



Kickstarting libraries of shared React
components **for** multiple teams

 Hi!

Me : Xavier Lefèvre

Job : Team leader of React and React Native projects

Company : BAM, a tech consulting and development agency

Passion : I love tech as much as I love travelling which means I'm super happy to be here! I do also love motorcycles.

Making reusable components across projects...



Contents

1. The origin and attempts of sharing components
2. A deep dive in some of our design decisions
3. How we organised ourselves to make it a success

It started with a wish

"It would be awesome if we could finally and successfully create **shareable components** in order to **re-use them on different apps**, give them to other departments or even one day sell them outside!"

Quote from our client in April 2018


Why?

With our client, we produce several web apps per year for different targets but with the same business behind.

So very similar components were remade from scratch many times.

Let's see what our client means 🤔

[My profile](#) [Fund selector](#) [My location United Kingdom](#)


**BNP PARIBAS**
ASSET MANAGEMENT


The asset manager for a changing world

[Home](#) [About us](#) [Our responsibility](#) [Our expertise](#) [Careers](#) [Press](#) [News](#) [Contact](#)


[BNP PARIBAS A FUND EUROPEAN MULTI-ASSET INCOME - EUR](#)
Balanced | Europe Balanced | Europe | Multi-Asset Flexible (Europe)
[Simplified prospectus / KIID](#) [Factsheet](#)

[Overview](#) [NAV](#) [Documents](#)

DETAILS  Delegated


Share type
Privilege, Capitalisa... 

Displayed currency
Euro

Morningstar rating
 3/5
at 1/31/2019

ISIN Code [Copy](#)
LU1078739452

Fund Manager
DUPIRE Clement since 4/2/2018

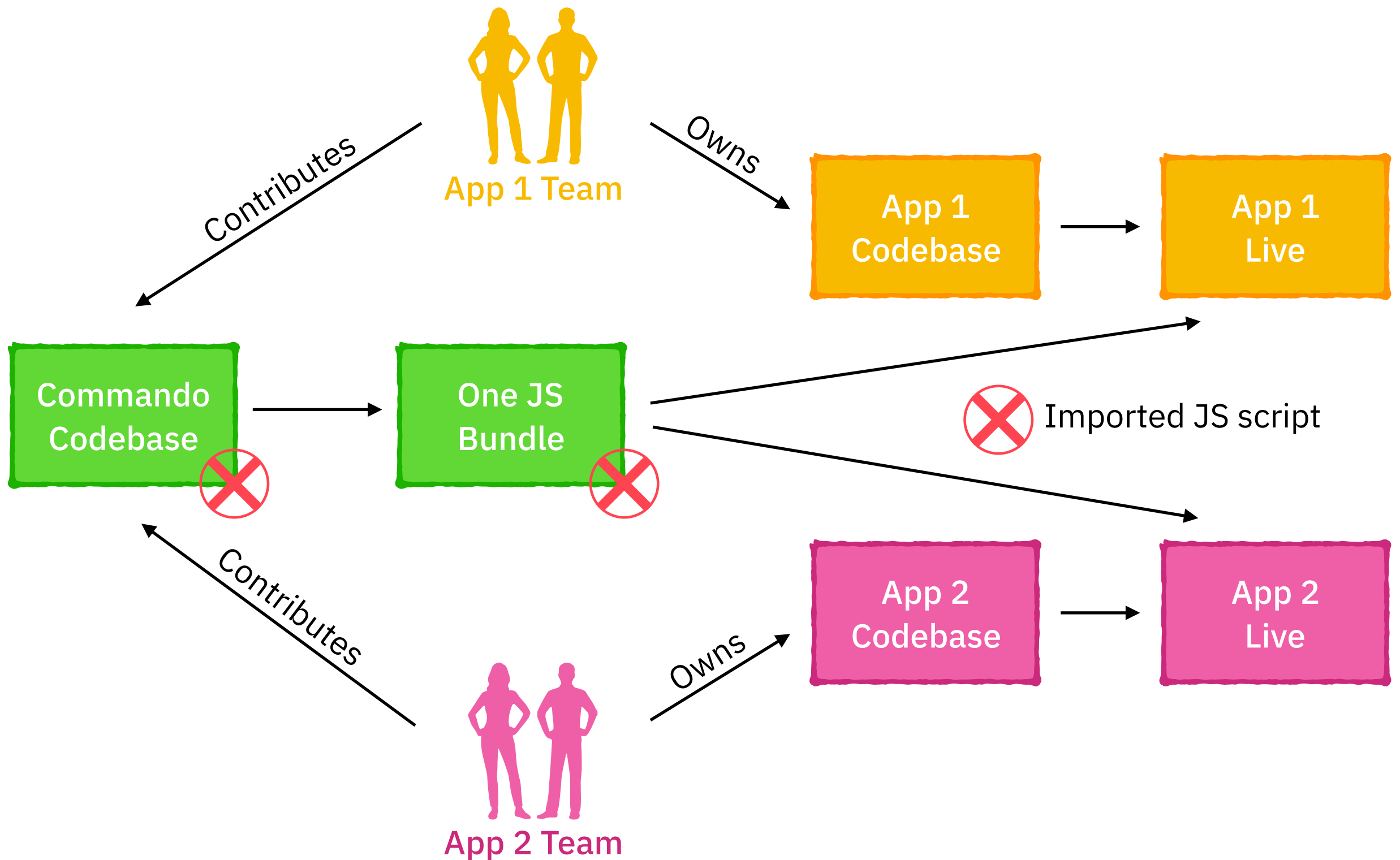
 Please note that there is (are) one (several) [hedged](#) share class(es) in this fund. For additional information, please refer to the prospectus.

RISK AND REWARD INDICATOR

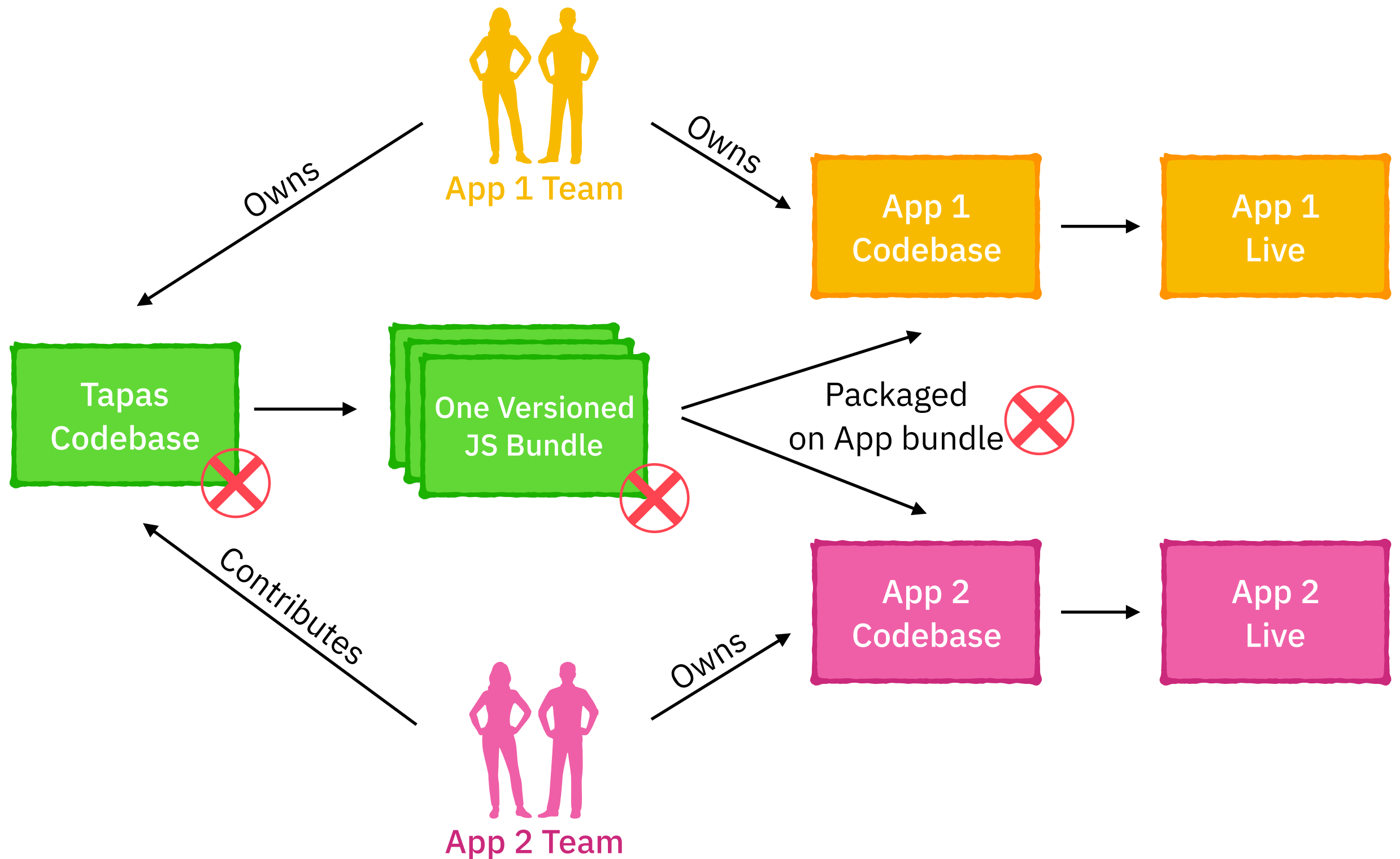
Risk and Reward Profile : 1 : Lower risk - Typically lower rewards. 7 : Higher risk - Typically higher rewards

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#)

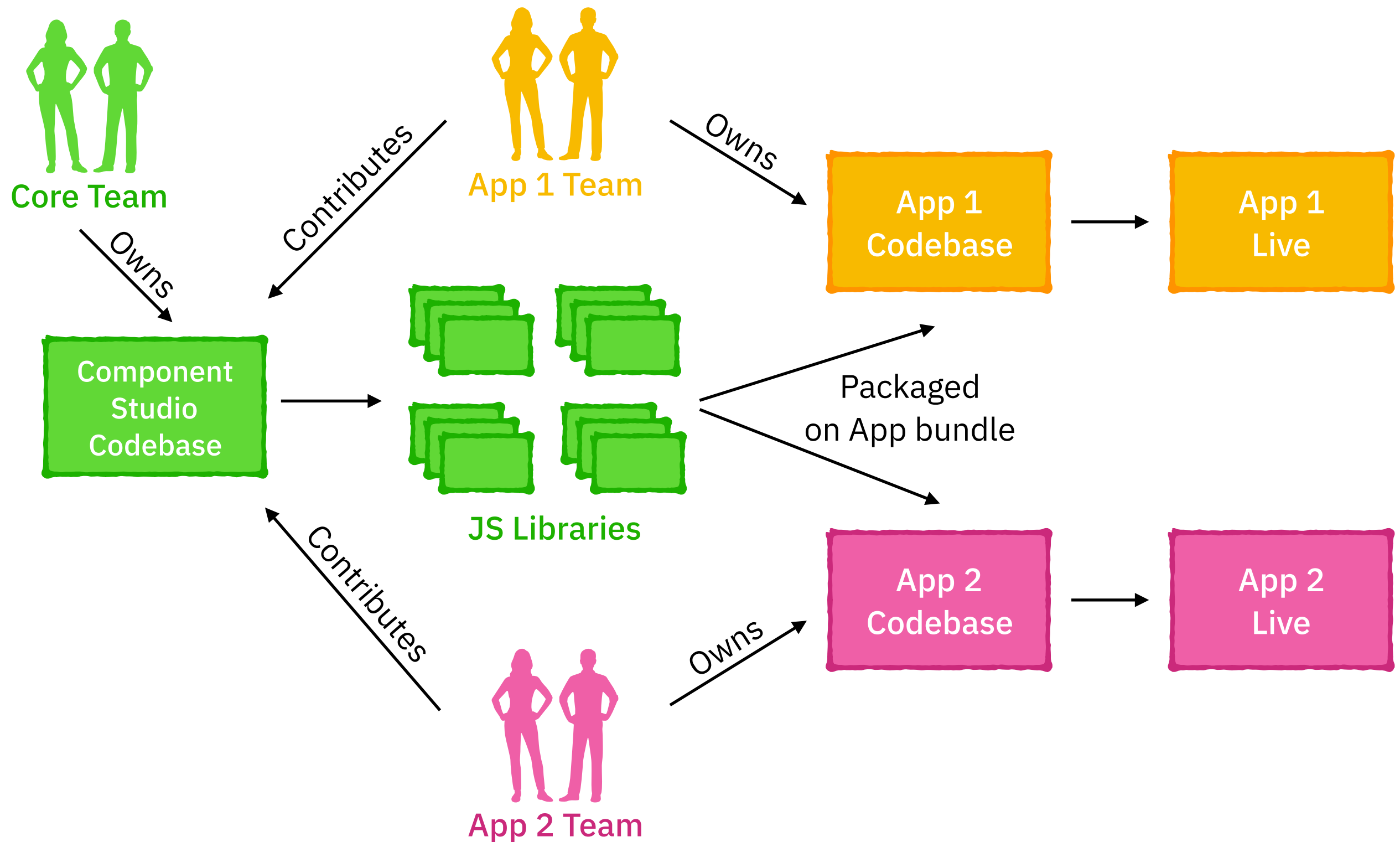
First attempt: Commando



Second attempt: Tapas 🌮



Latest attempt: Component Studio



Demonstration time 🍿

COMPONENT STUDIO

Filter

Products

Fundsheet

Fundsheet Light

Smartbot

ShareComparator

Fund Panorama

Utils

LanguageProvider

UI

Widgets

Authentication

Miscellaneous

Development

Fund

Marketing Text

Dividends

NAV

Characteristics

Documents

Header & Selector

Performance

Risk

< BNP PARIBAS AQUA - EUR

Actions | Actions internationales | Monde | Actions internationales - Core

Ajouter aux favoris

Rapport Mensuel

DICI

Aperçu

Performances

VL

Documents

DÉTAILS DU FONDS

ISR

Fonds délégué

Assurance-vie

Catégorie d'actions

Classic, Capitalisation

Devise

Euro

Note Morningstar

★★★★☆ 4/5

au 30/11/2018

Code ISIN

FR0010668145

Copier

Gestionnaire

AARTS Hubert Depuis le 19/09/2015

NIVEAU DE RISQUE

1 : risque le moins élevé. 7 : risque le plus élevé. SRRI : indicateur synthétique de risque et de rendement, plus le risque est élevé, plus l'horizon d'investissement recommandé est long

1234567

KNOBS

ACTION LOGGER

Samples

fr_FR-FR0010668145 (classic)

Fundsheet Params

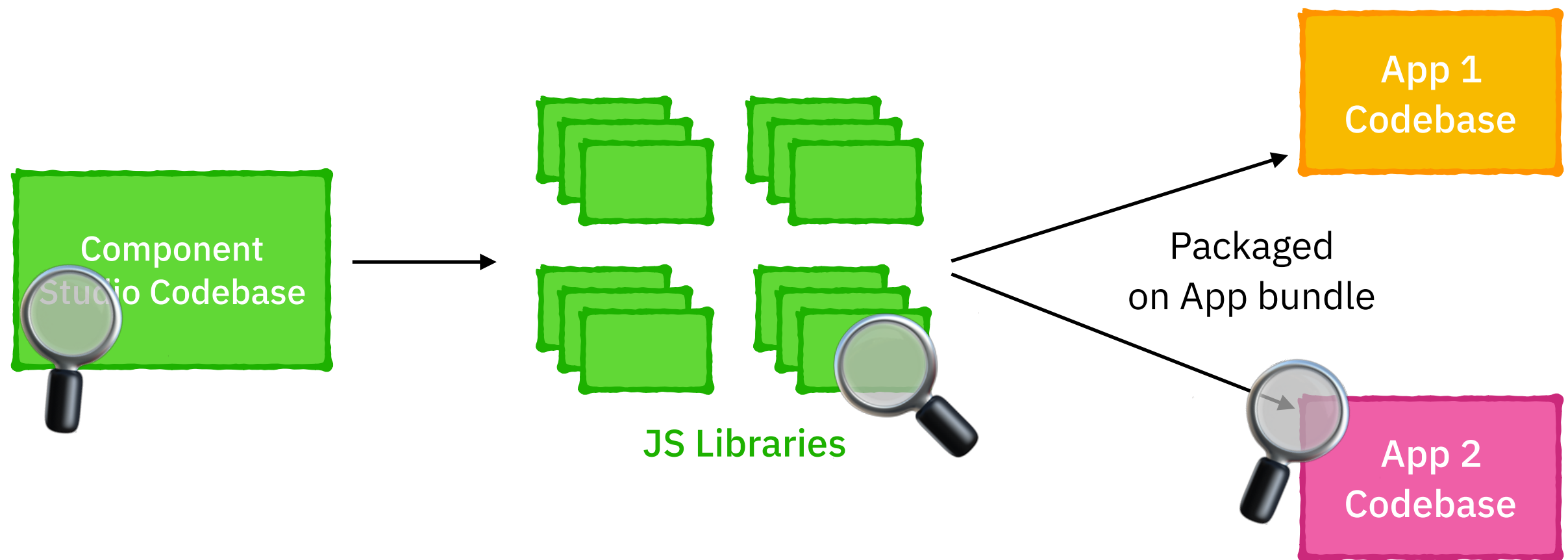
{ "segment": "IP_FR-IND", "language": "FRE", "country": "FRA", "isin": "FR0010668145" }

COPY

RESET

Let's dive in for some tips!

Let's zoom in 3 interesting aspects



2. Components
code structure

1. Packages and
components
organisation

3. How to limit the
impact of breaking
changes

We follow Atomic Design principles

Organism

< PARVEST AQUA - EUR

Equity | Global Equity | Global | Global Sector

Add to favorites

↓ Factsheet

Overview

Performance

NAV

Documents

DETAILS

SRI

Delegated

Share type

Atom

Classic, Capitalisation

Displayed currency

Euro

Morningstar rating

NR

ISIN Code

Copy

LU1165135440

Fund Manager

AARTS Hubert since 1/7/2015

RISK AND REWARD INDICATOR

Risk and Reward Profile : 1 : Lower risk - Typically lower rewards. 7 : Higher risk - Typically higher rewards

1

2

3

4

5

6

7

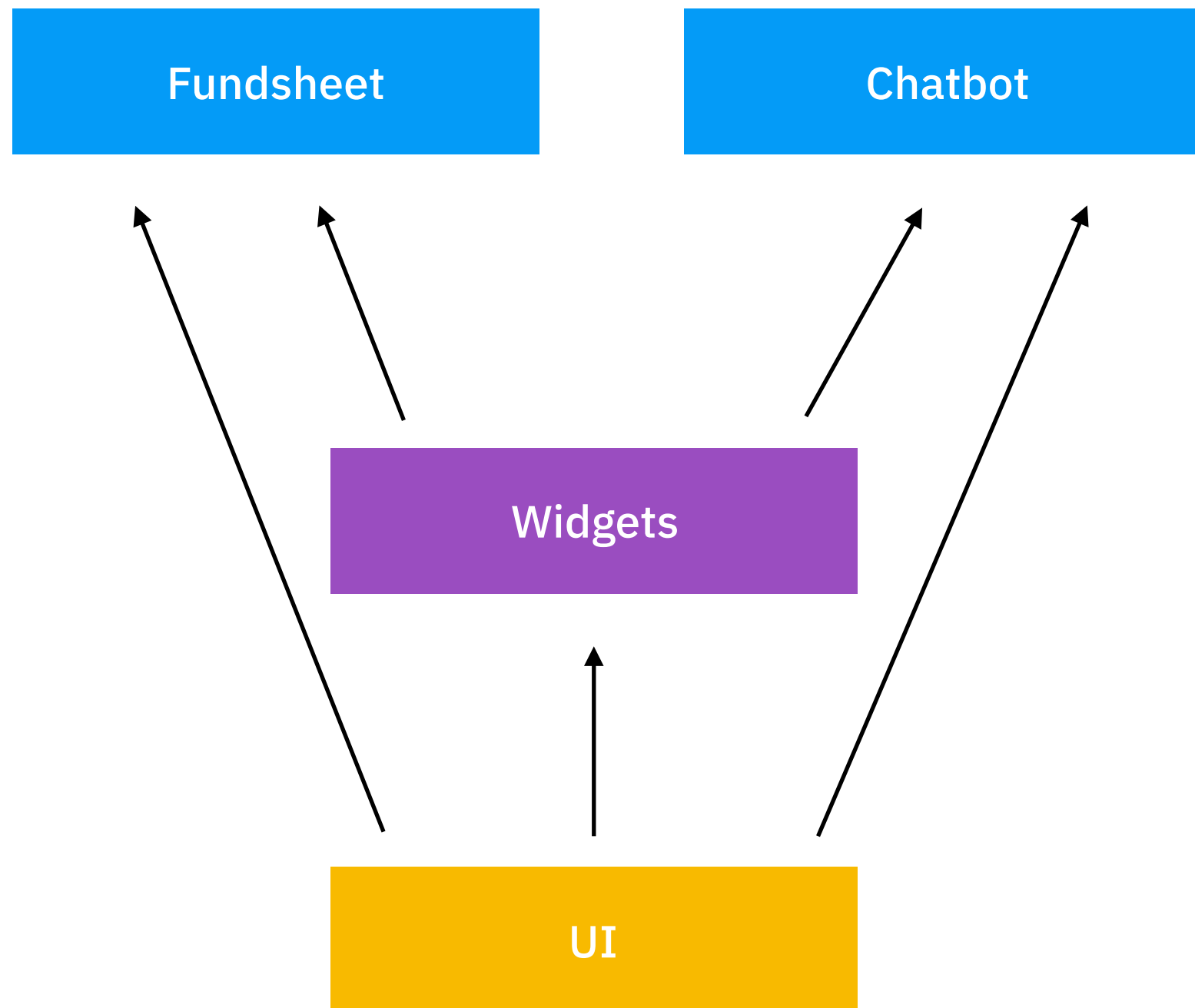
Lower risk

Higher risk

Typically lower reward

Typically higher reward

The impact on our code organisation



Organisms

Full page feature -
Satisfies a user need

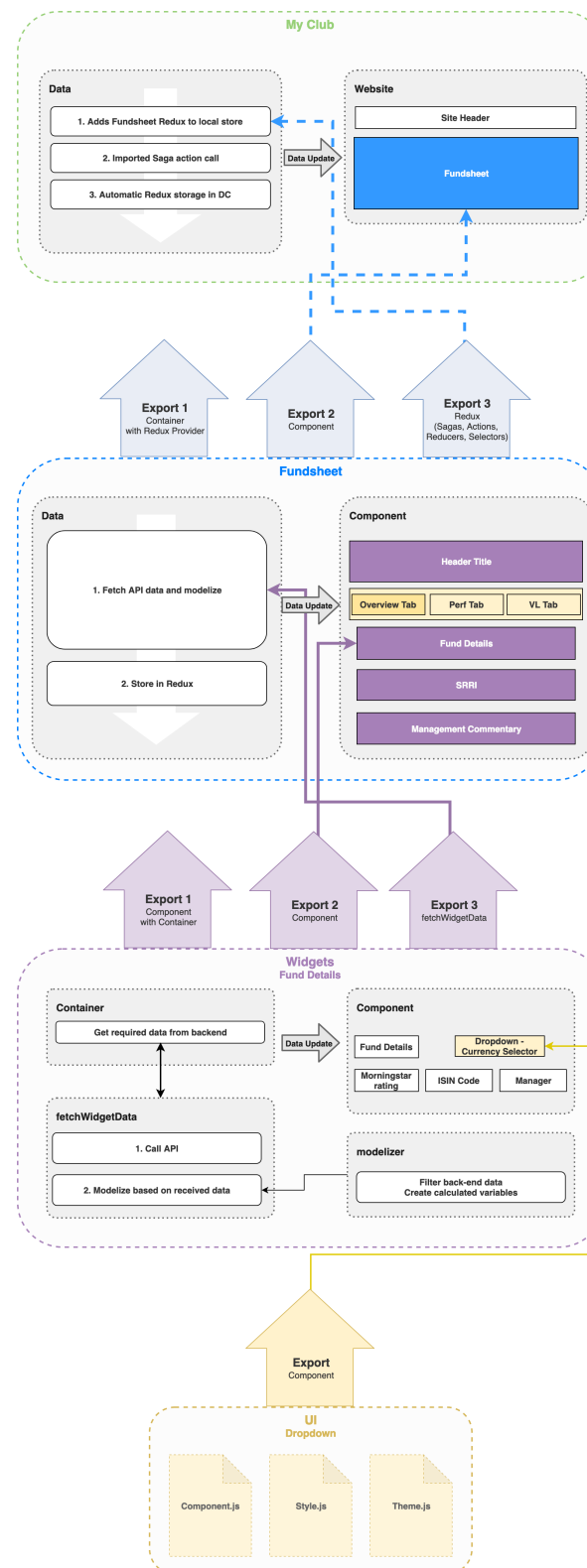
Molecules

Business connected -
Composable when sharing
a common business
purpose

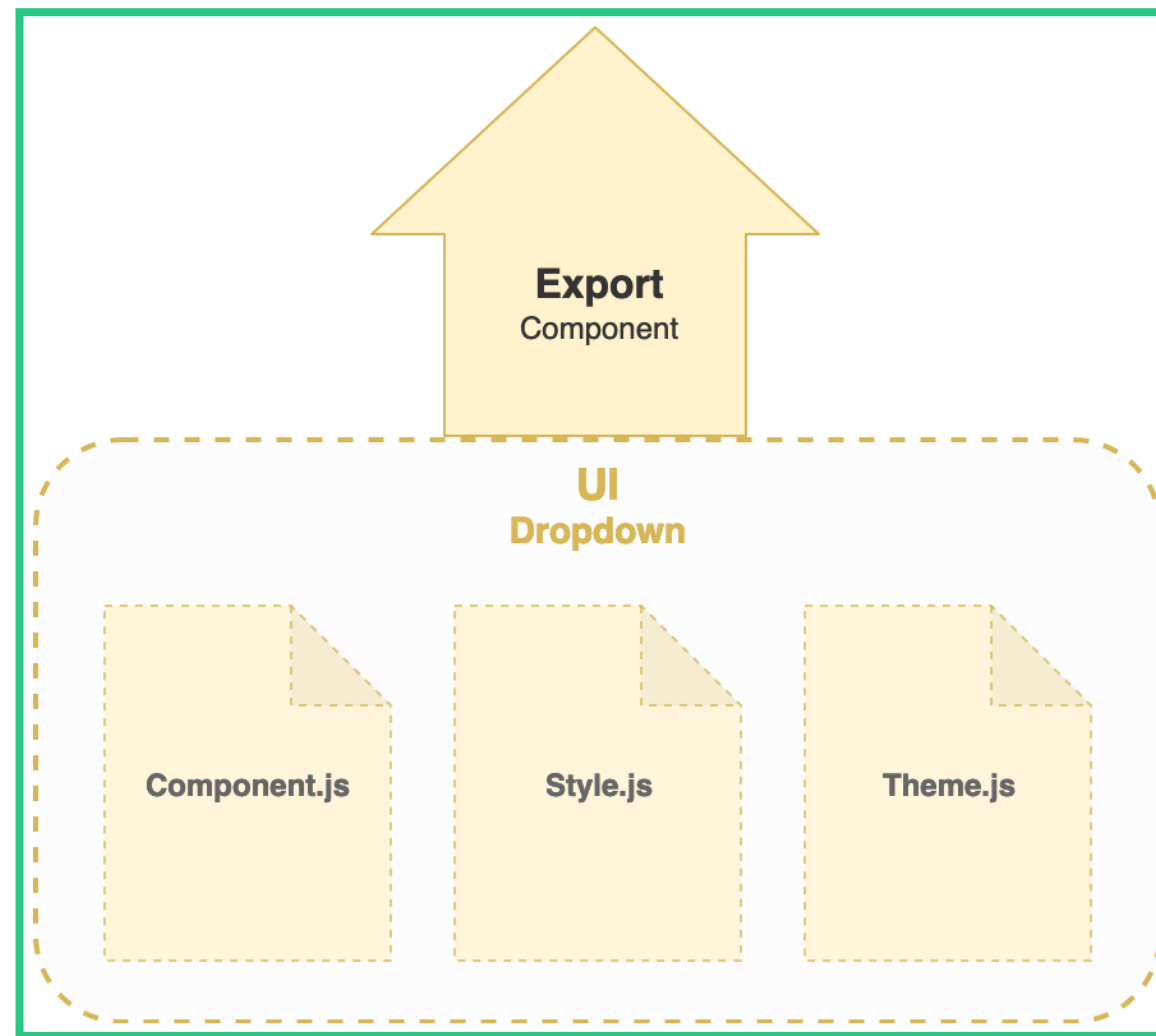
Atoms

Pure UI - No business -
Totally Customisable and
Composable

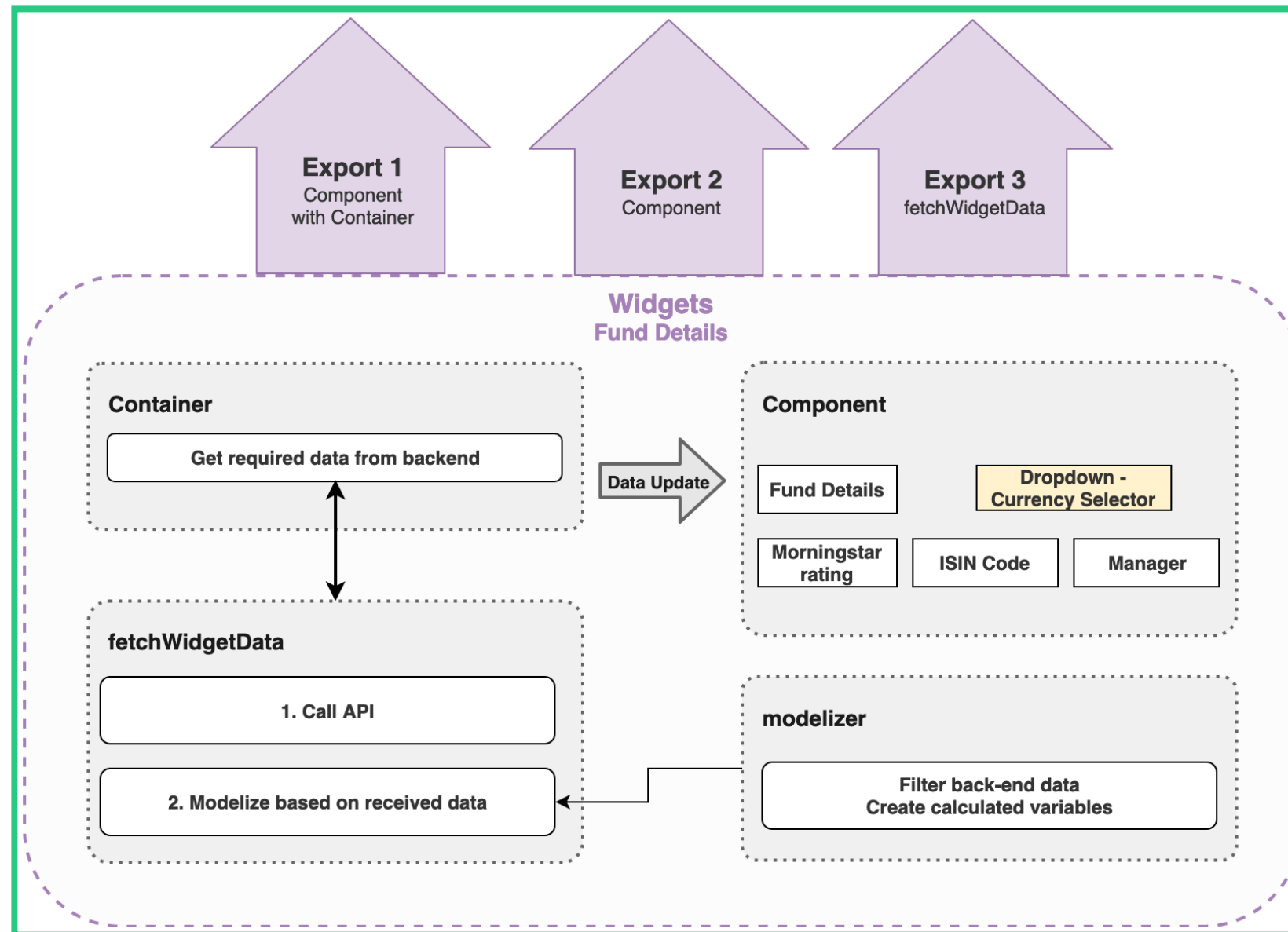
A deeper look at the components code



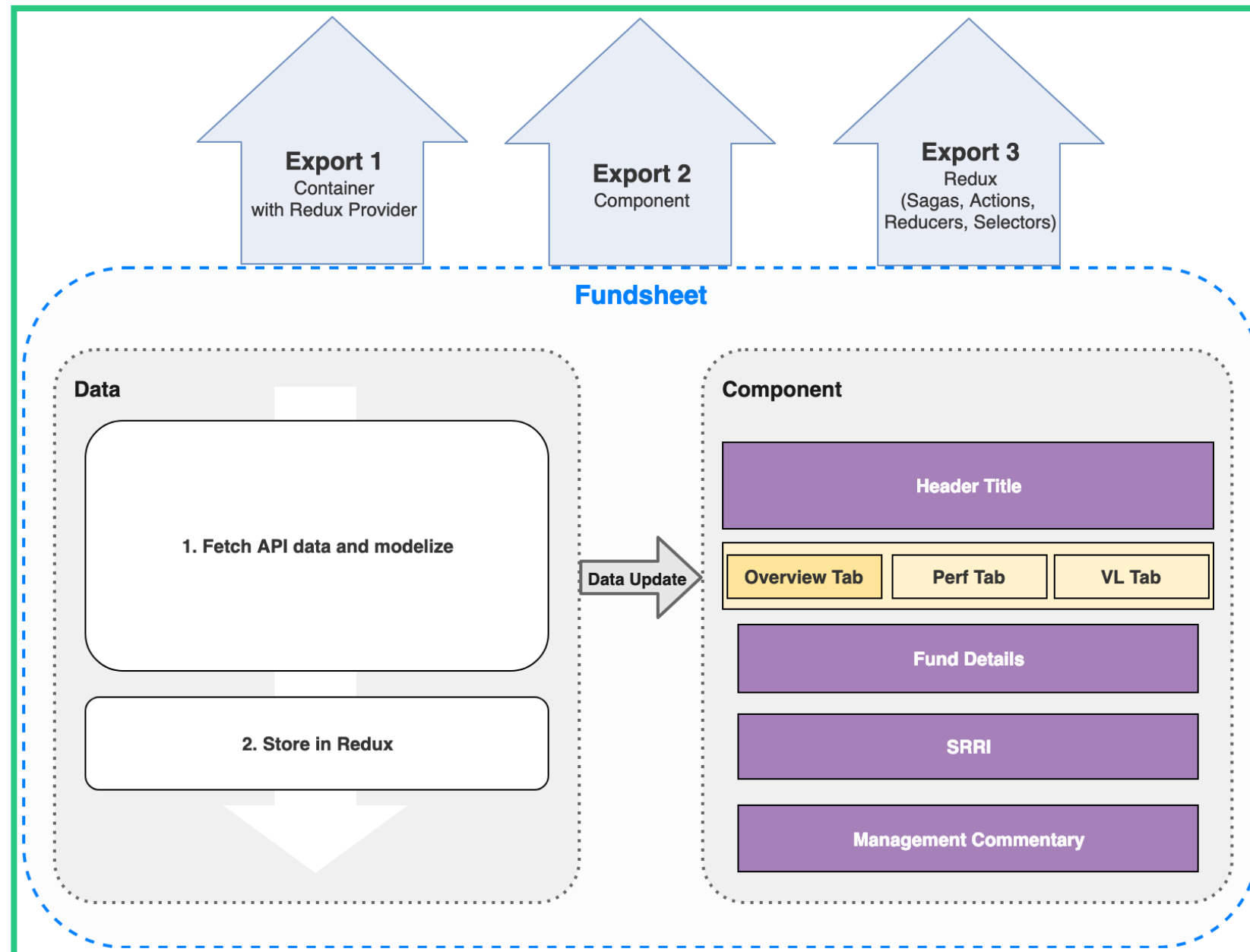
A deeper look at the UI code



A deeper look at the Widgets code



A deeper look at the Products code



What is a breaking change?

A **breaking change** is a **change** in one part of a software system that **causes other parts to fail**.

Which happens when you change the existing API of a library without warning your users.

What should I pay attention to?

- Exposed components and functions names
- Components props or function parameters
- The component style, but more risky its taken space (height, width...)
- An upgrade in a peer dependency which has breaking changes


```
export const Button = (props) => (  
  <button  
    disabled={props.disabled}  
    onClick={props.onClick}  
    type={props.type}  
    className={props.className}  
  >  
    {props.children}  
  </button>  
)
```

```
export default styled(Button)`  
  ${({ disabled }) => `  
    border: 1px solid ${colors.green};  
    border-radius: 3px;  
    ${disabled ? '' : 'cursor: pointer;'}  
    height: '18px';  
    line-height: 17px;  
  `;  
`;
```

What we set-up to help developers anticipate

- API exposed as exported Flow types
- Changes in the Flow API notified by Danger on the Pull Request
- Automatic SemVer based on commits, releasing major when breaking change
- Automatic changelog generation

```
declare type ButtonPropsType = {|
  children: React.Node,
  className?: string,
  disabled?: boolean,
  light?: boolean,
  onClick?: (event: EventType) => void,
  type?: string,
  small?: boolean,
|};
```

Warnings	
	<p>Breaking changes risk:</p> <p>You changed the exported flow types, be careful because this might indicate a breaking change.</p> <p>Find here how to anticipate breaking changes on Component Studio.</p> <p>The list of exported Flow files you changed:</p> <ul style="list-style-type: none">• packages/widgets/flow-typed/PackageTypes/widgets_vx.x.x.js

How do we know our project is
successful?

Our return on investment

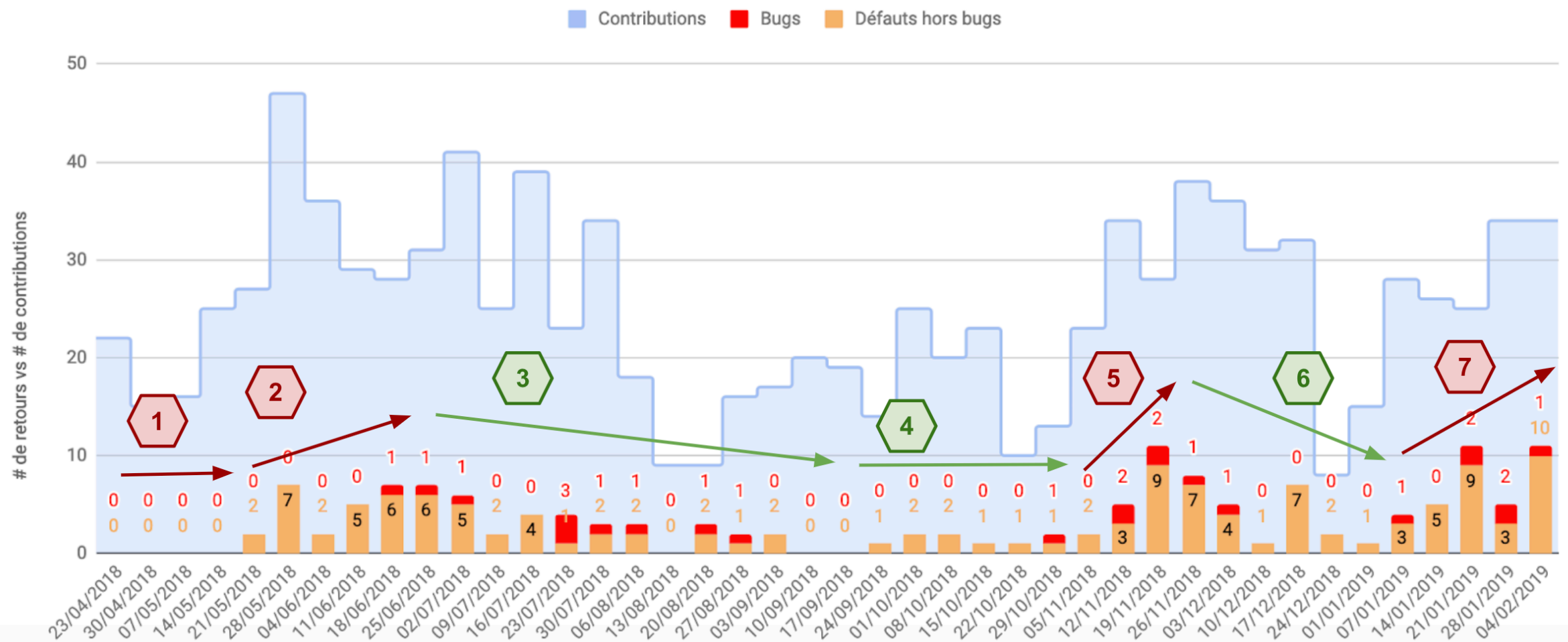
$$\sum_{\text{components}} \left(\text{Component cost if developed on "normal" project} \times \text{Number of times of reuse of the component} \right) - \text{Component Studio full cost}$$

The tech team main focus is to **reduce the project cost** by approaching a **flawless codebase** and a **fast development process**.

But we did not arrive to this point without **struggling** 🤯!

So we started tracking issues...

Number of issues VS Contributions



Every problem the contributors meet on the project they create an issue for the core team.

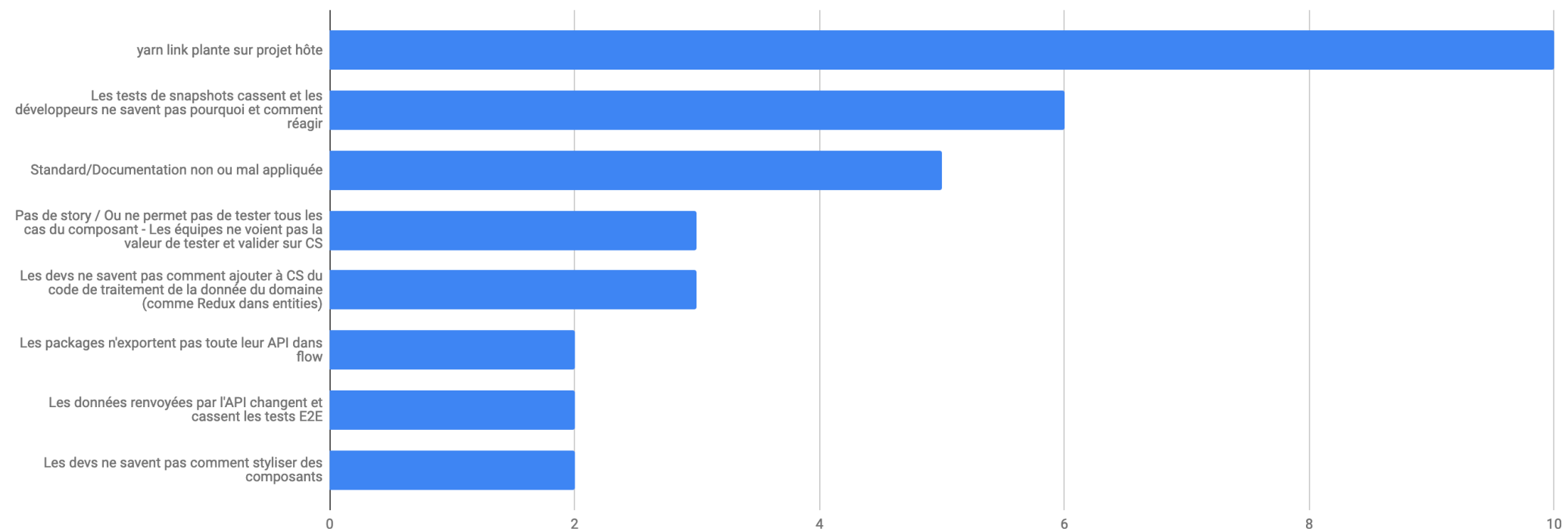
... and then tackling those issues!

1. An hour **every day** of issue prioritisation and solving:

- **10min:** We first take a look at the ongoing actions
- **20min:** We then prioritise based on the type of issue and the developer's lost time
- **30min:** Finally we take the most urgent issues and find their causes

2. **Weekly actions** to tackle the biggest root causes

Root Causes Pareto - Last Three months - (Edited 18/02/2018)



So, is it worth it?





We saved around 18% of development cost
since April 2018.

Now the UI is naturally more consistant across
all apps.

 Thanks!

Special thanks to everybody that worked
on this project:

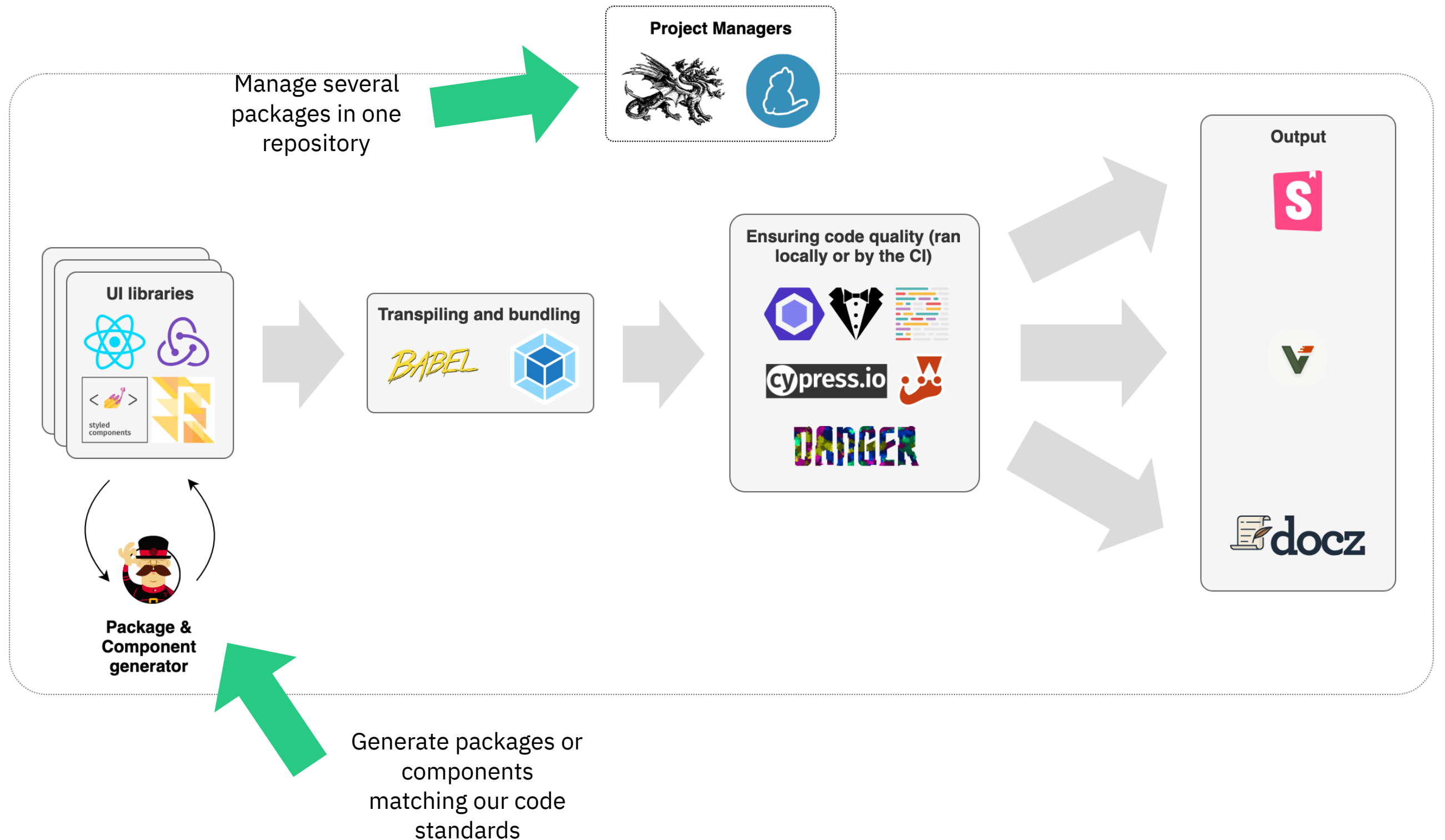
I am able to share all this because of a
huge team work since day 1.

 @xavi_lefevre
 xavierlefevre

Appendix

- Complete project stack
- Global architecture
- Atomic Design deep dive
- Development Flow
- Our current issues
- Links

What is the project made of?



Our stack in detail

Project Managers



Lerna: Multi package mono-repository manager



Yarn: Dependency manager

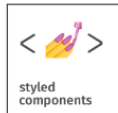
UI development libraries



React



Redux



Styled Components

Package & Component generator



Yeoman

Code bundling



Babel



Webpack

Code quality and tests



ESLint



stylelint



Flow: Static type checker



Prettier



Jest



Cypress.IO



Danger.JS

Showcase, store and documentation



Storybook



Verdaccio

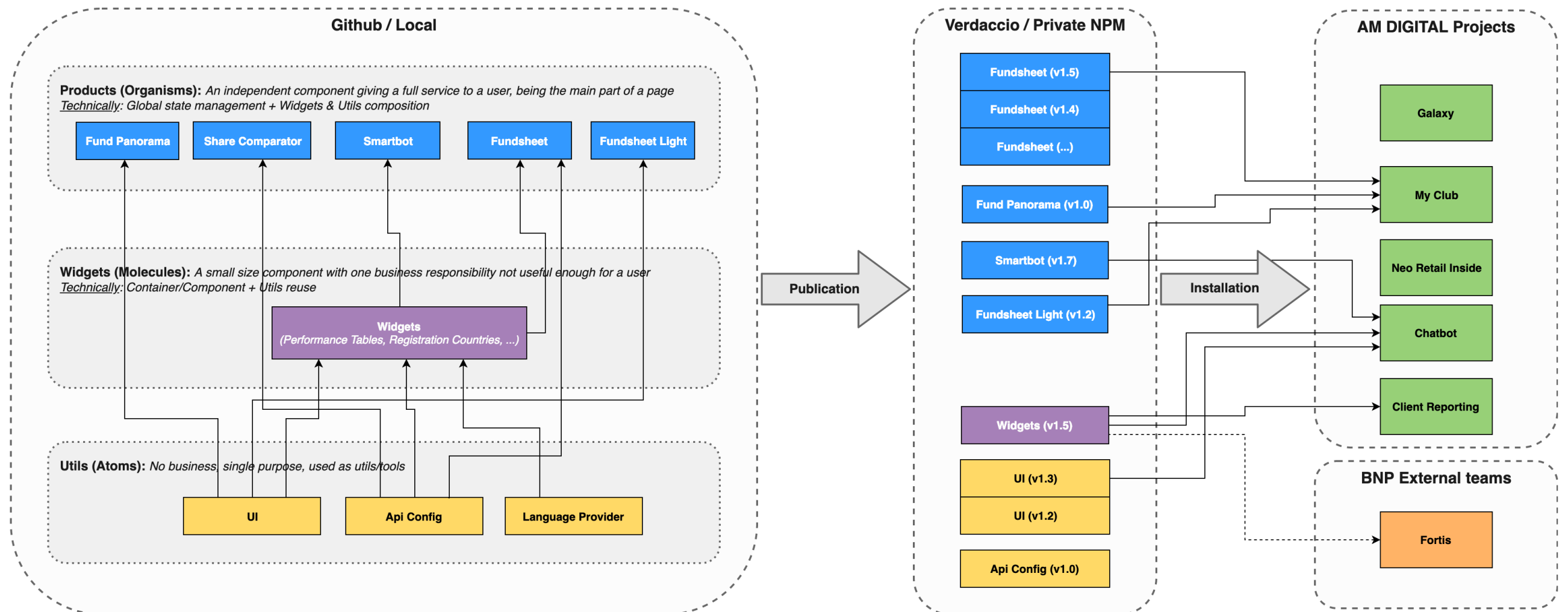


docz

Global Architecture

Global Component Studio Architecture

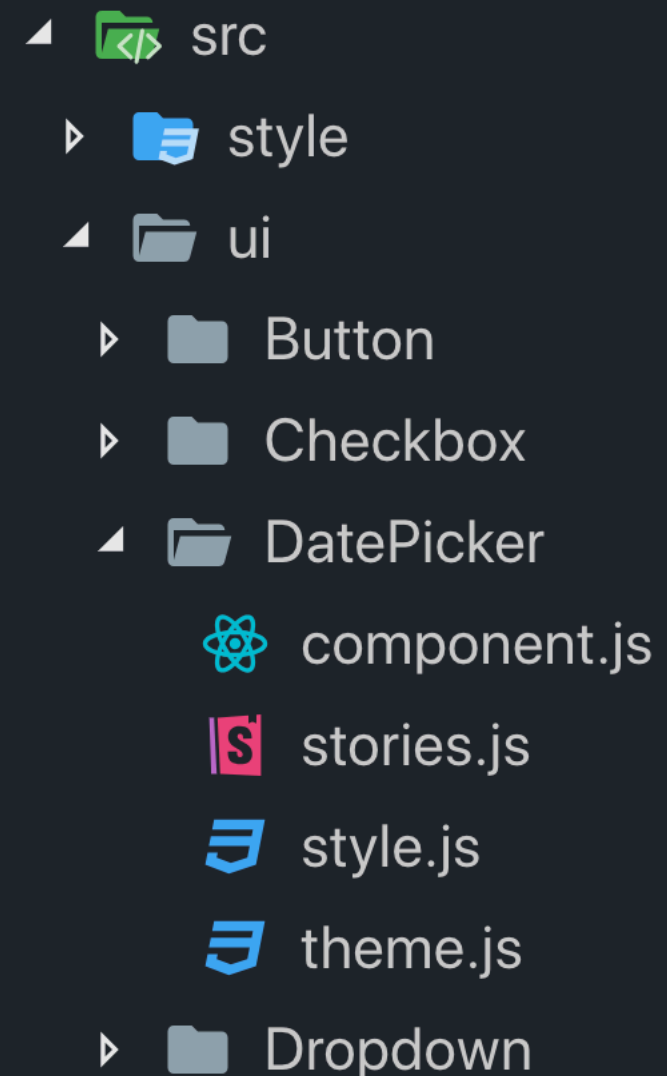
This diagram is a broad representation of Component Studio architecture, it does not show every links & uses between packages



How did we define an atom?

Our UI components:

- Extremely composable
- Highly customisable
- Not connected
- Not related to the business



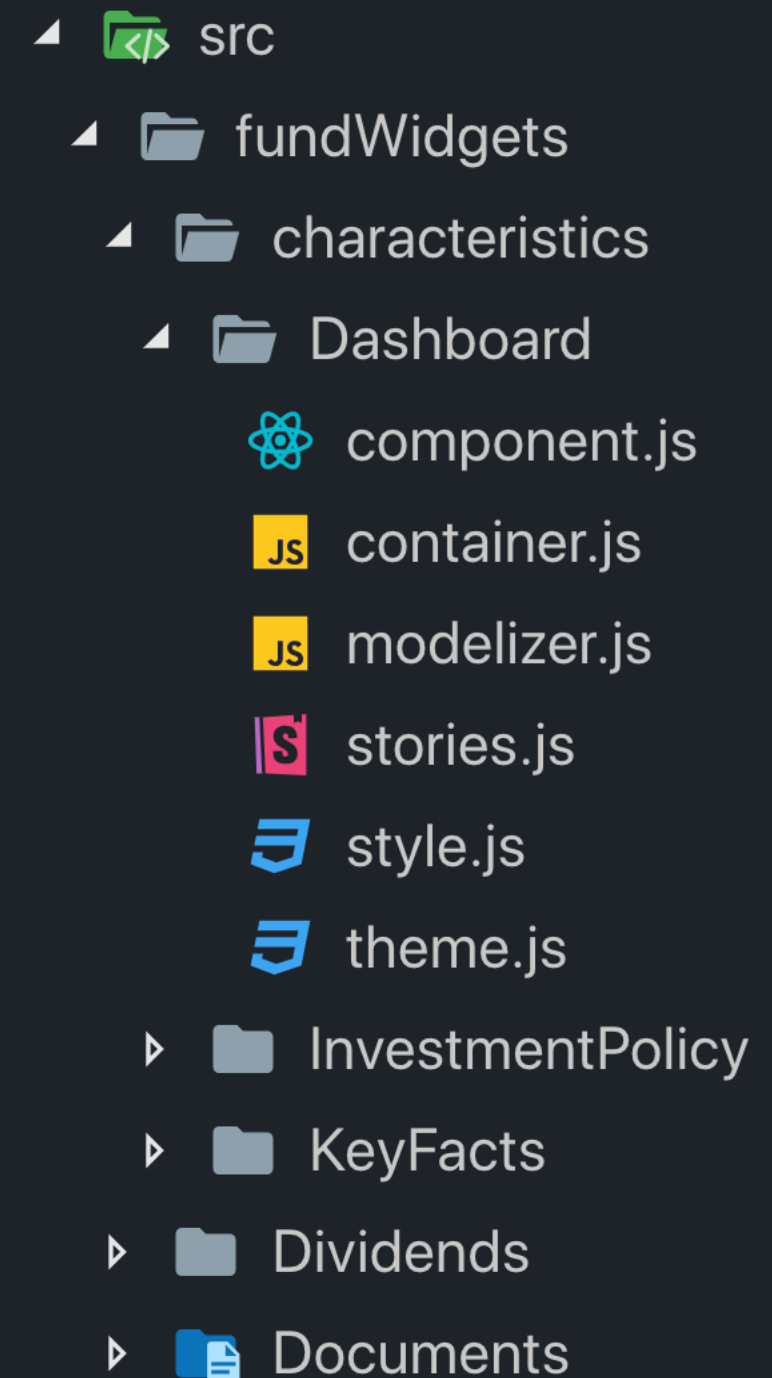
```
src
├── style
├── ui
│   ├── Button
│   ├── Checkbox
│   └── DatePicker
│       ├── component.js
│       ├── stories.js
│       ├── style.js
│       └── theme.js
└── Dropdown
```

The image shows a file explorer interface with a dark background and a blue border. It displays a directory structure starting with 'src'. Under 'src', there are three main folders: 'style', 'ui', and 'Dropdown'. The 'ui' folder is expanded, showing sub-folders 'Button', 'Checkbox', and 'DatePicker'. The 'DatePicker' folder is further expanded, revealing four files: 'component.js' (with a React logo icon), 'stories.js' (with a Storybook 'S' icon), 'style.js' (with a CSS icon), and 'theme.js' (with a CSS icon).

How did we define a molecule?

Our Widgets components:

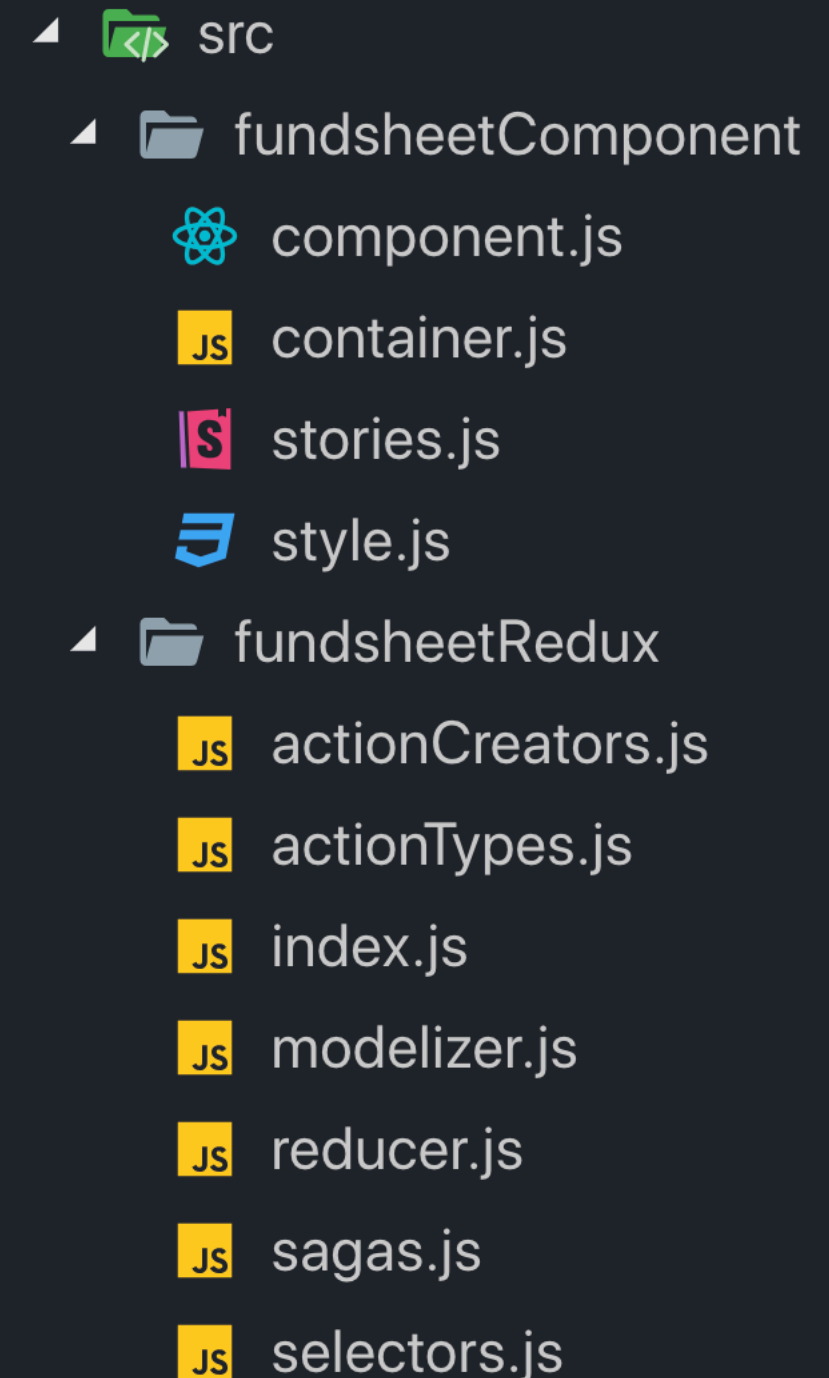
- Composable per business scope
- Reasonably customisable
- Represents one business part
- Has little to no value by itself for the final user
- Is composed of UI components
- Exported dumb or connected



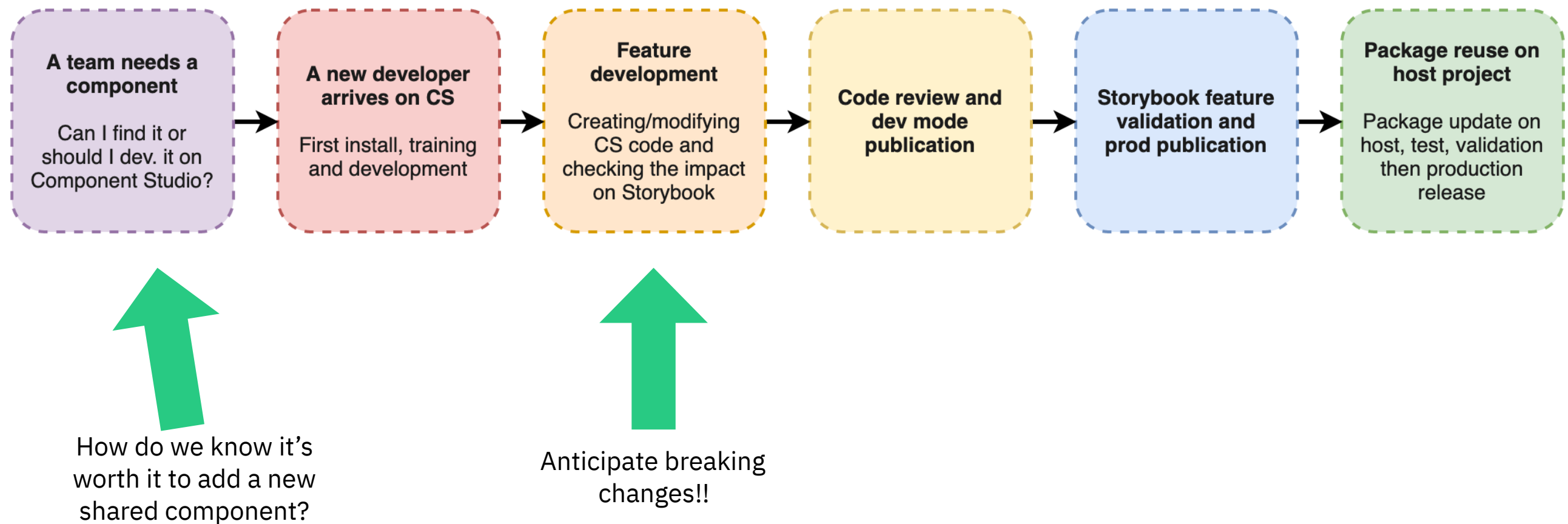
How did we define an organism?

Our Products components:

- Connected full-fledged feature
- Satisfy a user need
- Usually takes most of the page space
- Not composable
- Not customisable
- Is composed of Redux + Widgets and UI components
- Exported as a standalone or in two pieces Redux & Component



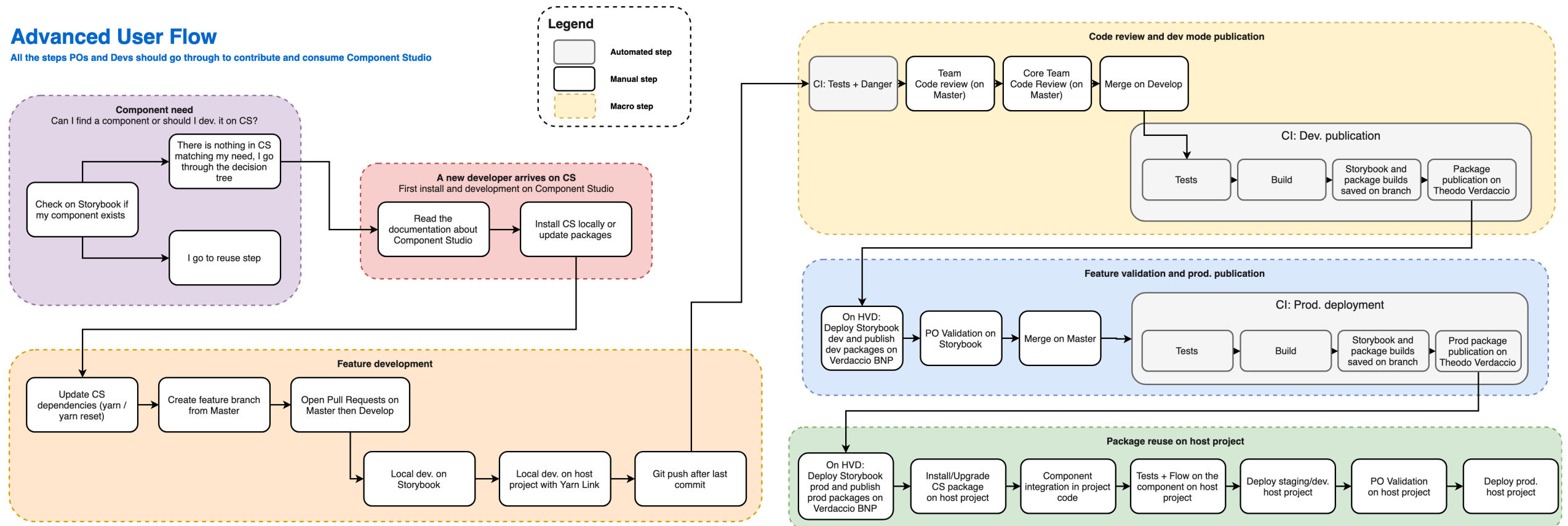
And this is our development flow



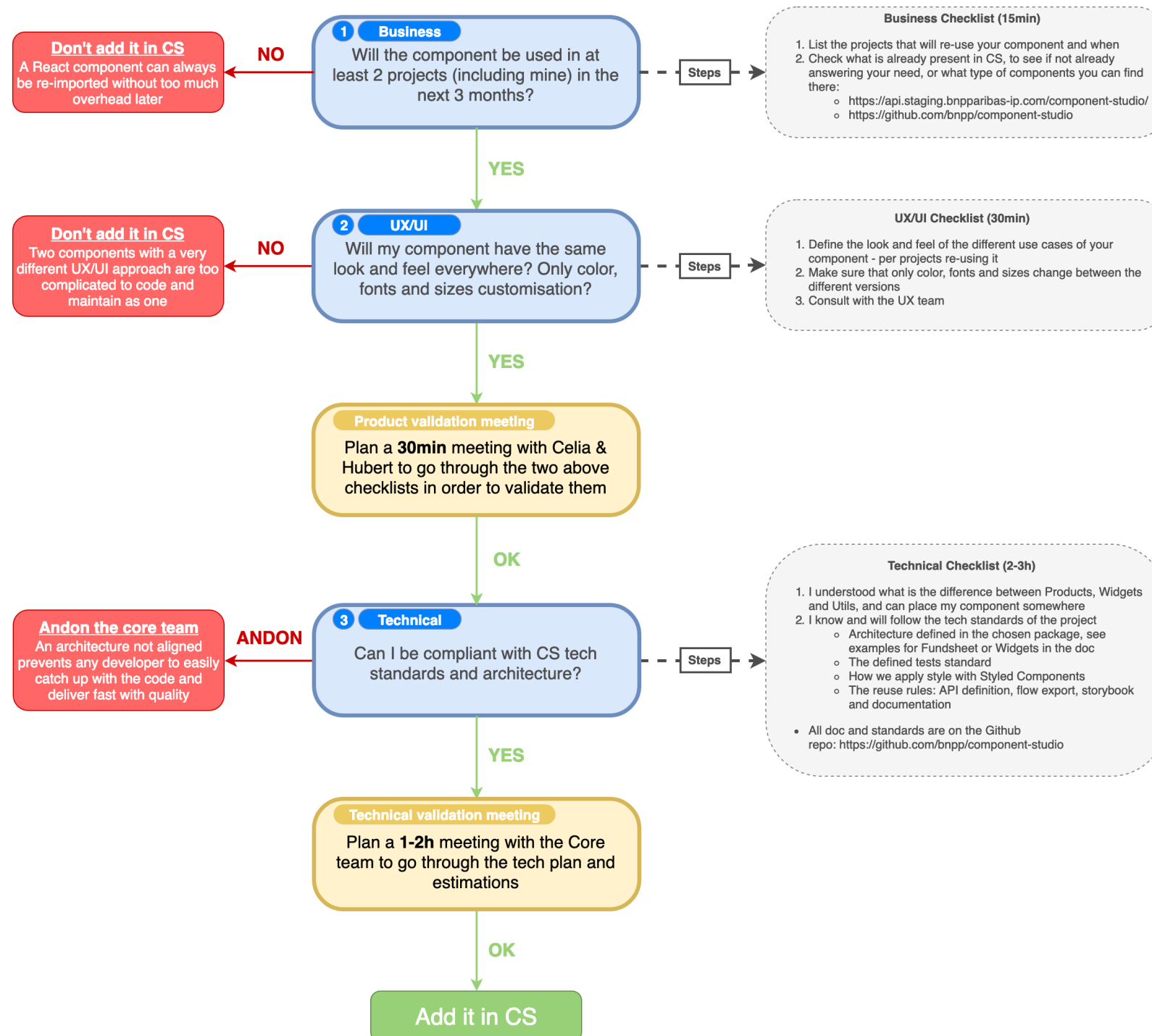
And this is our advanced development flow 🤯

Advanced User Flow

All the steps POs and Devs should go through to contribute and consume Component Studio



Know when to add a new component!



Our current challenges

- Better onboarding and documentation
- Yarn Link
- Better Flow coverage
- Re-definition of our testing strategy
- Performance - slimmer packages

Finding other design systems

A website listing most design systems: <https://adele.uxpin.co>

Top design systems:

- Pinterest - Gestalt: <https://pinterest.github.io/gestalt>
- Ant design: <https://ant.design>
- Palentir - Blueprint: <https://blueprintjs.com>
- Segment.io - Evergreen: <https://evergreen.segment.com>
- Telerik - Kendo UI: <https://www.telerik.com/kendo-ui>
- Element (Vue): <https://element.eleme.io>
- Argon: <https://demos.creative-tim.com/vue-argon-design-system>

Links

- Atomic design: <http://bradfrost.com/blog/post/atomic-web-design>