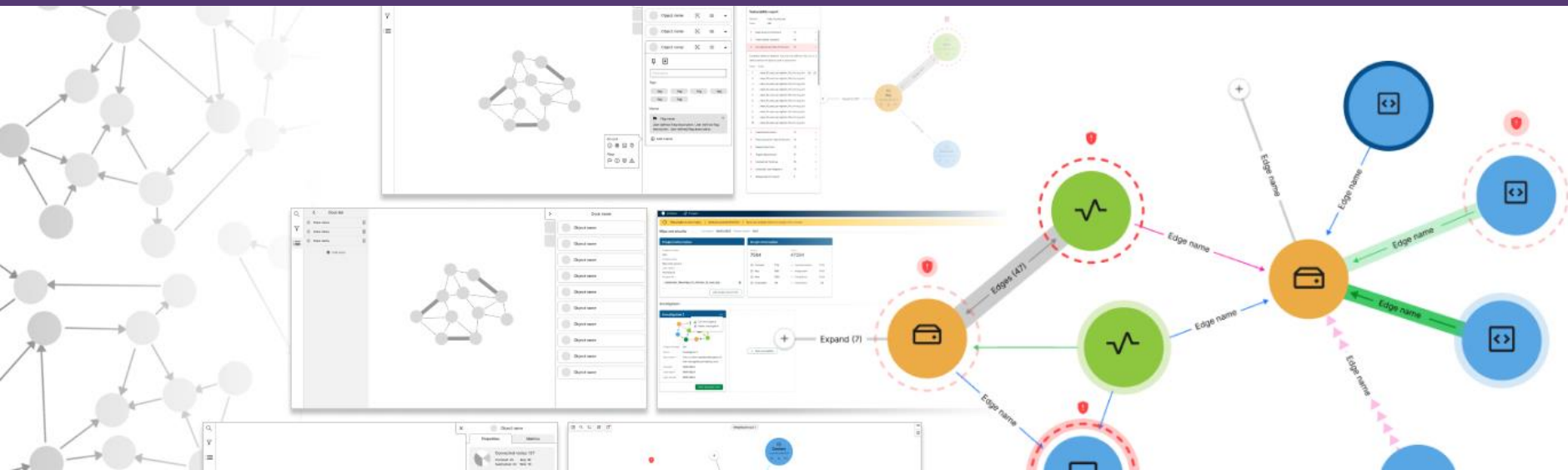




**Nick Osmanski**  
Senior UX Designer

## *Case study 2*

# Data visualization application



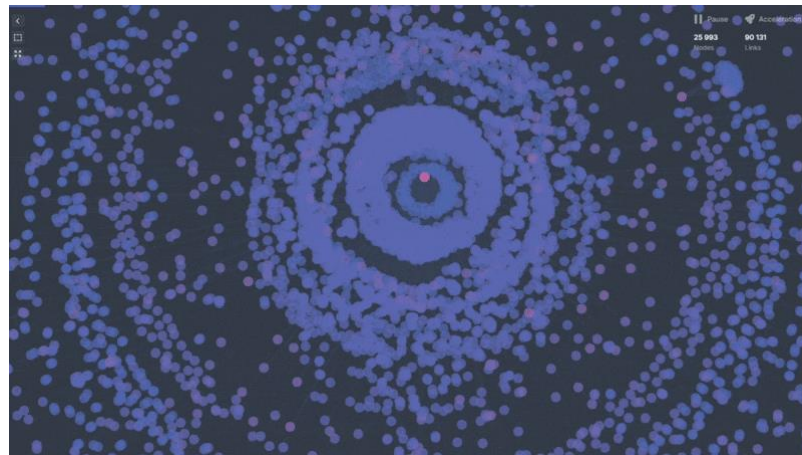
## Background

- Amida's management and engineering team wanted to leverage their expertise to fill a gap in the cyber security market
- They resolved to use an existing tool that incorporated graph theory and data visualization

## The problem

- The closest existing tool did not have the depth of interactivity needed to fulfill the intended use case
- Amida could not demonstrate their ability to fill the gap in the market without this functionality

Node and edge graph (Cosmograph)



(25,000 nodes; 90,000 edges)

## My role

- The sole designer assigned to this project team, supporting:
  - Front-end developers
  - Back-end developers
  - Internal SMEs
  - Internal PMs and upper management
- Final deliverables included:
  - Developer-ready design system components
  - Hi-fi mockups and prototypes
  - Demos and presentations
  - (5-6 month project time)

## Solution & impact

- A custom tool built in-house, incorporating graph theory in its data visualization, progressive levels interaction, and demos and marketing materials designed to pitch the solution to potential customers
- Positive reactions from internal management, SMEs, and potential customers
  - The prototype solution was pitched to a number of leaders in the cyber security field, to positive feedback, and new business leads for Amida

## Interviews

- Team SMEs:
  - Front-end engineers
  - Back-end engineers
  - Graph theory & computer science experts
- I spoke with the team as a group, and each member individually
  - I learned the basics of node and edge graph visualization
  - They explained current solution limitations
  - Suggested features

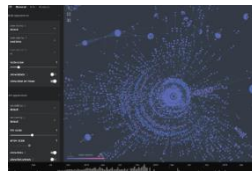
## Competitive analysis

Neo4j



- + Polished UI and useful features
- Limited depth of detail on nodes and edges, in terms of visual representation of the data

Cosmograph



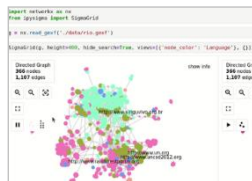
- + Large data set representation, real time graph movement
- Relatively simple UI features, and limited node/edge visual detail

yFiles  
library



- + Many different options for displaying data as node-and-edge graphs
- Limited detail and interaction for individual nodes and edges

ipySigma  
library



- + High level of customizability via library API access
- Once again, limited out-of-the-box node and edge interaction detail



### John Persona, 42 *Cybersecurity engineer*

John is an educated cybersecurity engineer and analyst. He leads a team of supporting analysts, and he reports directly to leadership about critical business decisions.

“

I need to access low and high levels of information density, while retaining context, in order to see the bigger picture.

”

#### Goals

- ! John wants to make informed decisions about technical issues quickly and accurately.
- ! John wants to provide critical updates to leadership, who may include non-technical people.
- ! John wants to communicate issues to the customer, simply and effectively.

#### Needs

- + John needs to navigate the graph quickly and intuitively, while retaining data context.
- + As an SME, John needs specifically requested required features to be implemented.
- + John needs the application to recommend solutions for given cybersecurity scenarios.

#### Frustrations

- John doesn't like the simplicity of the existing data visualization solutions.
- John doesn't like having to manually search through data that is difficult to parse.
- John doesn't want to waste time recreating the same analysis scenarios for different data sets.

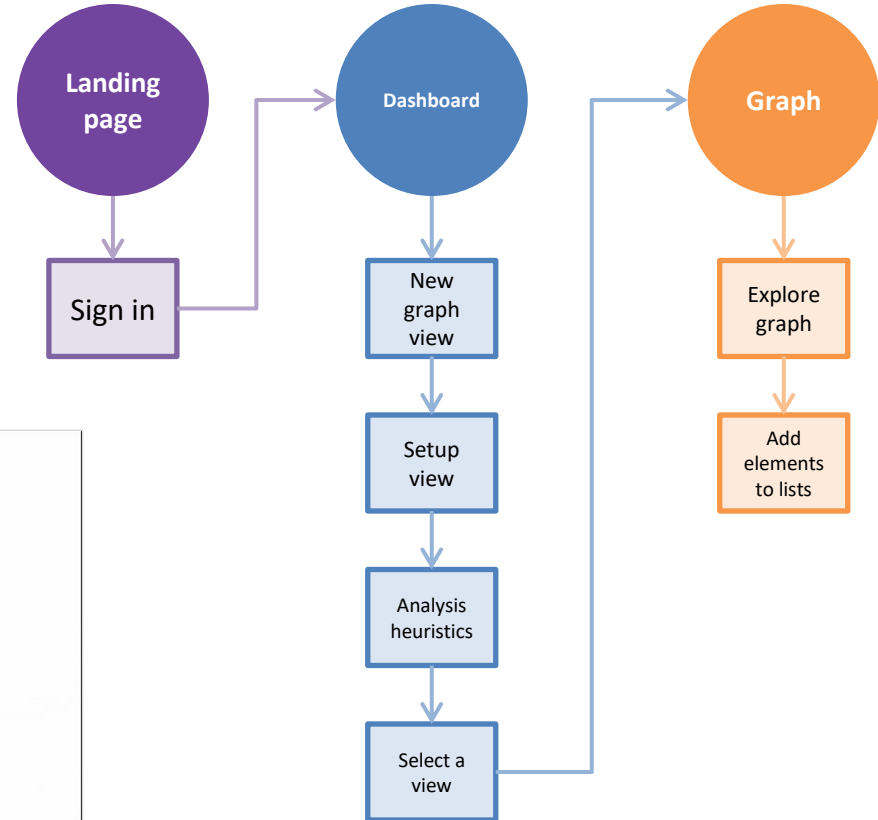
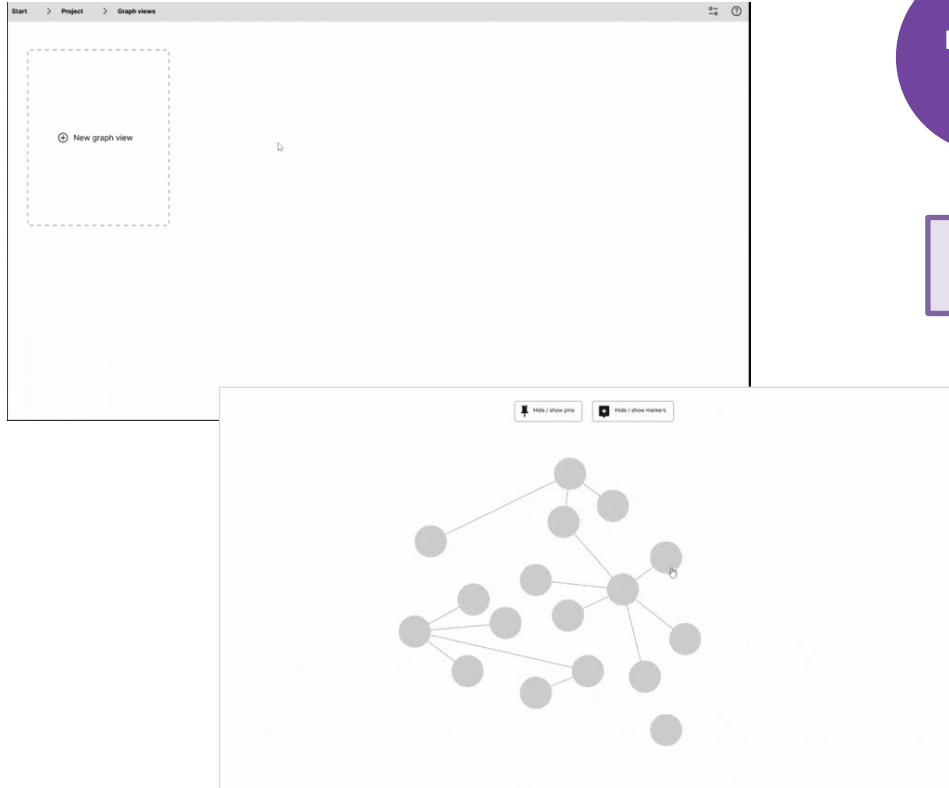
## MVP feature consolidation

- The team consisted SMEs, and each had a list of “required” features that would suit their workflows
- I used an importance/difficulty matrix to consolidate MVP features, including:
  - A dashboard
    - Project setup
    - Multiple graph “views”
    - Analysis heuristics
  - The graph
    - Graph element interaction
    - List of elements

### Importance (X axis) / difficulty (Y axis) matrix

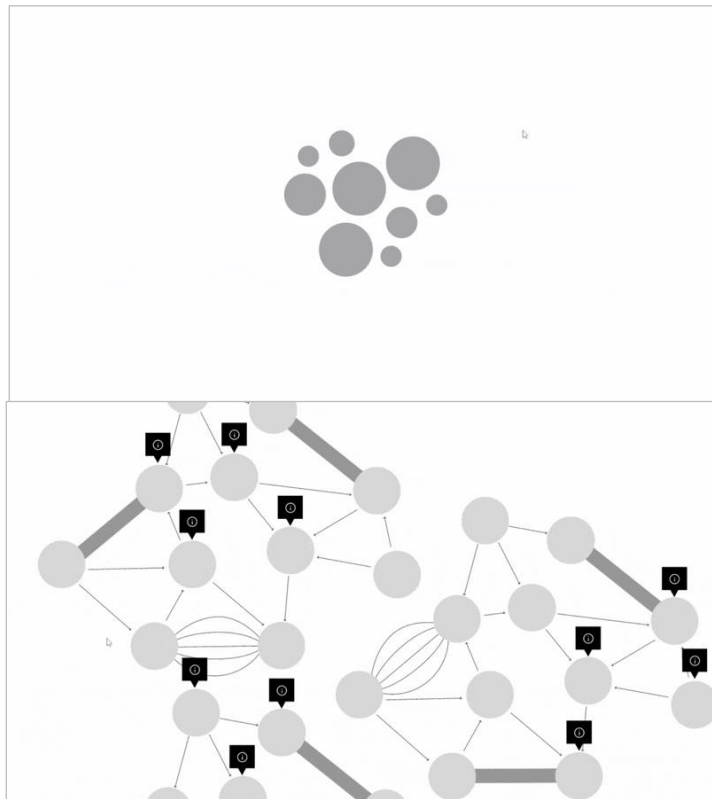
# Data viz application | *User flow*

Lo-fi dashboard (top) and graph (bottom)



# Data viz application | *Graph UI ideation*

Zoom/pan, edge groups, and marker clustering



- 6 To solve this problem, I represented large numbers of edges as a group, visually distinct from other single edges.

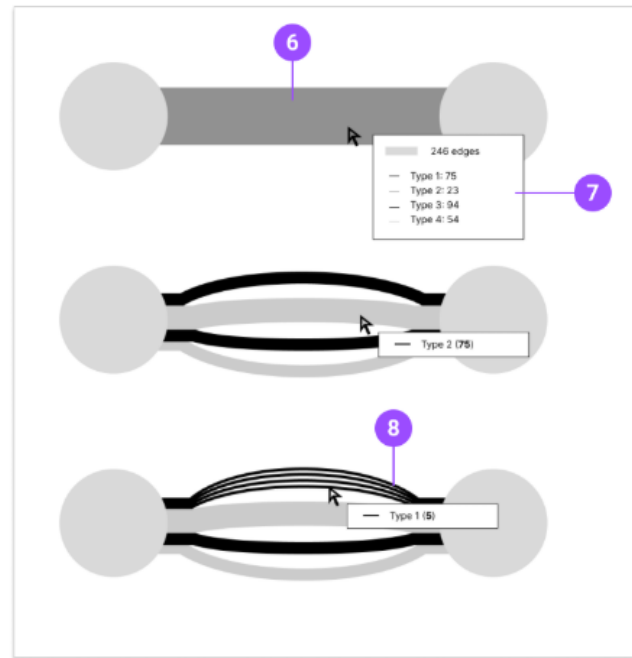
Through additional layers of interaction with the graph elements, users could reveal more information about this group.

- 7 Hovering or right clicking brings up a context menu, wherein users can choose to expand the edge group.

The same pattern of right clicking to access more actions, could be applied to groups of nodes as well.

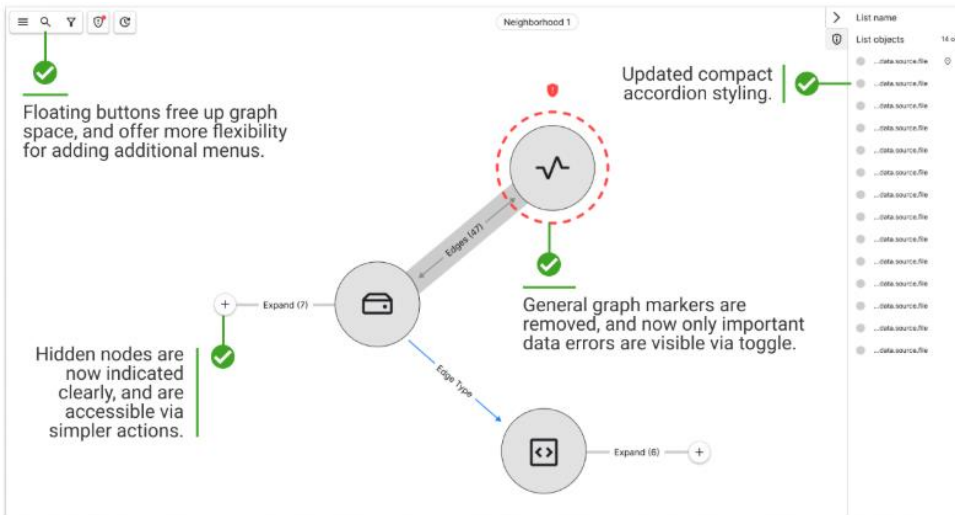
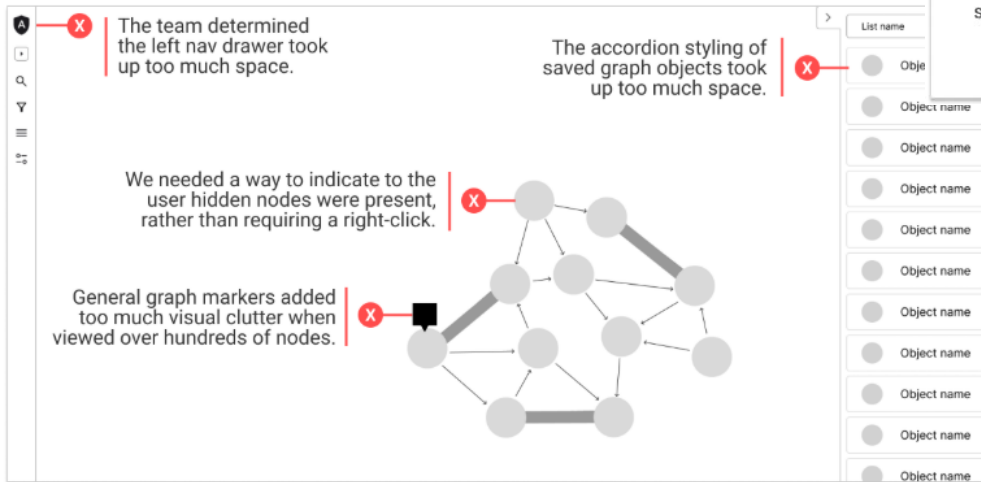
- 8 Once expanded, users can see each edge type as part of the group. Initially we thought it made sense to allow users to continue expanding nested edge groups, but this was removed and simplified in a later design iteration.

Arbitrary edge counts



## How & who

- Remote testing with internal SMEs, using Figma prototypes
  - (Devs rarely caught up to design)
- 5 SMEs, internal PM/management



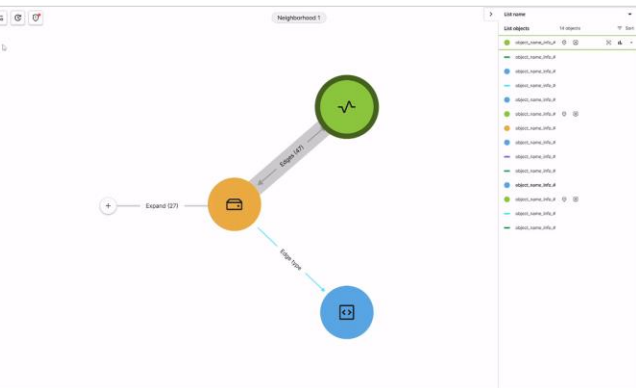
## Results

- Iterative updates to dashboard and graph UI
- Refined representations of MVP features

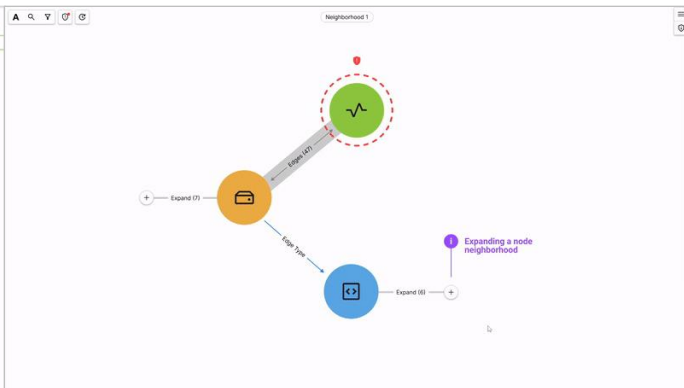
# Data viz application

## Final deliverables

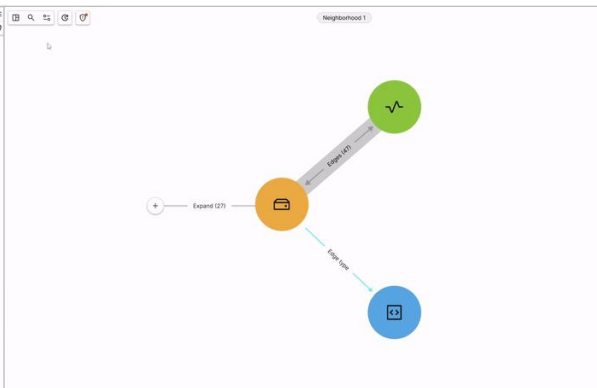
### Search & lists



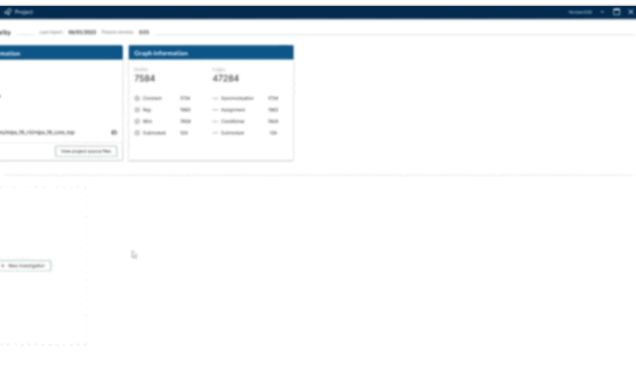
### Expanding neighborhoods



### Options menu & tabs



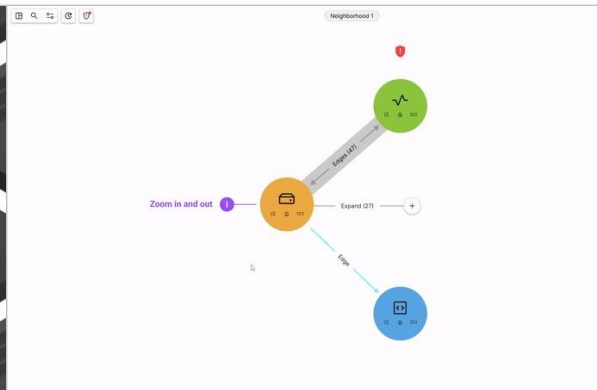
### Dashboard (blurred)



### Marketing animation



### Icon zoom



## Mini retro

- Lessons learned
  - Communicating with technical experts
  - Dealing with scope creep
  - Demo and marketing voiceovers / animation
- Do anything differently?
  - More robust testing (NDA issues?)
  - Look into 3D representations of layered data

Thank you!