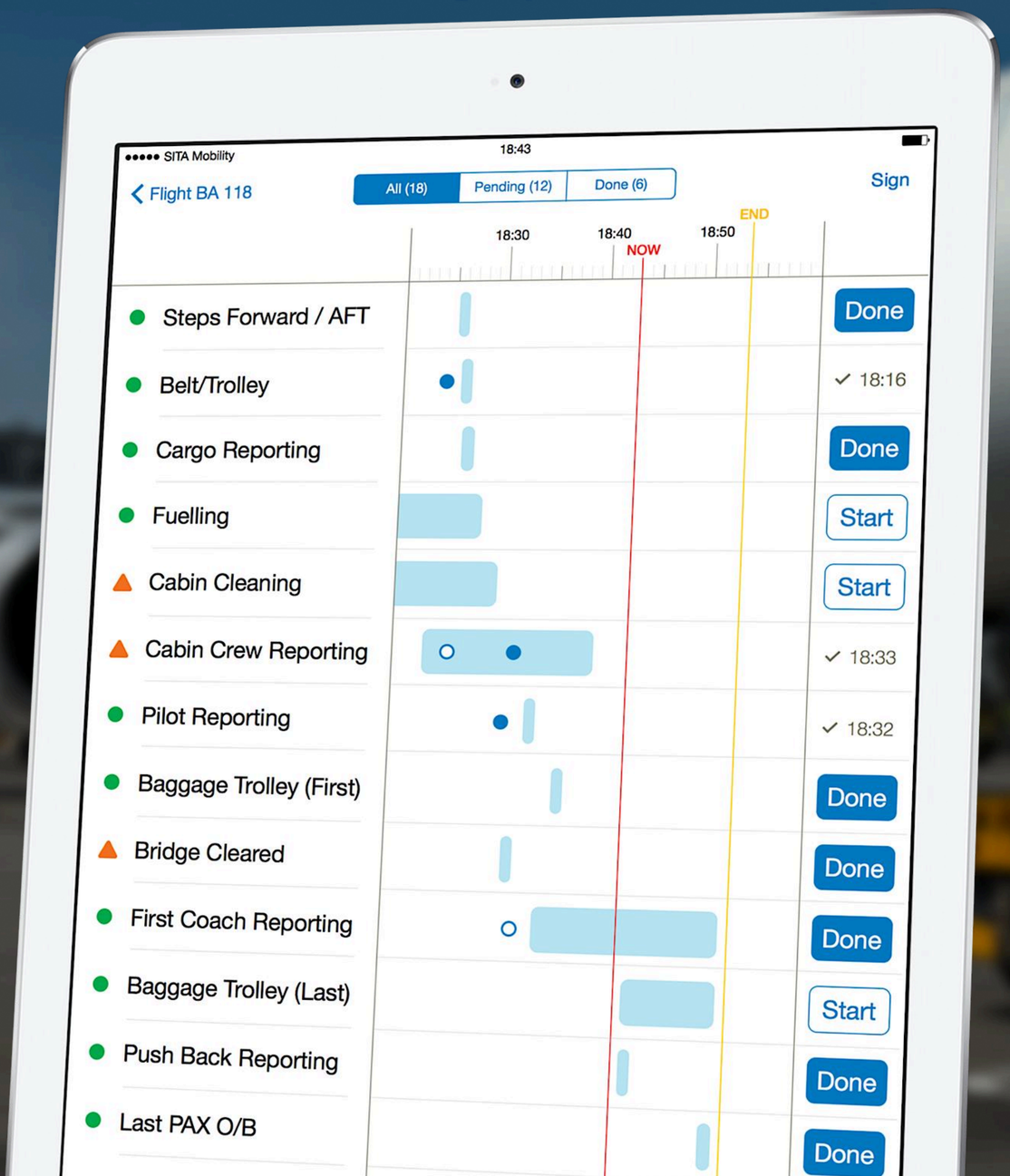
Ad-hoc interfaces

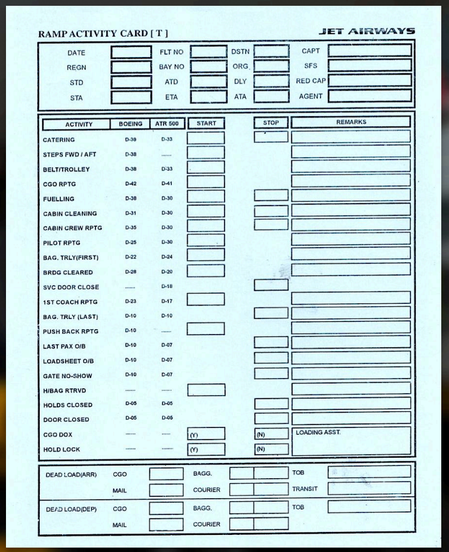
Some forms were too complex to be based on a mere HTML template. They required a bespoke native UI.

For example the **Ramp operations precision timing** form, shown here, which helps a supervisor monitor the time spent by a ground team during an aircraft's turnaround.

Visualizing the overall progress, the supervisor can anticipate delays. This is key because the longer the plane remains parked, the more the airport charges the airline.

Inspired by multi-track music sequencers, the UI provides an overview of how long operations are expected to take, warns if some run late, and can be filtered by operation status. Interaction is kept simple: the user records times by tapping an operation when it starts and tapping again when it's done.

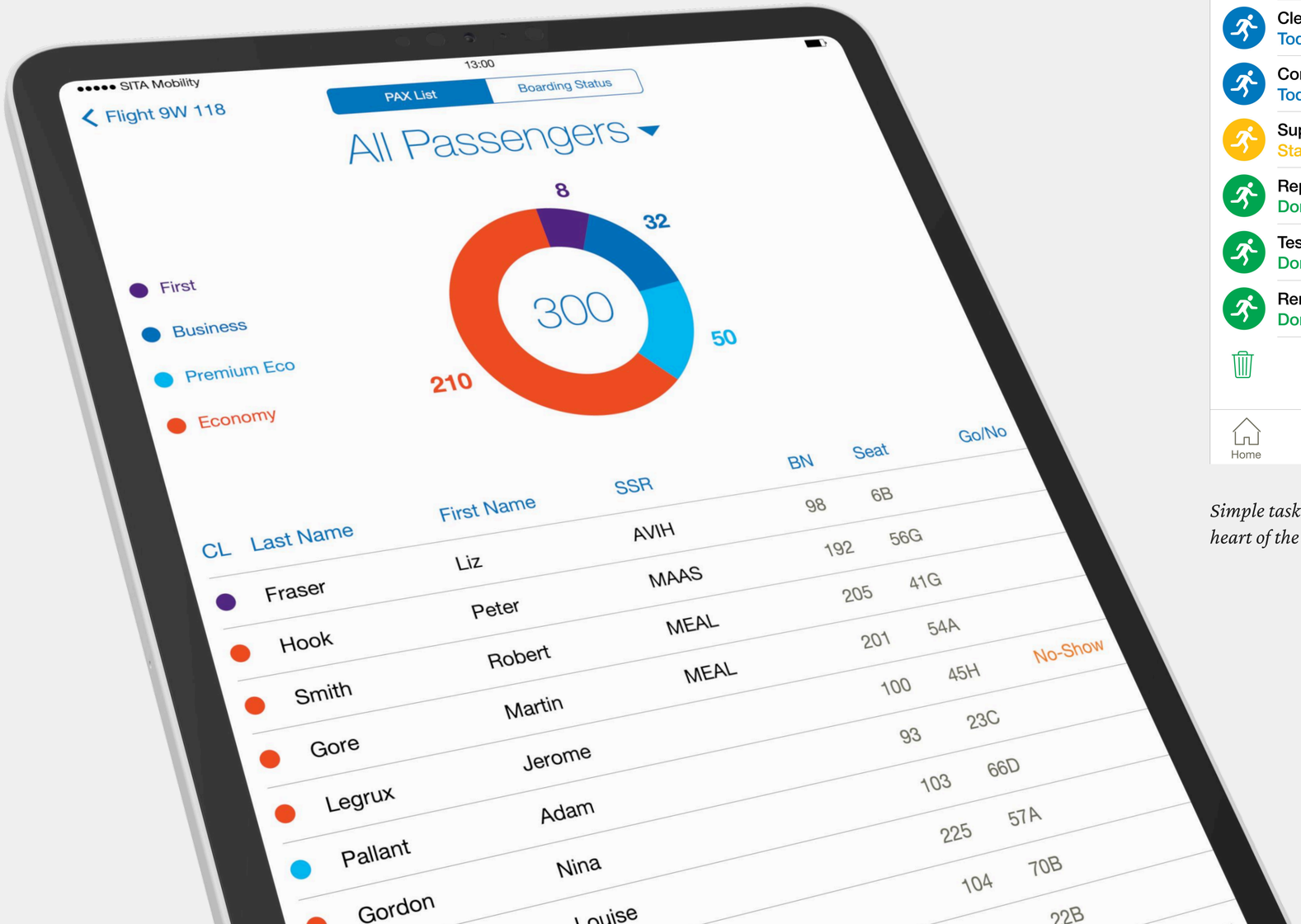


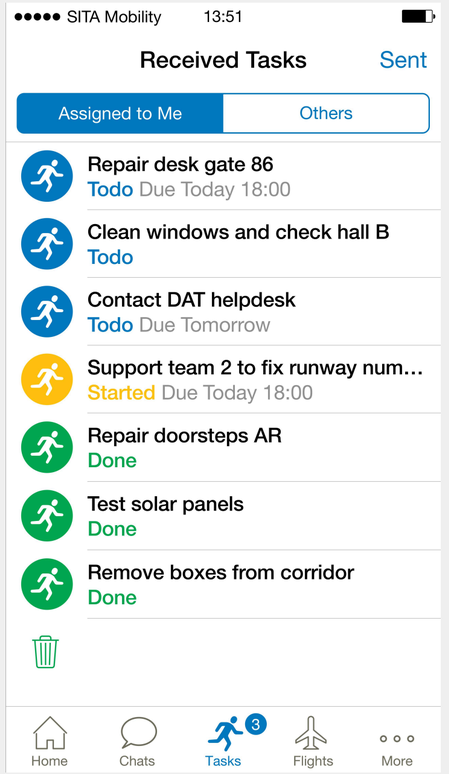


Original paper form example for the turnaround. Error-prone, static and hardly configurable.

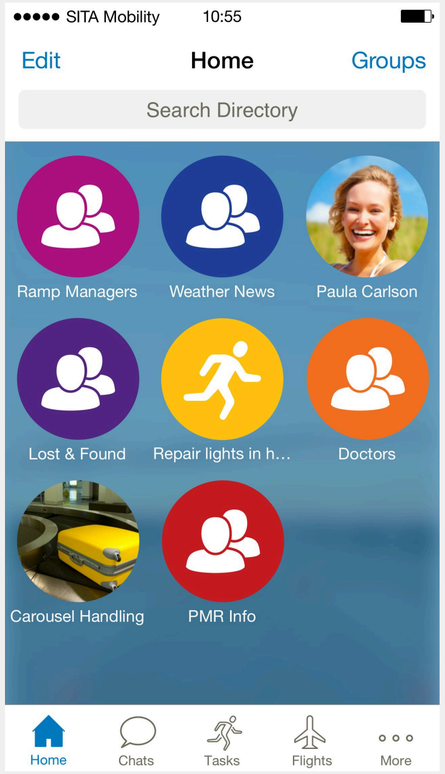
SITA Airside (3/6)

On tablet, the boarding / passenger list was accessible from the flight details view.





Simple task assignment and tracking, at the heart of the collaboration app.

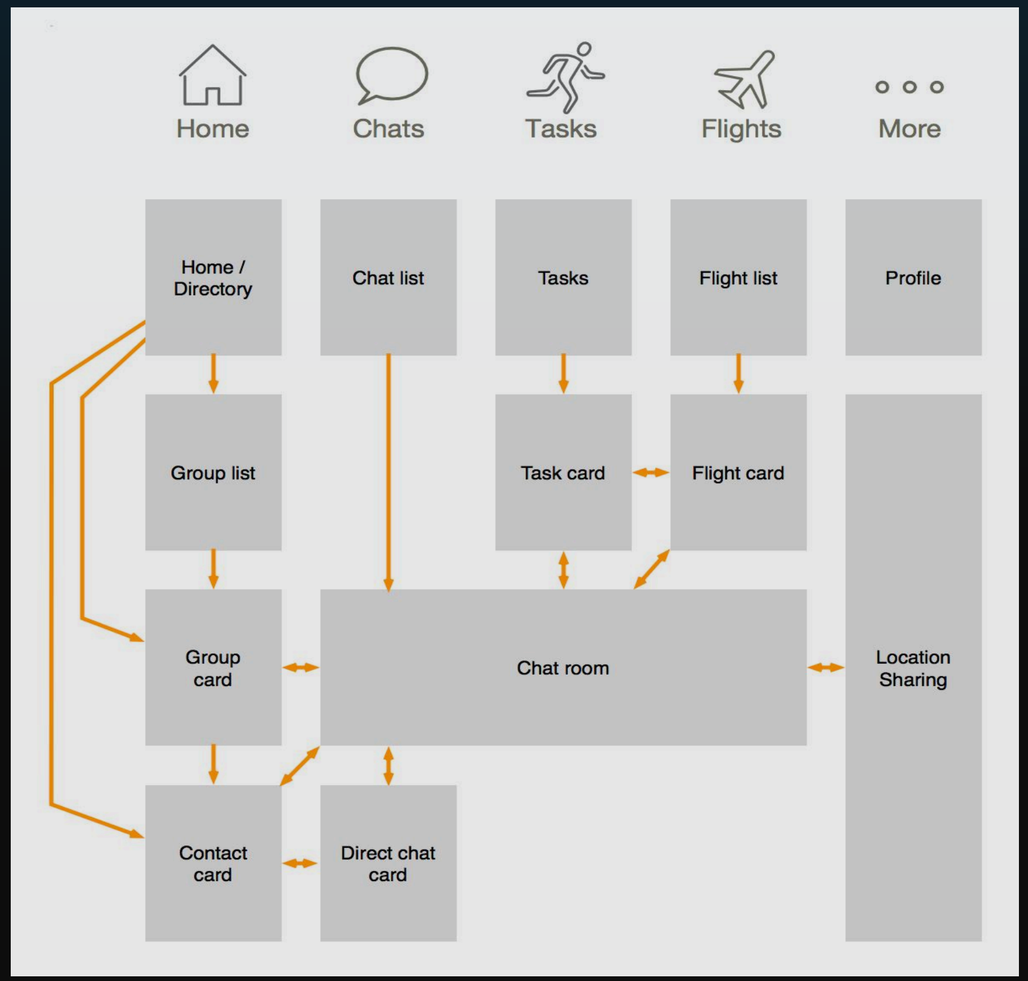


Users personalize their home view with shortcuts to any objects, such as a group, contact, task, or flight.

SITA Airside (4/6)

Architecture of the collaboration app

The information architecture overview shows the objects that people deal with in each area of the app, and how those objects relate to each other.



Simple idea: anything in the airport gets solved faster if people can easily regroup and talk about it.

The everyday “things” that make people team up tend to always be the same: flights, tasks, organizations...

Why don’t we flip it around: help users find those everyday things, and assign each thing a dedicated chat room. Got a question on flight AA123? Find flight AA123 and jump into the chat!

Objects of attention like flights, tasks, and organizations, already useful from a data standpoint, become *chat contexts*: reasons for people to come together and discuss.

Generalizing, I proposed five types of chat contexts, and more could be added if needed. The first three are managed by the airport and the two others by users directly.

- **Group**: for people of the same organization or department
  Example: Air China / IT
- **Flight**: for people involved or interested in a particular flight
  Example: AA123
- **Location-based**: for location-sharing people who are detected near the same airport beacon
  Example: Terminal B South Wing
- **Task**: for people involved or interested in a particular task
  Example: Wheelchair to gate 22
- **Direct chat**: for people just willing to chat privately and informally
  Example: Lisa, Ben and I

SITA Airside (5/6)

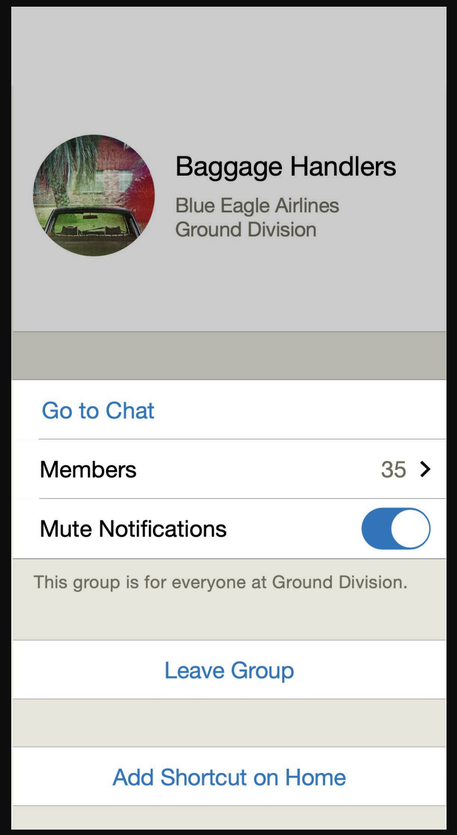
In the UI an object is presented as a card with specific features (depending on the object's type) and common features (such as chat).

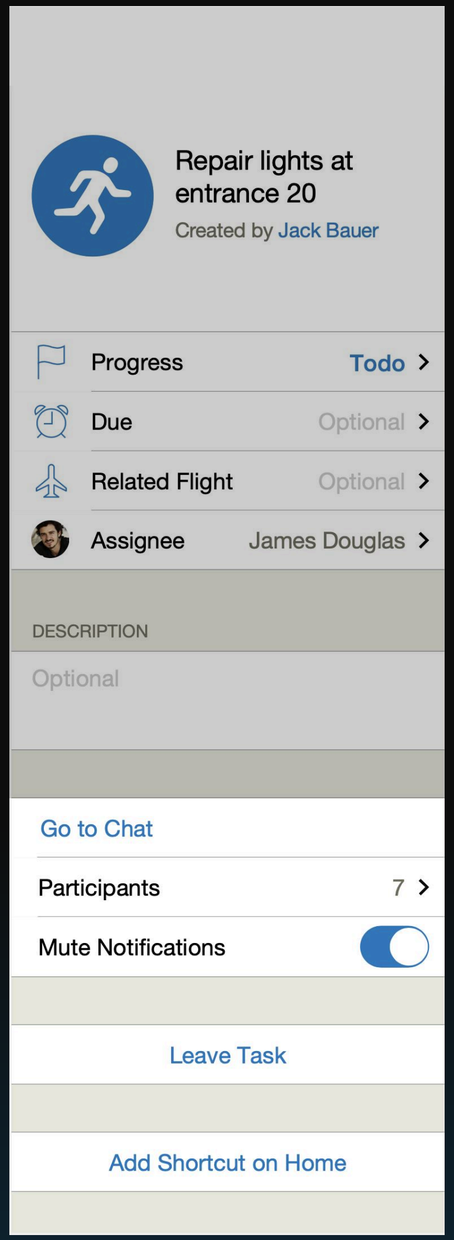
A chat always works the same way, regardless of context, which makes it universal and easy to evolve.

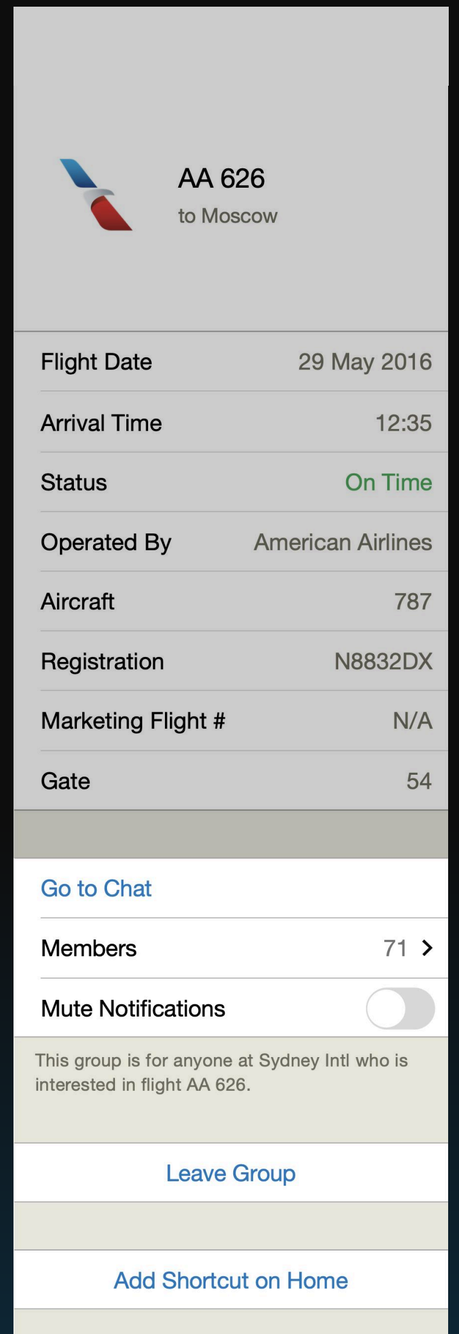
To further add leverage and flexibility, here were additional universal rules:

- Every object can be shared in a chat
- Every object can be pinned to the Home screen

Rules that can be made universal often contribute to global simplicity and robustness because they mean less special cases.







Header

Specific

Common

SITA Airside (6/6)

How we worked

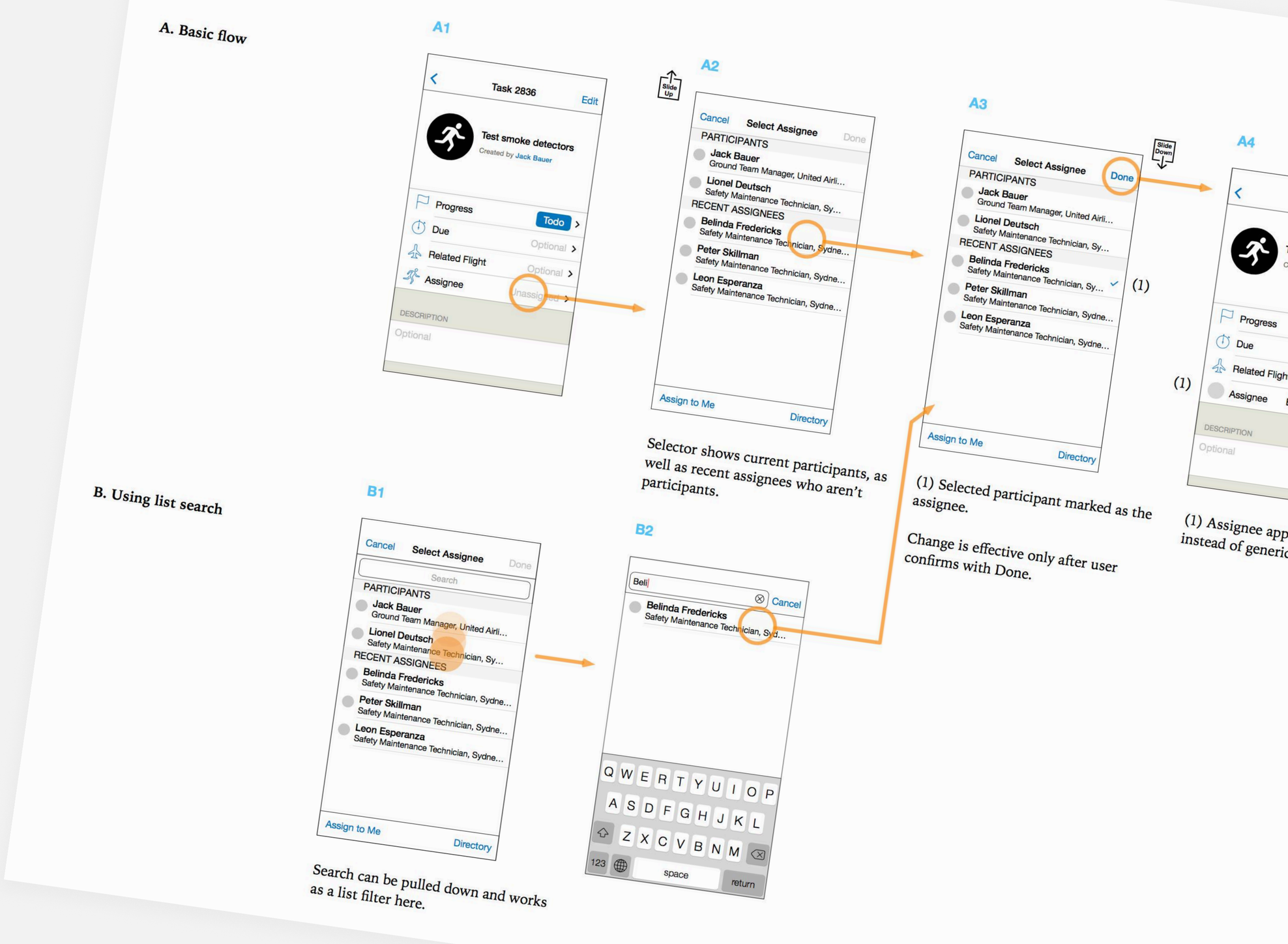
A very geographically distributed team, with PM, designer, and devs all in different timezones. A study was conducted by SITA with some airports to shape the need. Design and development progressed in parallel, one feature group at a time. We presented prototypes to potential customers and adjusted based on feedback.

I organized the scope in feature groups, for each of which I produced:

- Requirement list and definitions
- Interaction design (wireframes)
- Visual design (redlines & assets)

This made the spec easy to grasp and update. Requirement lists formalize business rules that aren't related to the UI spec and are better off specified separately.

Wireframe example. *Favoring clarity over completeness, the spec says the essentials but for the rest I like to rely on readers' common sense and experience, as well as on direct communication with them.*



Design principle:

Emergence

Emergence is when a system shows interesting new characteristics and behaviors that weren't designed, but that just *happen* at runtime. Classic example: surprising shapes created by flocks of birds. Birds are not programmed to do shapes. Shapes simply result from a few basic rules that the birds follow.

When it's hard to tell what the users of a system will do in the future, when too many scenarios seem relevant, designers should focus on the seeds rather than the fruits; on the DNA rather than the flesh. In other words, on the basic rules, to leave users space for creativity and adaptation.

I had eureka moments related to emergence before, though I didn't understand it right away. Once, I was trying to prototype a communication involving multiple phone users; another time I was designing a location sharing feature for teams.

Each time, the emergence-based solution felt simpler, and at the same time, infinitely flexible.

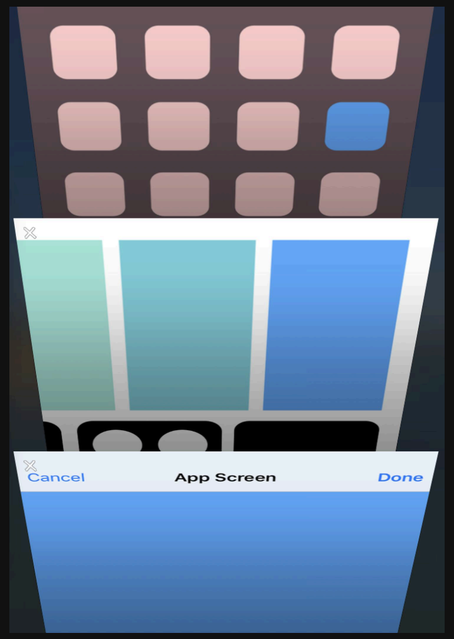
"One of the founding principles of agility, and the closest one to pure magic." – Andrew Hunt

More

A selection of extra projects and initiatives that helped me grow as a designer, analyst and manager.

The Sentinel is an extraordinary game (2017)
Article on Medium

Design essay on a video game classic, released in 1986 by Geoff Crammond. The article shows how, by embracing design constraints, Crammond achieves a coherent system with timeless gameplay.



Homeless iPhone (2017)
Article on Medium, well received and featured on J. Dalrymple's The Loop.

Before the announcement of the groundbreaking iPhone X, in this experiment I speculate about the removal of the iconic Home button, and show with animation how swipe gestures can replace it. Got close to what Apple announced a month later.

See also the follow-up story that dissects my process.

Nokia X (2013-2014)
A fun, Android-based and affordable Nokia smartphone, just before Microsoft acquired Nokia. Memories of great user research in the field, with teenagers of Indonesia; this was also my main team lead experience, at a wonderful company.

My role:
Team lead (6 interaction designers)
Design lead (for 2 of 3 device models)
Design reviews and delivery
Direct contributor for some features

My Orange (2012-2013)
Evolution of the app used by Orange mobile subscribers to manage their account and options. A simplified and modular redesign to adapt to the needs of different markets/regions. My main project in London, where I learned a lot.

My role:
Design lead 2012-2013
Information architecture overhaul
Interaction design (iOS, Android)
Design patterns

Updated June 2026

Email
fabrice@uxriver.com

Site
uxriver.com

LinkedIn
linkedin.com/in/fabdubois

uxriver