Building an Enterprise

Design System

for 2024



Brion Silva

Associate Technical Lead at WSO2

github.com/brionmario

brion@wso2.com

Today’s Lineup

- 📝 What is a Design System?
- ❓ Does Your Company Need One?
- ⚙️ How to Implement a Design System
- 🌟 How WSO2 Built a Scalable Design System





Design System?





What is a **Design System**?

An architectural **framework** that consists of:

- Design Guidelines 📐
- Foundations 🔥
- UI Components 💅
- Patterns ✨
- Documentation 📚

Benefits of a **Design System**?

- **Single Source of Truth**
  - Centralizes UI components and guidelines.
- **Promotes Consistency**
  - Ensures brand and user experience alignment.
- **Boosts Efficiency**
  - Streamlines design and development workflows.
- **Enables Scalability**
  - Adapts to growing business needs.

Does Your Company Need a

Design System

Ask yourself these questions

Is you company:

✅ Managing multiple products or platforms

✅ Have large, cross-functional teams

✅ Has products with inconsistent UIs

✅ Has enough resources

✅ Only manage small projects

Reconsider

✅ A Startup

Reconsider

How to implement a
Design System...

Assess your **needs**

- What are the **current inconsistencies** in our UI?
- How do teams currently **collaborate on design** and **development**?
- What are the most frequent **design issues** or **bottlenecks** encountered?

Define the **scope**

- Which products or features **share common UI** elements?
- Are there specific **teams that would benefit** most from a design system?
- What is the **timeline** for implementing the design system?

Create a cross-functional **team**

- **Designers & UX Engineers**
  - Guides the visual direction and component design.
- **Developers**
  - Ensures that components are implemented correctly and efficiently.
- **Quality Assurance Engineers**
  - Makes sure that the system is functional and fulfills the needs outlined.
- **Content Creators**
  - Creates clear, comprehensive documentation for users.

Establish design **principles**



- **User-Centricity**
  - Prioritize the needs of your users in every design decision.
- **Consistency**
  - Strive for uniformity in design elements.
- **Scalability**
  - Ensure that the design system can grow and adapt.
- **Accessibility**
  - Make design choices that enhance accessibility.

Develop the **design:** Define a Visual Language

- Color palettes
- Typography
- Spacing
- etc





Inter Typeface

H1	**56**	**The quick brown fox**
H2	36	The quick brown fox
H3	28	The quick brown fox
Body	16	The quick brown fox
Caption	12	The quick brown fox

Develop the **design:** Design Tokens

⭐ Highly Recommended

Raw Value

`#23a06b`

↓

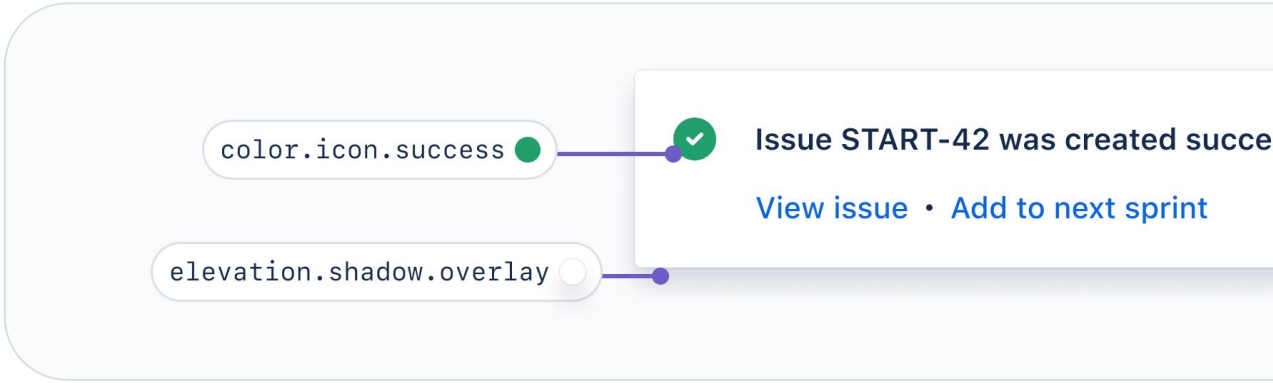
Base Token

`Green800`

↓

Design Token

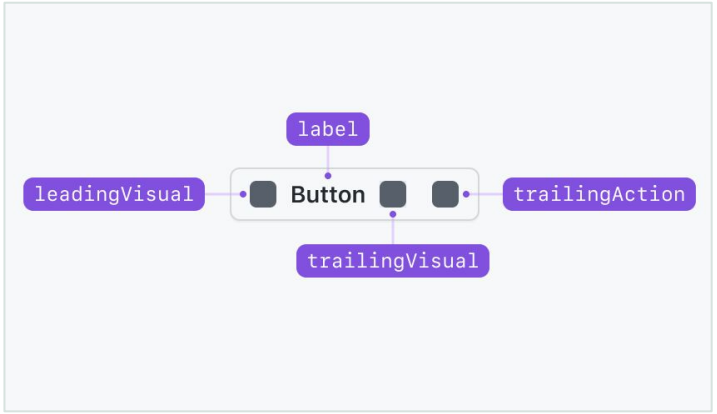
`color.icon.success`



Develop the **design:** Design the Components

Example: Button 🎨

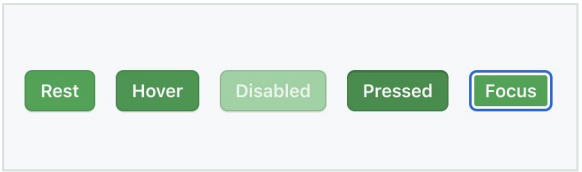
Anatomy



Variants



States



Start **implementing:** From scratch

- 👍 Allows full control over the design, behavior, and customization.
- 👎 Will take forever to implement
- 👎 Very labor-intensive
- 👎 Require a significant maintenance effort

Start **implementing:** Wrap existing libraries

- 👍 Saves time.
- 👍 Can inherit styling and customize the theme as needed.
- 👍 OOTB Accessibility and performance.
- 👍 Can quickly integrate.
- 👎 Less freedom compared to implementing from scratch.

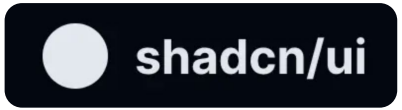






Start **implementing:** Using a component collection

- 👍 Increasingly popular choice for design system implementation.
- 👍 Promotes a copy-and-paste strategy.
- 👍 Very easy to get started.
- 👍 Uses established styling frameworks & primitives like Tailwind & Radix.







Start **implementing:** Build on top of primitives

- 👍 Offer you the full control over styling
- 👍 Relatively easy to implement compared to from the scratch implementation.
- 👍 Can inherit functionality, accessibility & performance from the primitives.
- 👎 Need expert UI engineers to style.







Create **Documentation**

Good documentation should cover:

- Design Principles 📐
- Component Usage 🤔
- Code Snippets 👨‍💻

Can utilize tools like 

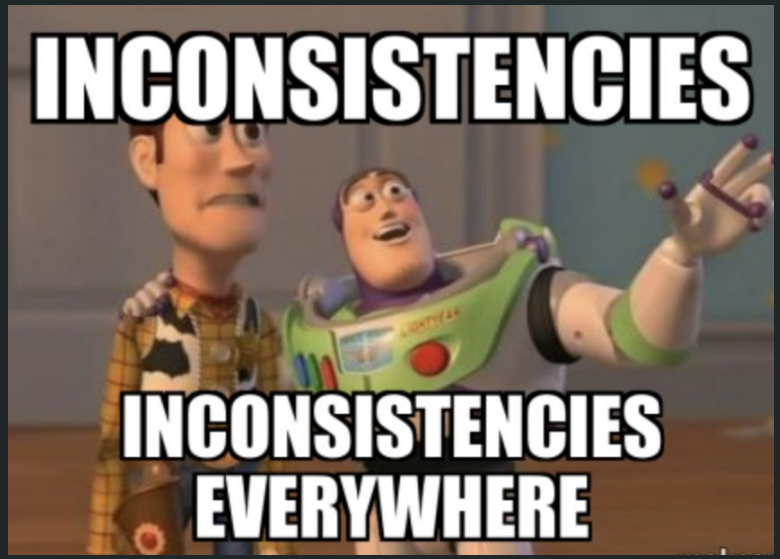
How **WSO2** Built a Scalable **Design System**

The story of OXYGEN UI

github.com/wso2/oxygen-ui

Why?

Inconsistencies across features in the same product



Why?

Design & Implementation was not in sync.



Why?

Different teams were using different UI libraries.



Why?

There were limitations in the UI frameworks

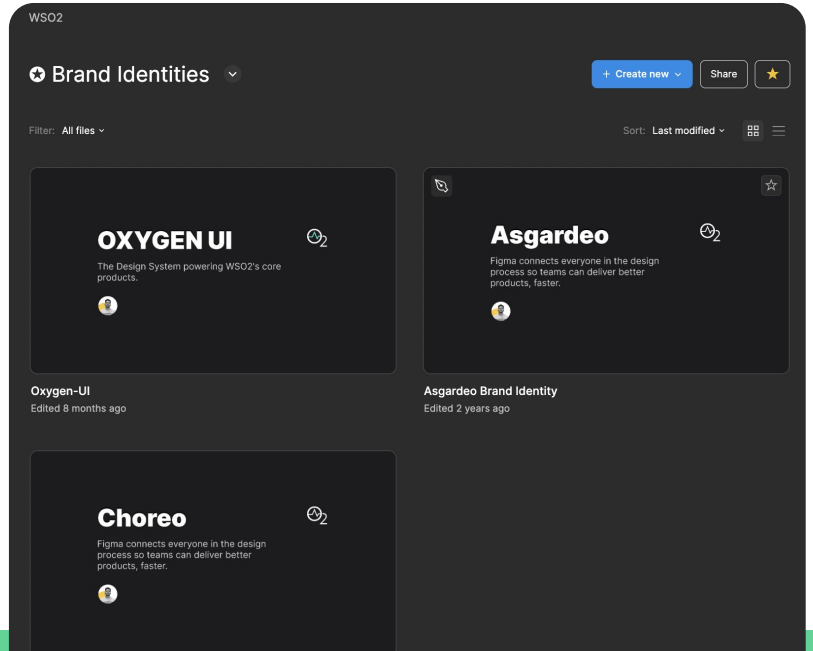


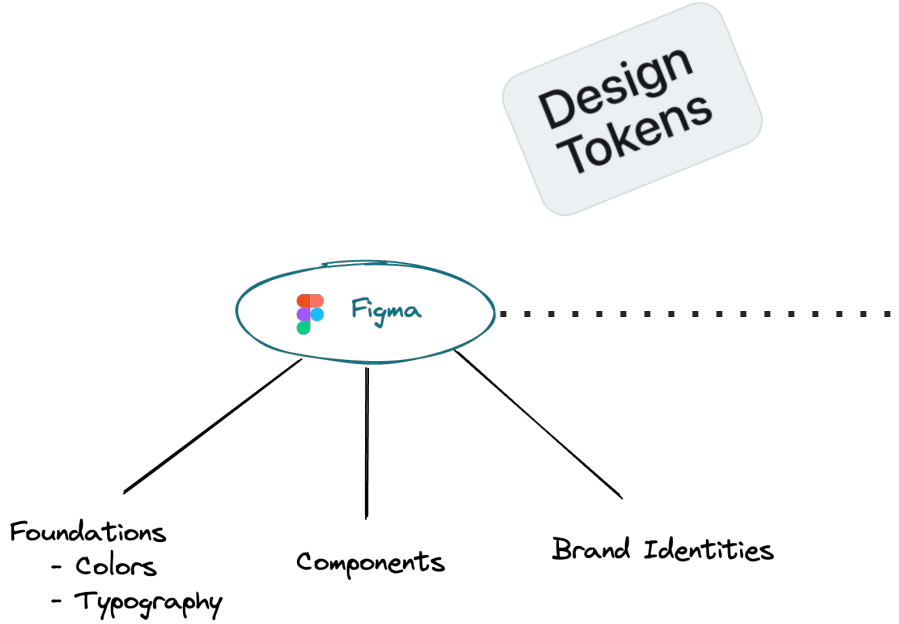
What we wanted?

- Design First Development
- Scalable Foundation
- Sync Design & Development
- Task Automation
- Less maintenance effort
- Should benefit all the teams

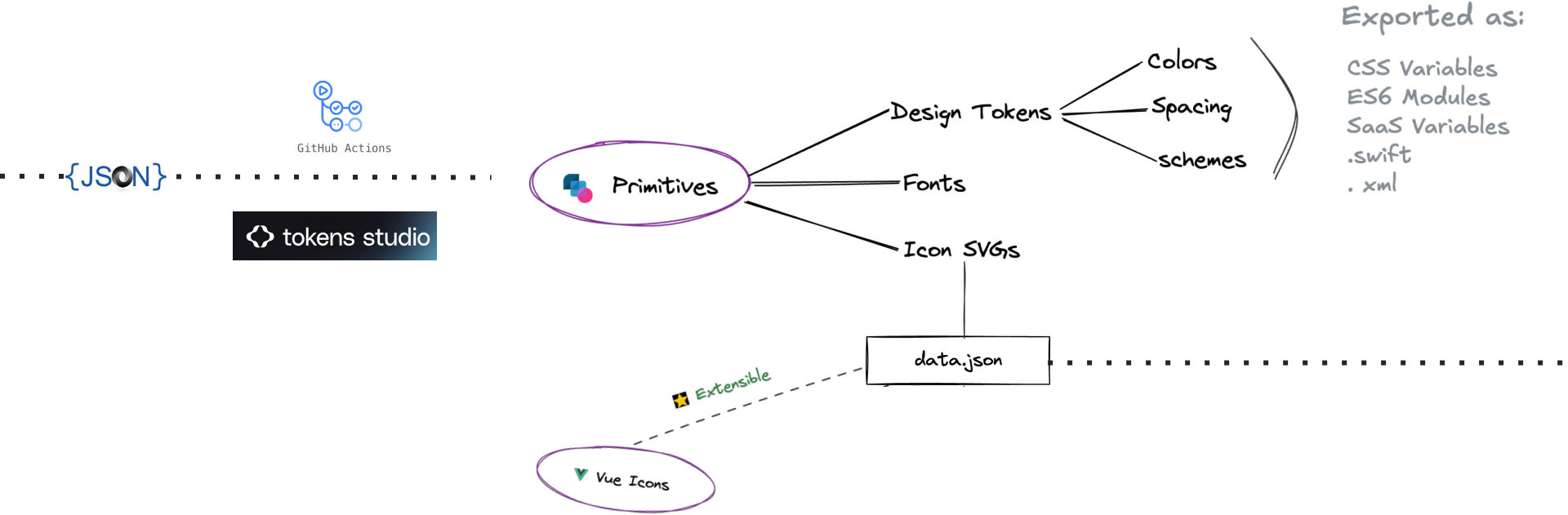
Architecture of **Oxygen UI:** Figma Design 🎨

- Single source of truth

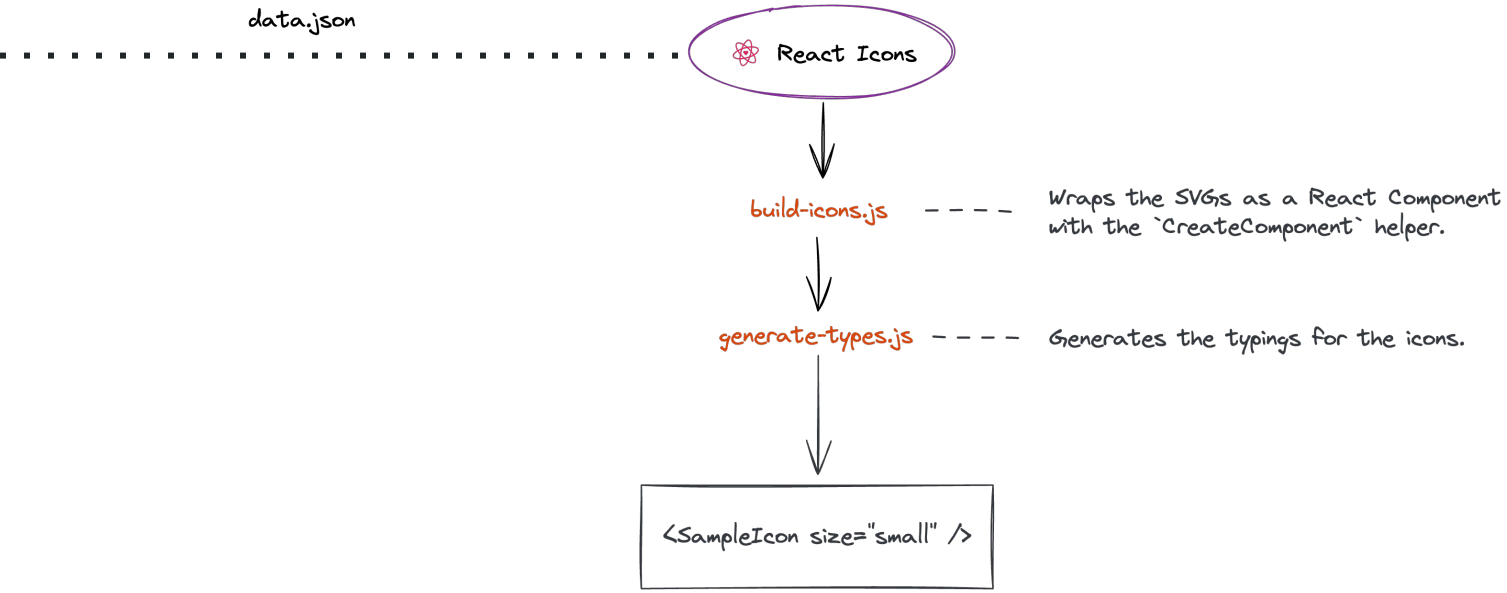




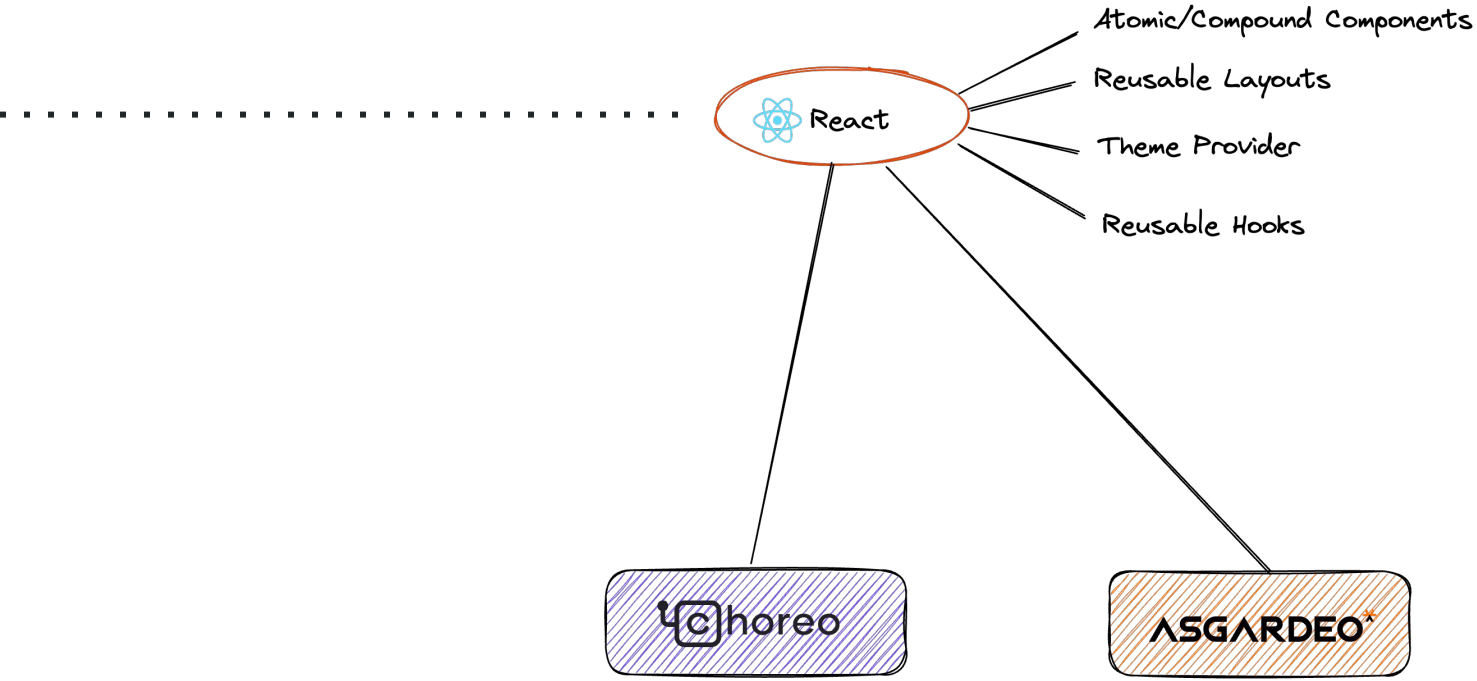
Architecture of **Oxygen UI:** Primitives 🔥



Architecture of **Oxygen UI:** React Icons 🔎



Architecture of **Oxygen UI:** React



Architecture of **Oxygen UI:** Recap

Tech Stack

- Design
  - Platform -> Figma
  - Design Token Manager -> tokens studio
- Primitives
  - Token Transformer -> Style Dictionary
- React
  - Base UI Framework -> MUI
- Documentation
  - Base Site -> NEXT.js
  - React Components -> Storybook

Published packages npm

- @oxygen-ui/primitives
- @oxygen-ui/react
- @oxygen-ui/react-icons

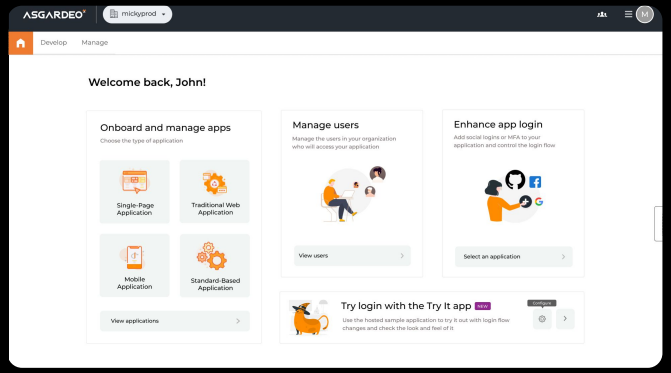
Implementation of **Oxygen UI**

wso2.github.io/oxygen-ui

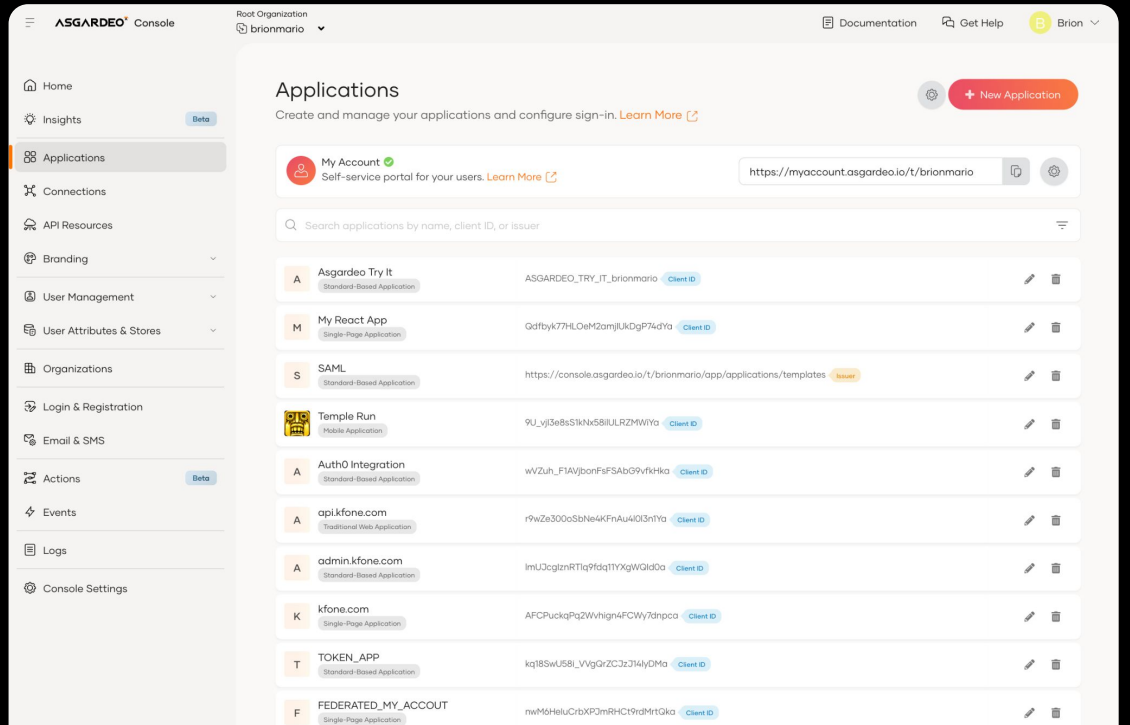


Oxygen UI in ASGARDEO

Before



After



Recap

- Assess the needs well 🤔
- Don’t Default to Starting from Scratch 🛠️
- Get Inspired 💡
- Maintenance is Key 🔧
- Embrace coexistence 🤗

Q & A

THANK YOU FOR COMING 🙏

Feedback



References:

- Atlassian Design System
- Shopify Polaris
- GitHub Primer
- Google Material Design
- IBM Carbon