Embrace the runtime

How the BEAM helps me build my software+services solution.

What's alzo ?

Data :
- Archive of EVERYTHING an architecture agency did
- Tools for searching and organizing

Tools :
- Document (call for offers, communication) production apps
- Websites
- Integrations with software suites (MS Office, Adobe's creative suite)
- ...

Phoenix + LiveView (+ Vue)

alzo

Accueil / Projets

Outils
- Intégrations
- Importer des données
- Applications
- Contribuer
- Suivi BDD

Données
- R&D
- Documents

Gestion
- Options d'instance
- LiveDashboard
- Oban Web
- Utilisateurs

Classement
- File d'attente
- Critères de tri
- Critères qualitatifs
- Critères quantifiables

Taxonomies
- Types de dates
- Types de surfaces
- Types de montants

Agence
- Structures
- Équipe
- Métiers
- Publications et prix

Entreprises
- Annuaire
- Rôles projet

lucas@lucassifoni.info admin

déconnexion

Version 0.9.98-stable

Parcours des images et projets

Recherche

(●) Recherche simple ( ) Par critères qualitatifs ( ) Recherche avancée

Affichage

(●) En liste ( ) En images ( ) Sur une carte

Filtre rapide par code ×

Filtre rapide par titre ×

Recherche libre : tags, titres, détails archi, prestataires... ×

☐ N'afficher que les projets présents sur le site.

Vignette	Dernier document	Caractéristiques	Actions
		**24_EHSJ / Extension de l'Hôpital Saint-Jacques** Le Mans (72)	Médico social · Hopital : Gériatrie · Hopital: MCO
		**23_CSC / Complexe Sportif des Collines** Rennes (35)	
		**23_PNM / Passerelle du Nouveau Monde** Nîmes (30)	Chantier en cours · CCP · Neuf · Ouvrage d'art · Mobilités douces

Créer un nouveau projet

Document production apps / websites

Not whitelabel templates. Custom developed for each client
- Adapt to existing practices and vocabulary
- Truly adapt to graphic design guidelines
- Adapt interactivity and behavior levels

fully automatic
no interaction
background generation

manual layout
design rules are the program
real-time collaboration

Fiche A4

Centre Culturel des Arts Modernes

Centre Culturel des Arts Modernes

Marseille (13)

Robo-MO

1000m² SDP

10000000€ Travaux HT

Public, Mission de base

Le projet consiste en la construction d'un centre culturel moderne qui inclura des galeries d'exposition,

Équipe

Générer le document

Centre Culturel des Arts Modernes

Marseille (13)

Le projet consiste en la construction d'un centre culturel moderne qui inclura des galeries d'exposition, des ateliers pour artistes, des espaces de performance et un café, contribuant ainsi au dynamisme culturel.

Maître d'ouvrage
Robo-MO

Type de mission
Public, Mission de base

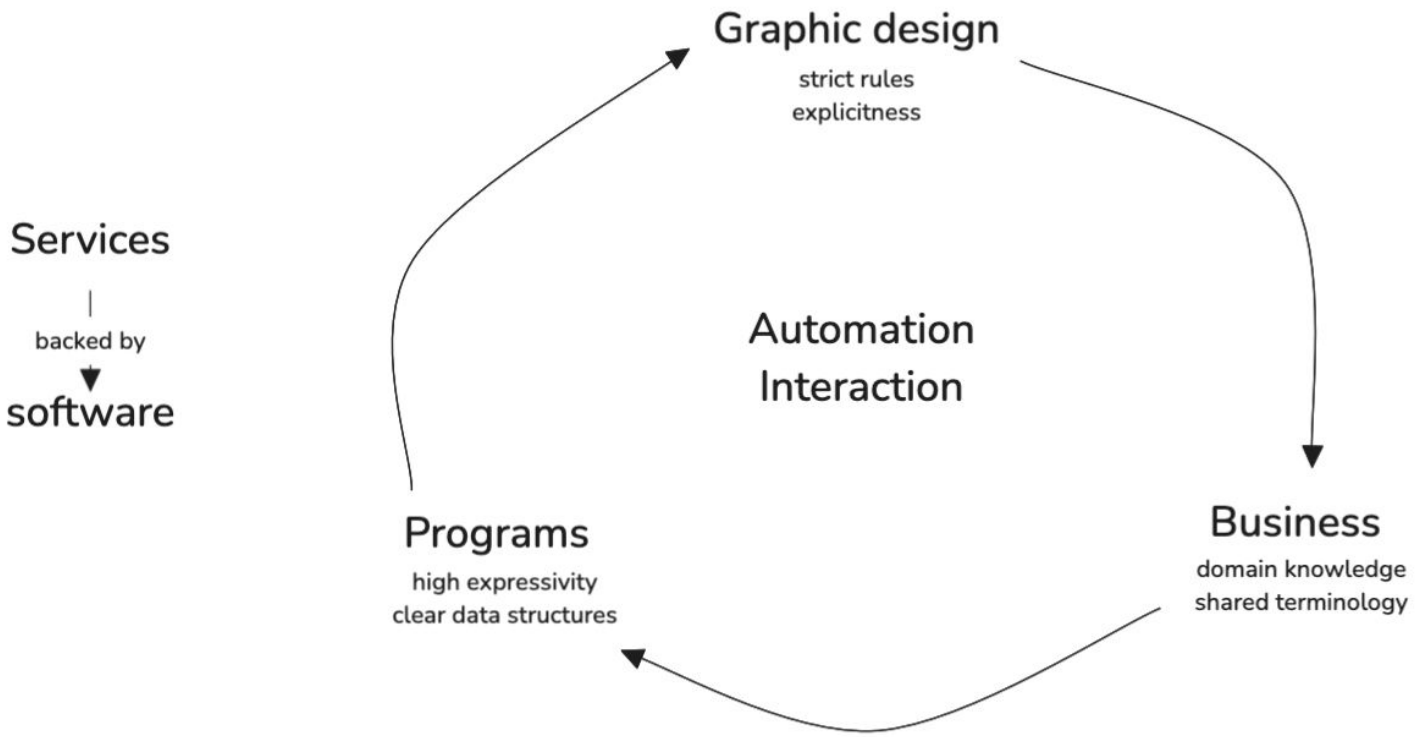
Surface
1000m² SDP

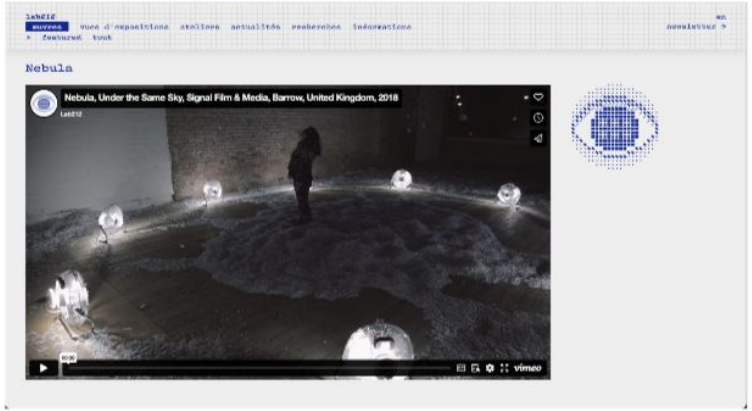
Équipe

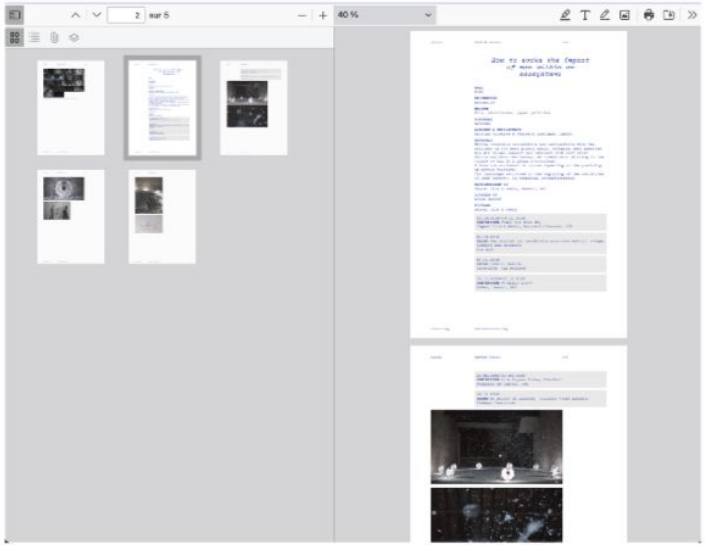
Coût
10000000€ Travaux HT

alzo

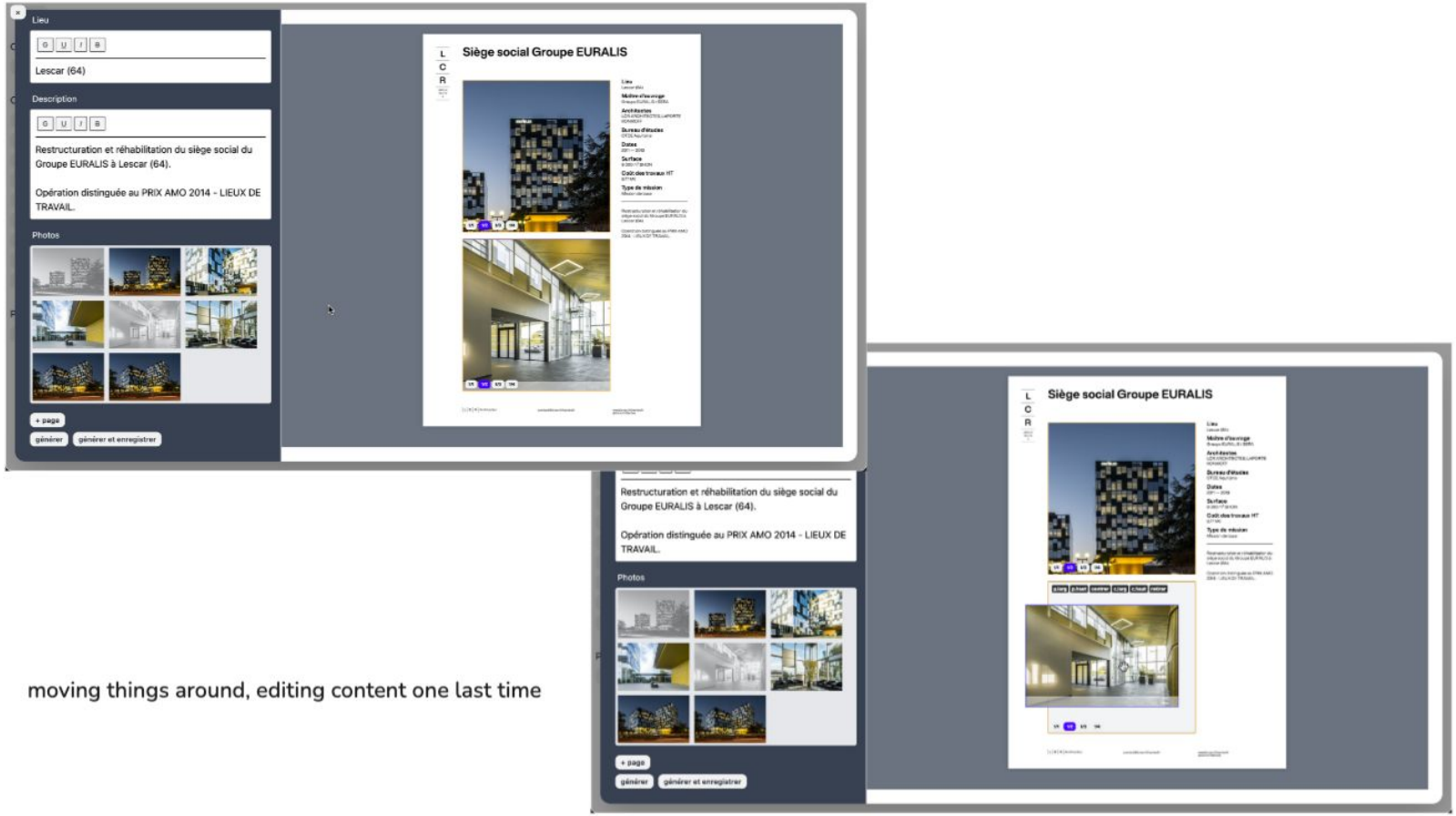
contact@alzo.archi
www.alzo.archi


Graphic design
strict rules
explicitness
Services
backed by
software
Automation
Interaction
Programs
high expressivity
clear data structures
Business
domain knowledge
shared terminology

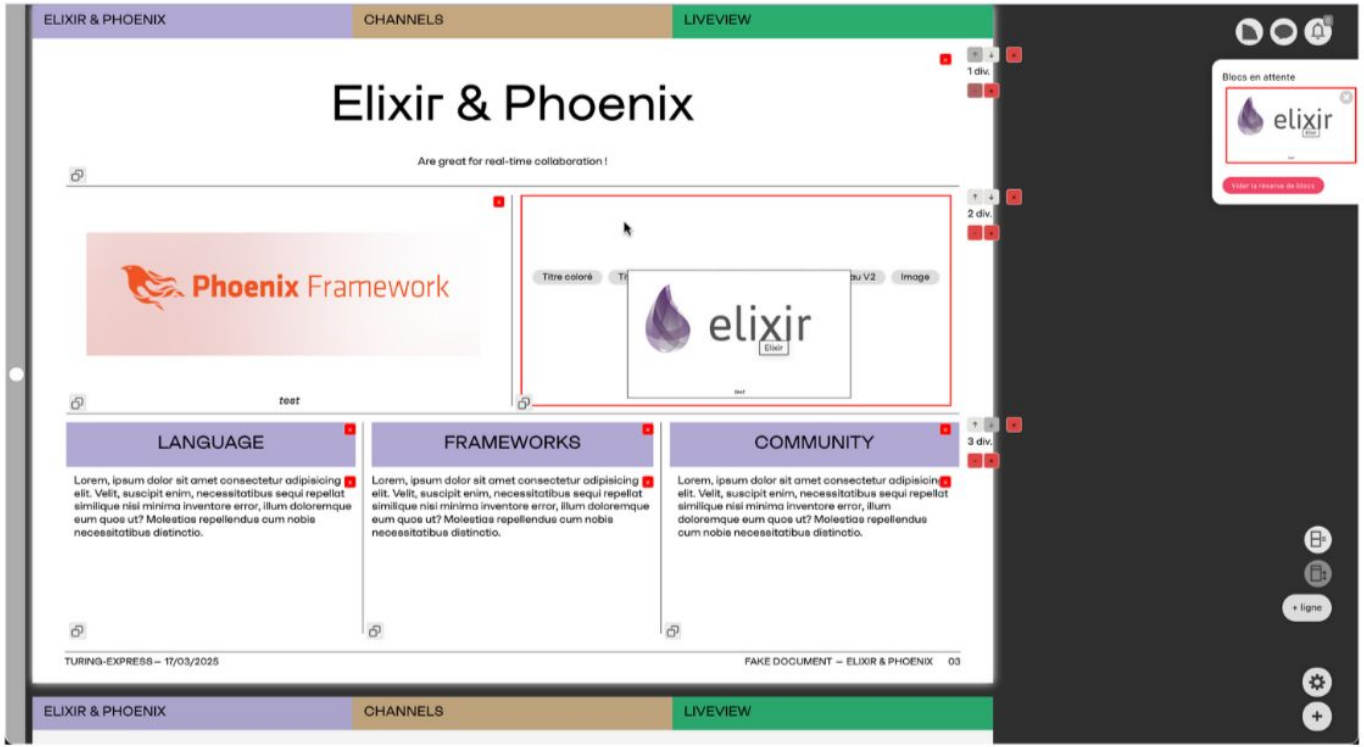




straight data to templates



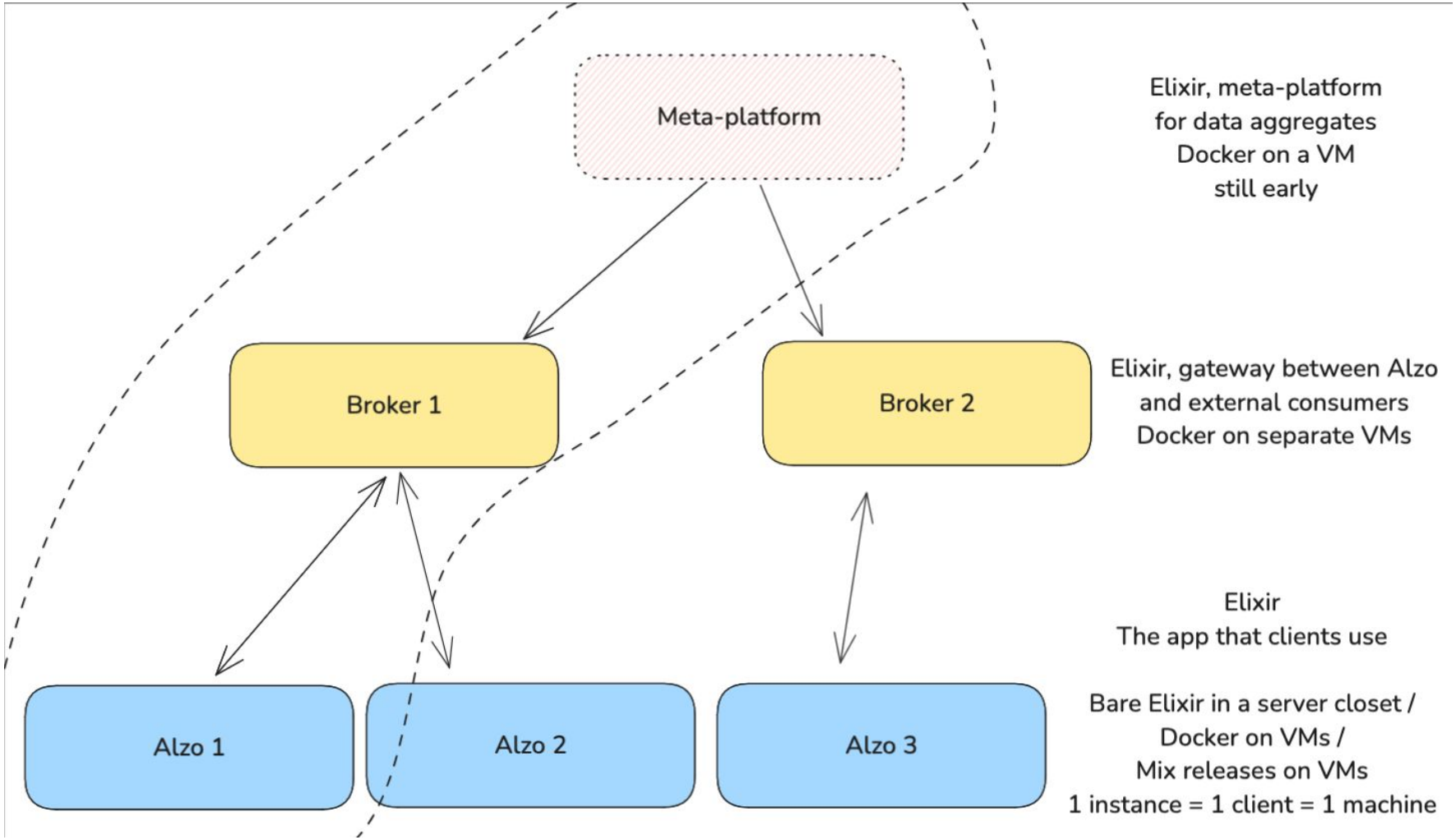
moving things around, editing content one last time

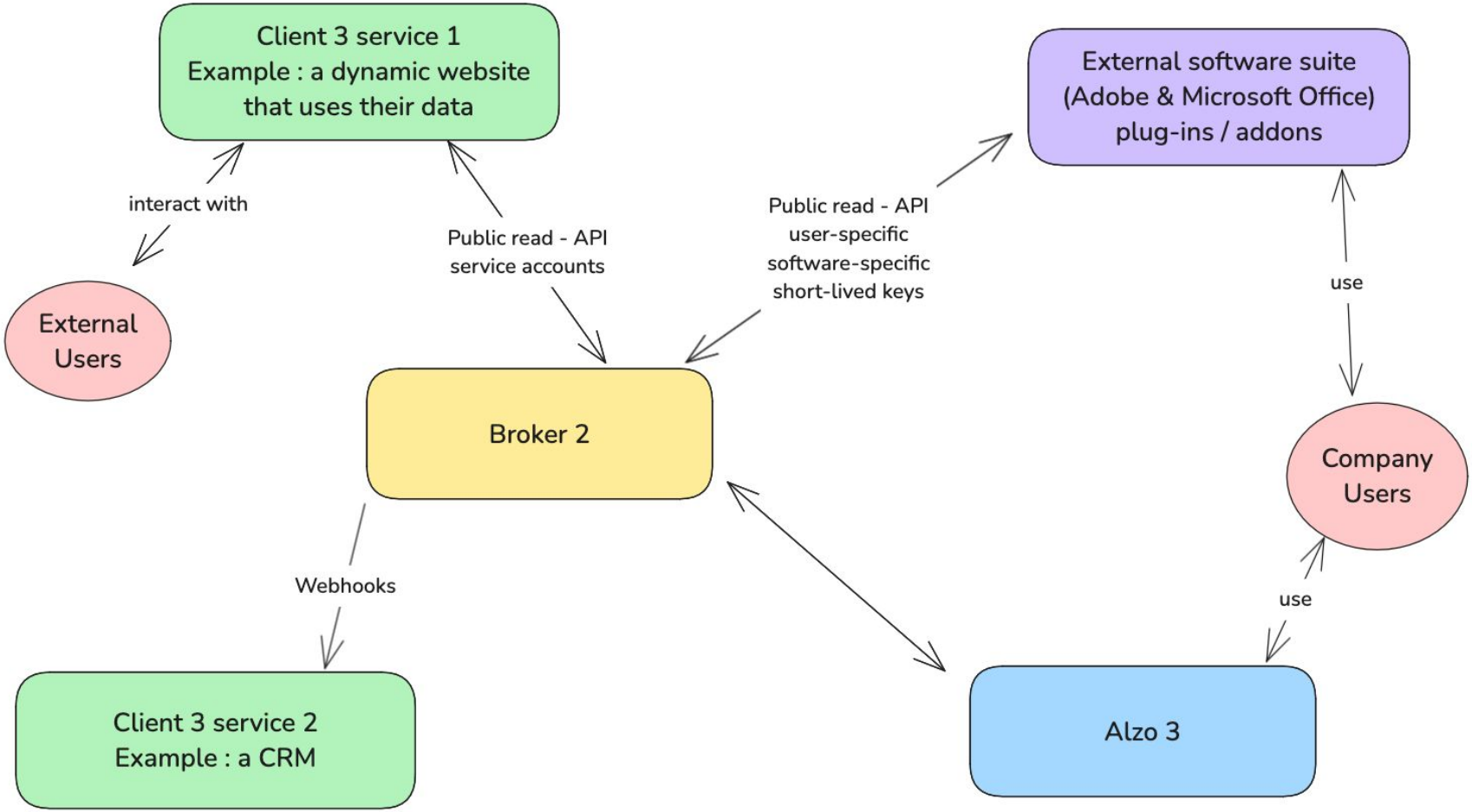


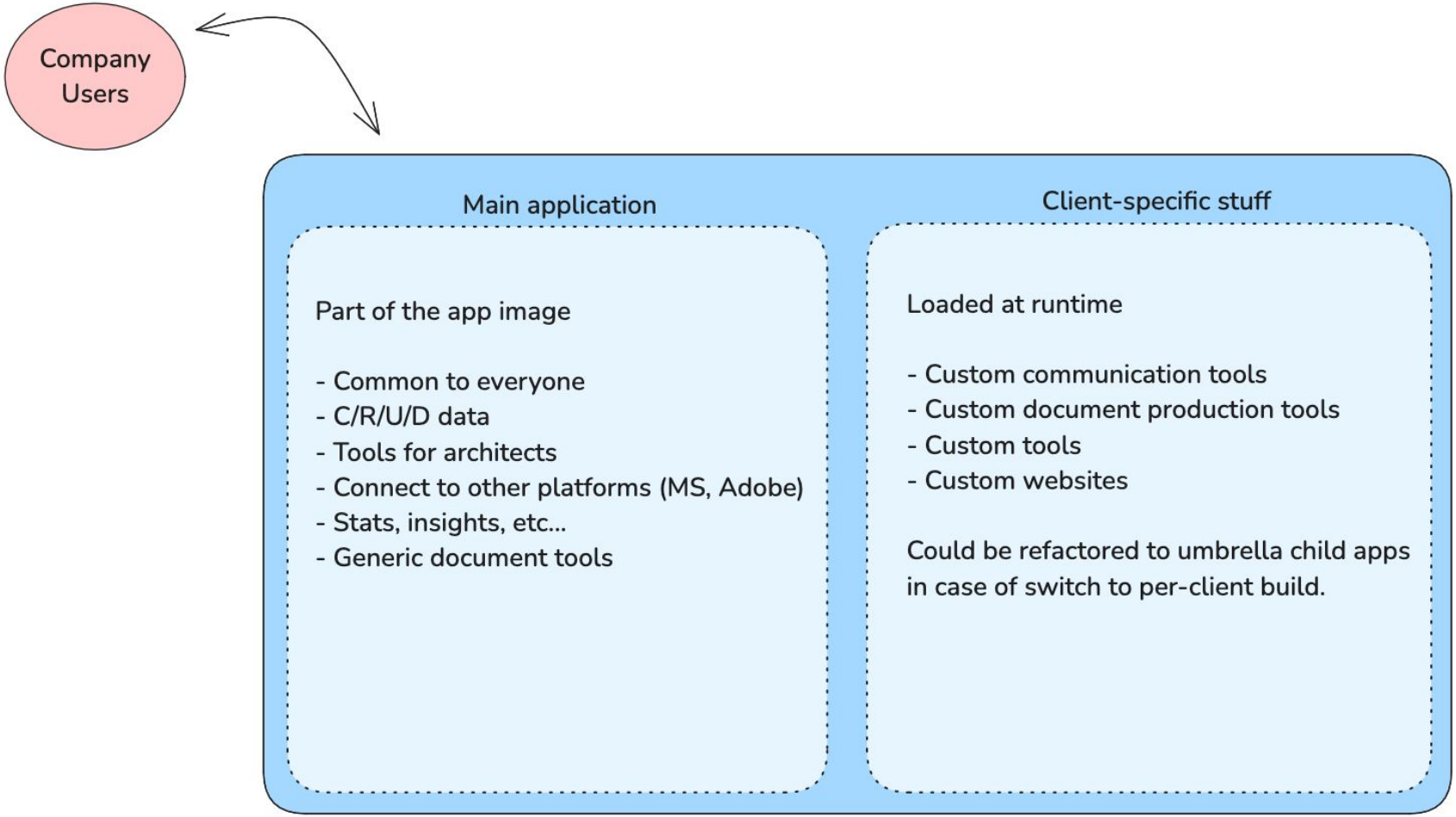
multiple people collaborate in real-time on building long chaptered documents
(not about elixir but you get the idea)

You're at *super* low scale. You could use anything. Why Elixir ?

- long-running processes in supervision trees
- orchestration of third party stuff
- hot code loading
- introspectability
- massive concurrency for coordination services
- stack consolidation
- runtime behavioral changes
- personal preference

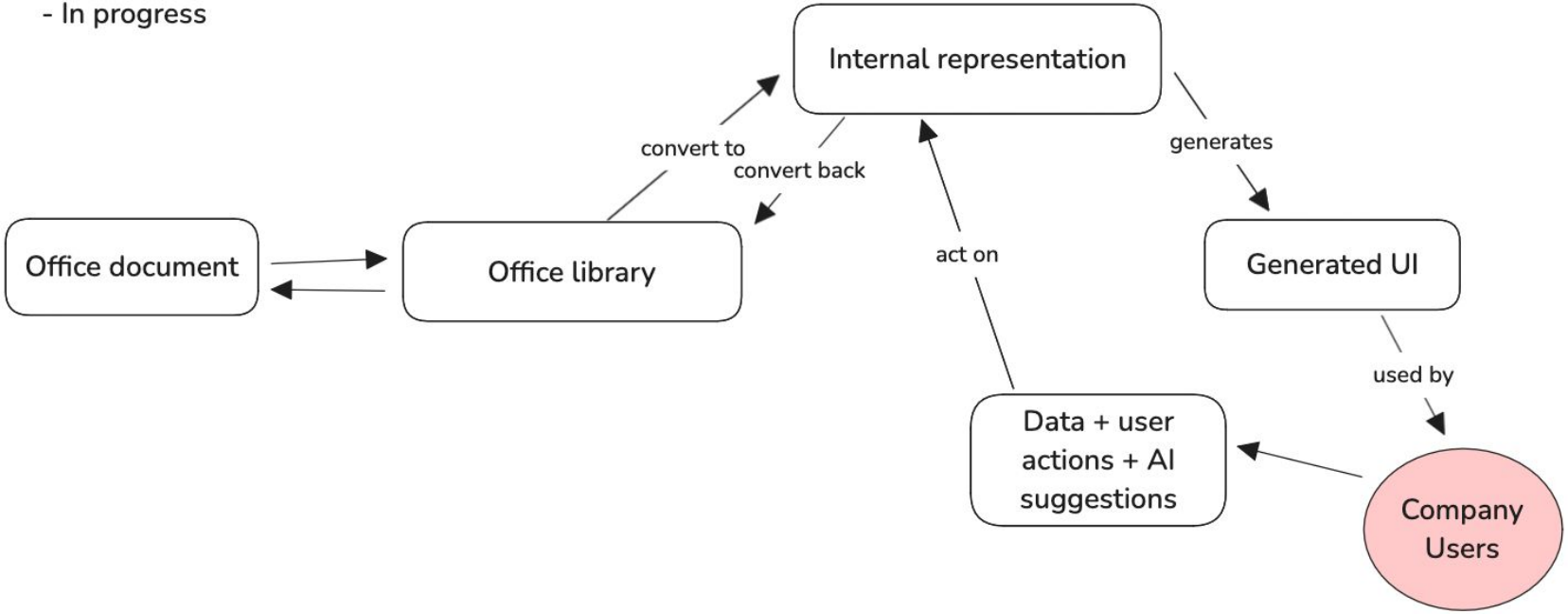

Meta-platform
Elixir, meta-platform
for data aggregates
Docker on a VM
still early
Broker 1
Broker 2
Elixir, gateway between Alzo
and external consumers
Docker on separate VMs
Elixir
The app that clients use
Alzo 1
Alzo 2
Alzo 3
Bare Elixir in a server closet /
Docker on VMs /
Mix releases on VMs
1 instance = 1 client = 1 machine


Client 3 service 1
Example : a dynamic website
that uses their data
External software suite
(Adobe & Microsoft Office)
plug-ins / addons
interact with
Public read - API
service accounts
Public read - API
user-specific
software-specific
short-lived keys
use
External
Users
Broker 2
Company
Users
Webhooks
use
Client 3 service 2
Example : a CRM
Alzo 3


Company Users
Main application
Part of the app image
- Common to everyone
- C/R/U/D data
- Tools for architects
- Connect to other platforms (MS, Adobe)
- Stats, insights, etc...
- Generic document tools
Client-specific stuff
Loaded at runtime
- Custom communication tools
- Custom document production tools
- Custom tools
- Custom websites
Could be refactored to umbrella child apps in case of switch to per-client build.

Generic document tools

- Read + Write Word,Excel,Powerpoint documents
- Preferably a single library
- User makes themselves a library of automated MS office documents
- In progress



Idea 1 : leverage PHPOffice

PHPOffice

573 followers The Internet http://github.com/PHPOffice

Pinned

PhpSpreadsheet Public

A pure PHP library for reading and writing spreadsheet files

PHP 13.5k 3.5k

PHPWord Public

A pure PHP library for reading and writing word processing documents

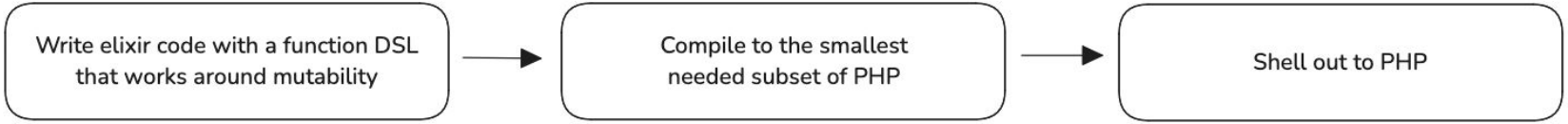
PHP 7.4k 2.7k

PHPPresentation Public

A pure PHP library for reading and writing presentations documents

PHP 1.3k 526

Idea 1 : leverage PHPOffice

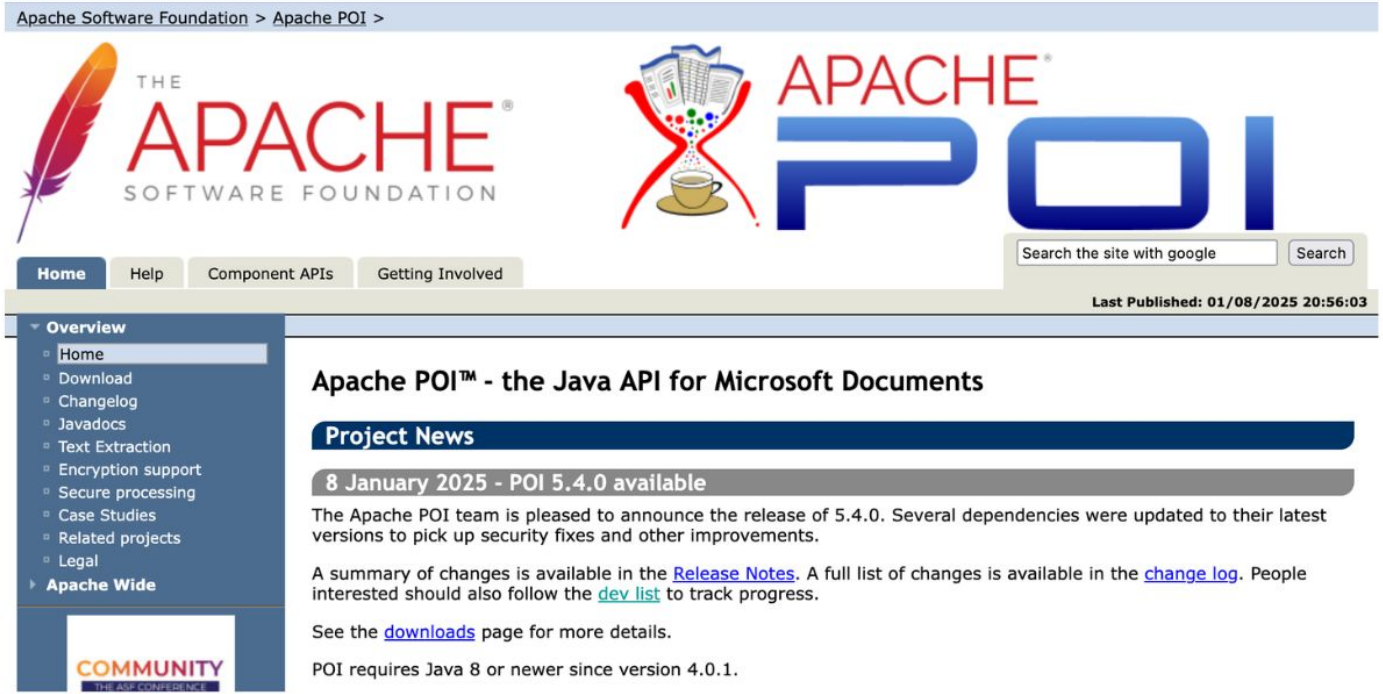


Pros :
- the lib does what it needs to
- no background service

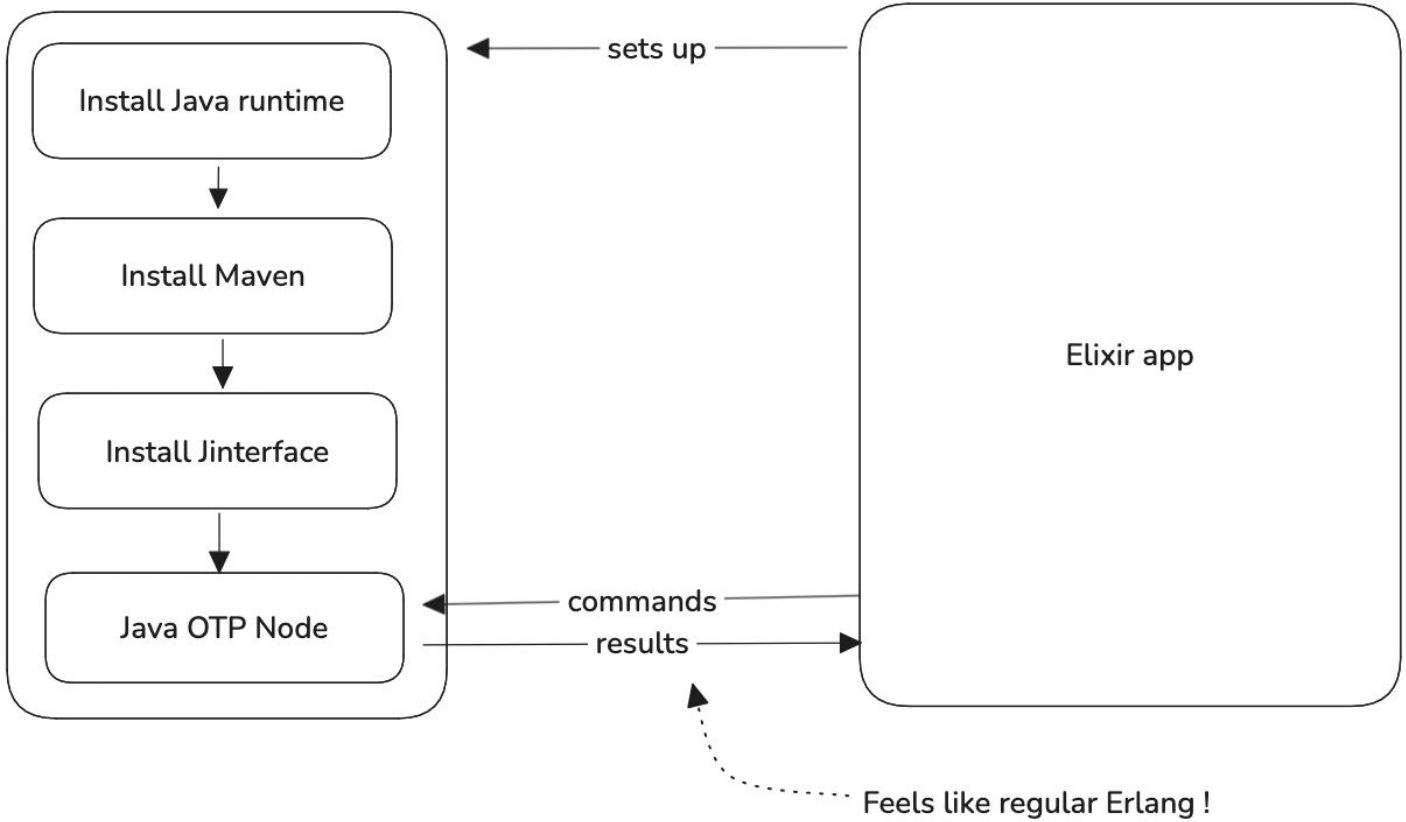
Cons :
- DSL magic
- bound to the library's idioms
- one/off invocations (= no straightforward messaging)
- manage a PHP runtime

Idea 2 : leverage Apache POI

- Java library
- Backed by the Apache foundation
- First release 24 years ago
- Thanks @EleusisT !



Idea 2 : leverage Apache POI



Idea 2 : leverage Apache POI

Pros :
- stable library
- native feel with JInterface
- LLMs know Java & POI *very* well
- very easy to find help
- easy to extract to a separate service if needed

Cons :
- surface of a Java service feels very large
- JInterface considered legacy but unbreakable

Client-specific stuff

Common docker build with only the main app
Client-specific stuff gets hot loaded at startup

Pros :
- single repo
- tested in-app
- easy to refactor common things
  to the main app
- relatively easy to convert to an umbrella
  if I change strategy

Cons :
- not so common today
- a few temporary hacks over liveview

Client-specific stuff : in development

- lib/clients/<clientX>/<appX>
- test/clients/<clientX>/<appX>

DynamicApp behaviour

```
-> tree lib/clients/fooclient/barapp/
|- bar_app_entry.ex
|- components/                    HTML and event handlers
   |-- bar_app_foo_component.ex
   |-- bar_app_baz_component.ex
|- state/                         UI-Pure state
   |-- bar_app_state.ex
|- logic /
   |-- bar_app_logic.ex           UI-Pure logic (might be multiple files)
```

DynamicApps can call the main app. The reverse is of course forbidden.

In CI :

```
rm -rf lib/clients
rm -rf test/clients
```

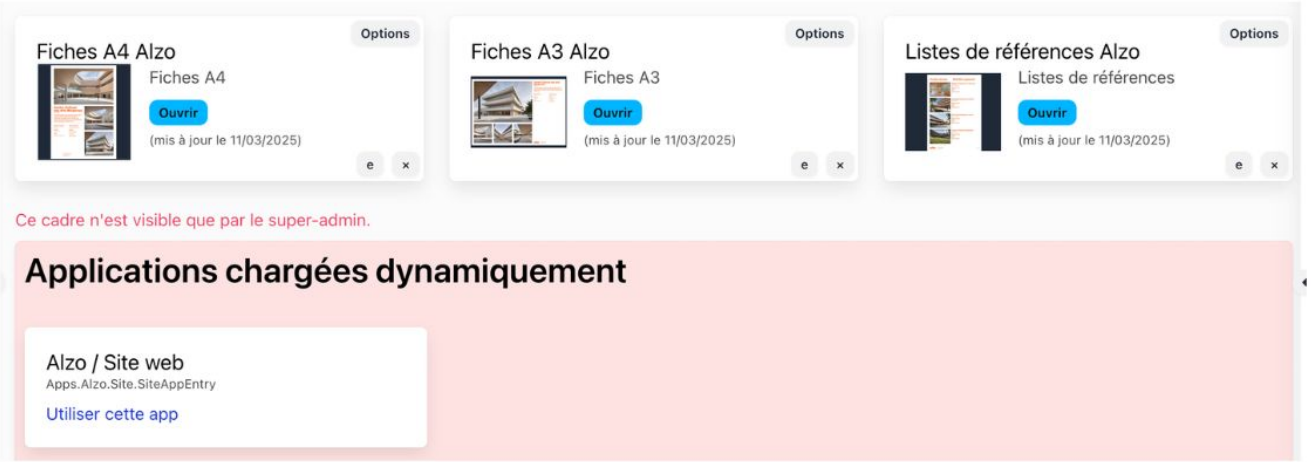
Client-specific stuff : in production

Alzo.Applications.App schema

Super-admin (me) can upload a signed tarball of an application
Available applications have an icon, description, friendly name

Loaded on first open (newly added) or at app startup

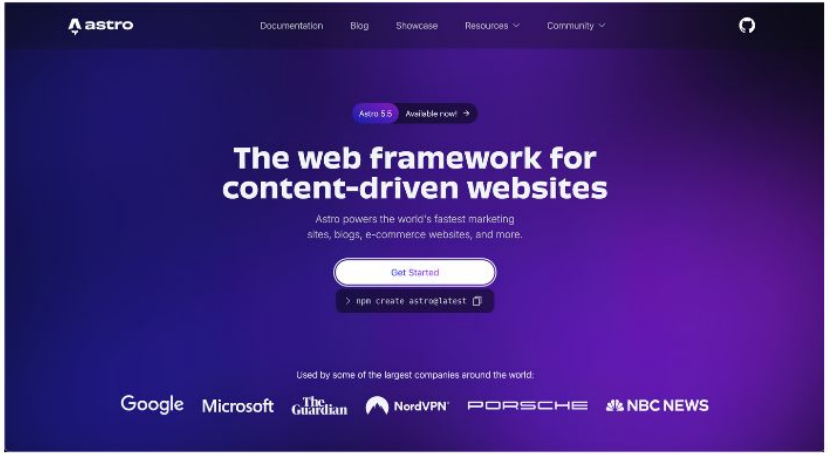
Apps have permissions + capabilities + optionnally register themselves to a DynamicSupervisor



Websites

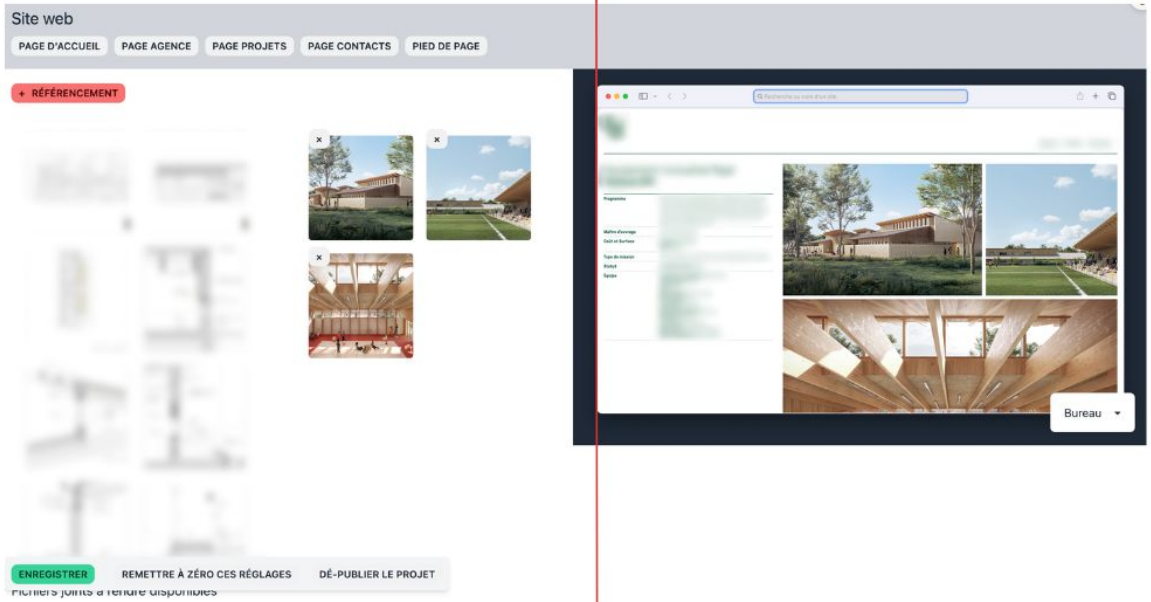
- Must be cheap and fast to host -> static/edge hosting
- Must continue to exist if the client leaves me -> no dependency on Alzo as a backend
- Must provide a live preview to the editor -> should be embeddable and reactive to changes
- Should be easy to develop by freelance friends -> HTML + CSS + Vanilla JS
- Should leverage proven solutions for SEO & performance -> state-of-the-art static site generators
- An instance Must be able to do multiple websites -> no "monolith + 1 website" fixed architecture
- Must be a 2025 development experience -> no bespoke solution

Control Astro.build from Elixir



Liveview

Astro's dev server, proxied by Phoenix

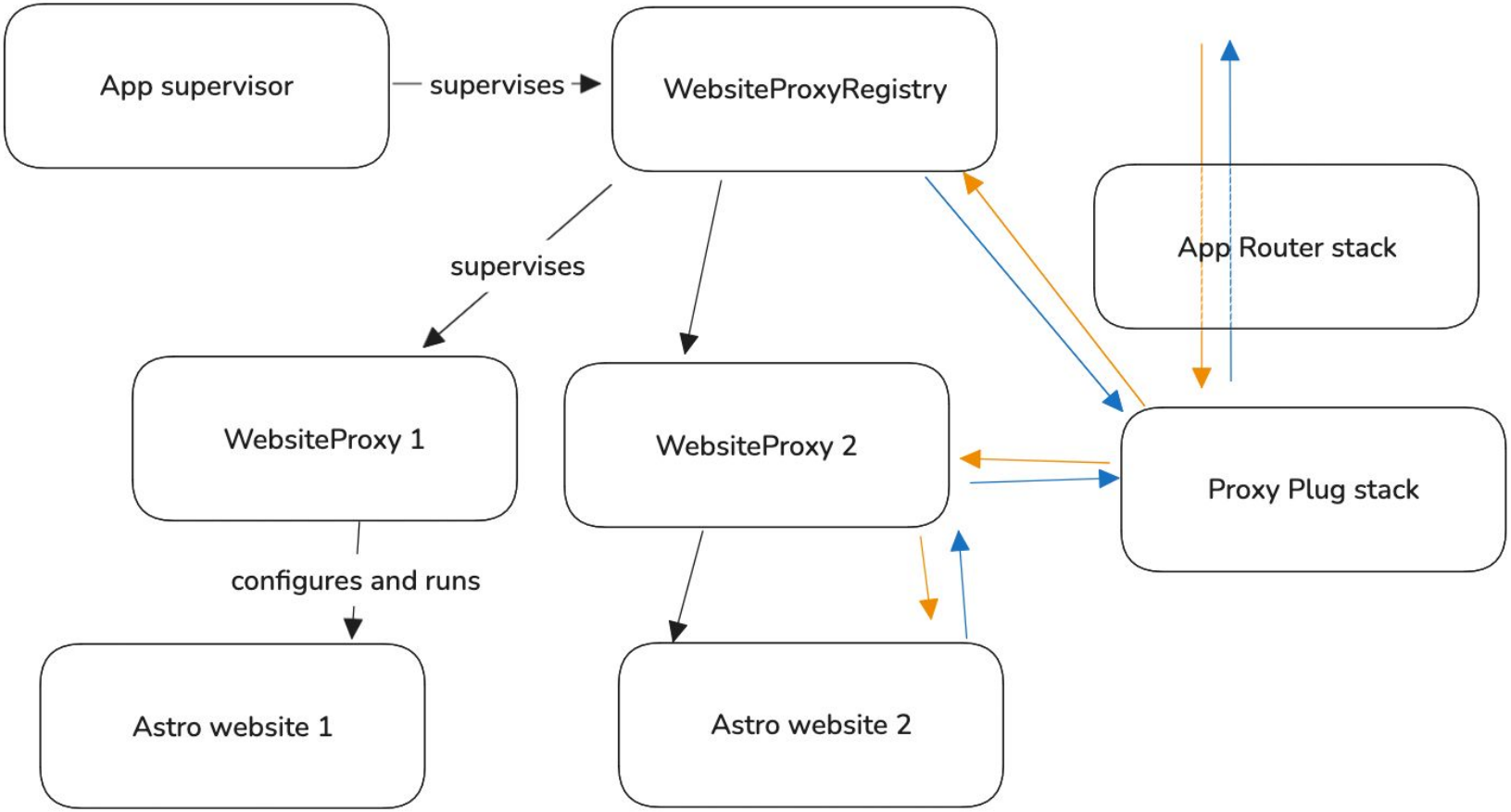


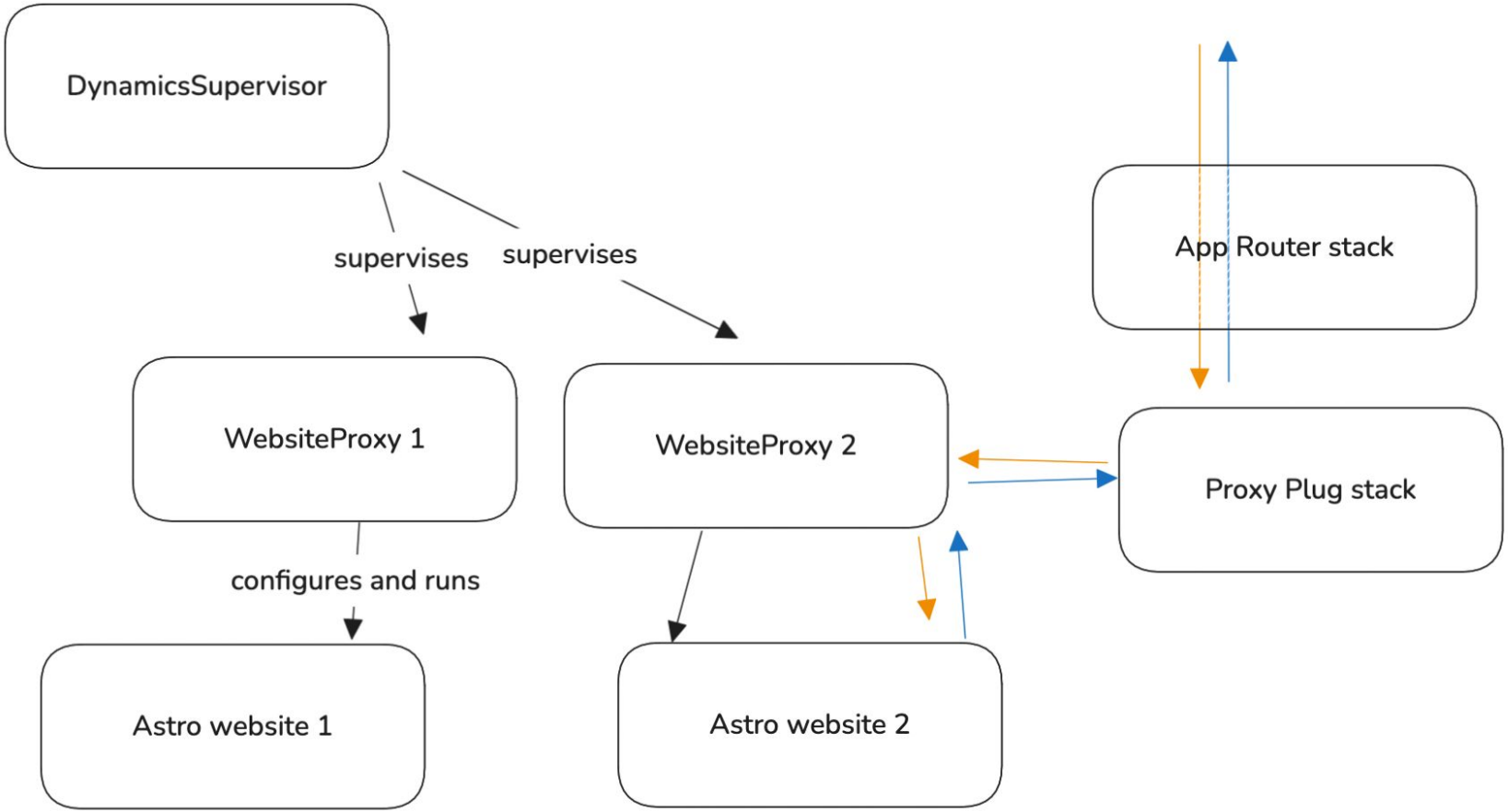
Nodelix

(By Pierre Cavin / sheerlox)
- Install various versions of Node/NPM at runtime
- (in progress) run arbitrary one-off scripts from Elixir
- (in progress) run long-running Node processes in ports

Differences with DenoRider
- Node, not deno
- Shells out to Node, not a NIF

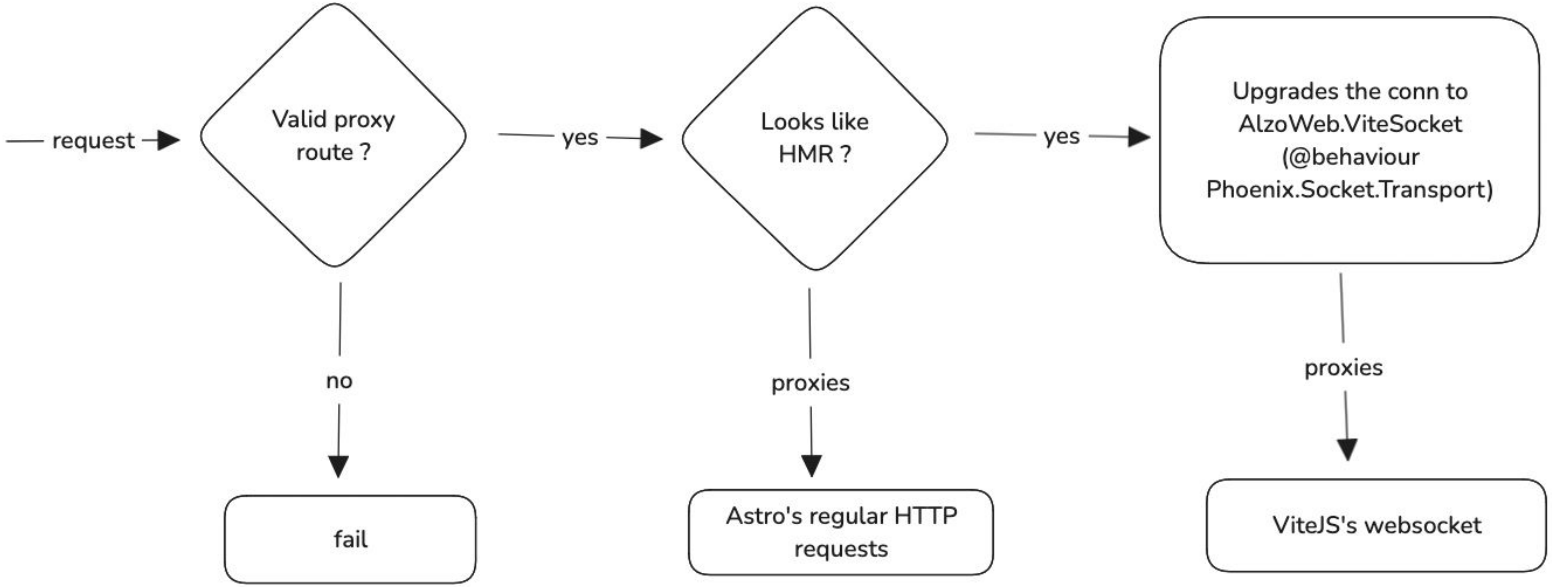
My usage :
- Moved a few Node scripts to post-process PDFs in my Elixir trunk
- Astro.build integration

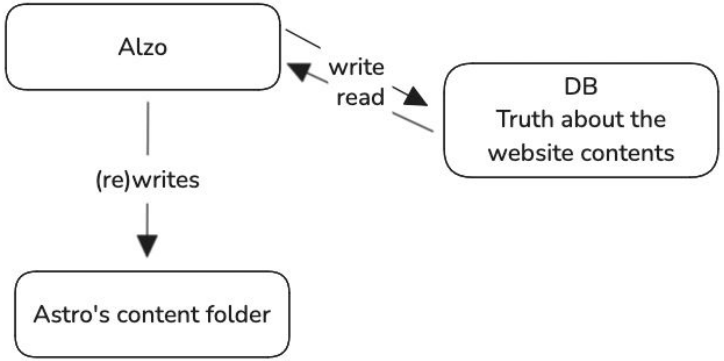

App supervisor
supervises
WebsiteProxyRegistry
App Router stack
supervises
WebsiteProxy 1
WebsiteProxy 2
Proxy Plug stack
configures and runs
Astro website 1
Astro website 2


DynamicsSupervisor
supervises
supervises
App Router stack
WebsiteProxy 1
WebsiteProxy 2
Proxy Plug stack
configures and runs
Astro website 1
Astro website 2

Proxy Plug stack

- Determines if the request should 404 or if it should be proxied
- Routes to the correct proxy
- Determines if it should upgrade to a websocket for Vite's HMR





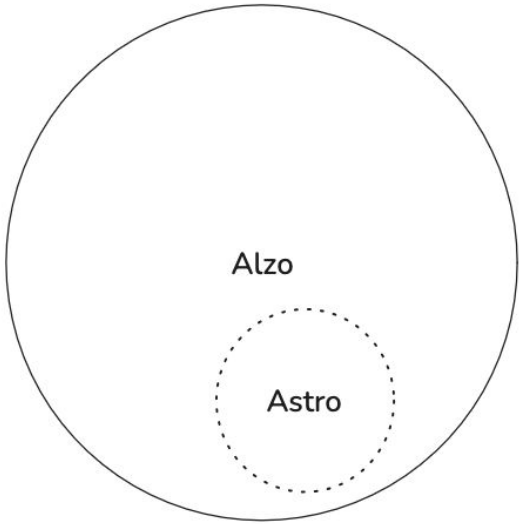
If a change on a project happens :
- via the site editing UI
- via Alzo in general

There's no difference for Astro
-> hot content reload triggers
-> sync out of the box !

If a colleague edits a project's title
While you were editing the site
-> both see the truth update

Pros :
- native hot reloading
- dev server not exposed
- astro has been stable
- can be 100% independent
- can be made to look like messaging
- integration tests can look like normal tests

Cons :
- a bit early to tell
- static site generation (quite fast with change tracking)



To wrap it up

I prefer	over	so I need
a single codebase	multiple services	
abstracted transport with messaging	knowledge of transport	pluggable behaviour
to keep extraction in mind	coupled services	wrappers to get messages
high modularity	strict monolith	build-time / runtime parts
to have runtimes on-demand	more initial dependencies	well-known framework integrations
to have easy collaboration	arcane technologies	

risks

codebase scaling
third-party deprecations
many similar but different apps
nonstandard liveview features

mitigations

low client count
stable third party choices
build genericity over common patterns
watch project directions