ADR: Frontend Tech - Frontend Framework

Status	COMPLETE
Impact	HIGH
Driver	@gw
Approver	
Contributors	
Informed	Engineering
Due date	Jan 26, 2024
Resources	ADR: Frontend Tech

Relevant data

Rather than assuming we cannot or should not change frontend frameworks this exams the possible choices out there.

Options

React

Currently the “standard” for frontend frameworks and what we mostly utilize. This is by far the most popular framework, react vs solid-js vs svelte vs vue | npm trends.

Pros

- Popularity means increased access to tools, libraries, and community
- Easiest to hire for and plenty of material for training
- React Native is arguably the strongest part of React today (could consider this separate)
- “Good enough”
- Internally people have varying levels of knowledge (can help others)

Cons

- Footguns galore
  - Easy to have poor rendering performance that has negative impact (flex ticket has/had this happening)
- Larger bundle size than other frameworks
- Worse performance than other frameworks
- Baked in primitives are exceedingly primitive and often bad
  - State management is poorly compared to other frameworks and libraries
- Internally people have varying levels of knowledge (requires more time helping avoid footguns)
- Has been promising React Forget for over 3 years
- Next ↔ React relationship seems problematic

Vue

Only utilized for a portion of eTicket and was not maintained (v3 came out in 2020 and the package.json v2 is being utilized). Nuxt, utilized by eTicket, hindered adoption of Vue 3 as it took them 2 years to move to it (Announcing 3.0 · Nuxt Blog). Vue 3 was released on September 2020, Announcing Vue 3.0 "One Piece" | The Vue Point , but then took over a year to become the default, Vue 3 as the N ew Default | The Vue Point on Jan 2022.

Pros

- Community appears to be producing as much if not more than React today
  - vite & vitest are examples of projects worked on by many members of that community
- Is working on “Vapor Mode” to improve performance
- Less footguns
- More first class functionality such as Pinia
- Faster than React
- Has largely analogous projects to React such as Nuxt

Cons

- We started to move away from it
- Likely have less knowledge around this
- Fits an intersection of more popular than other frameworks, but also slower than the others
- No construct in SST for Vue’s metaframework(s)
- Similar but still different than React

Svelte

Not utilized currently.

Pros

- Svelte’s compiler is effectively what React wants to build to solve the inherit framework problems
- SvelteKit is also supported in SST
- Smaller community with less fragmentation
- Svelte 5 is likely to become one of the fastest frameworks based on preview (Interactive Results)
  - Moving to “signals” and general improvements
- State of JavaScript 2022: Front-end Frameworks (2023 not yet released), puts Svelte near the top on many aspects such as retention & interest
  - Interest makes hiring for a smaller framework fall into The Python Paradox

Cons

- Fairly different than others due to leveraging a compiler
  - Larger learning curve than Vue
- Smaller community than Vue
- No knowledge within company

Solid

Not utilized currently

Pros

- Has historically pushed what everyone else is trying to do or is now doing
  - Signals, fine grained reactivity
- One of the fastest & smallest frameworks available

- Supported by SST and used by SST
- Very similar to React allowing for easier adoption
- Near the top like Svelte on many aspects such as retention & interest

Cons

- Incredibly small community
  - This has led to even slower creation of additional tooling & libraries
  - Less third party integrations
- Despite being similar to React there are still Solid specific aspects
- Very little knowledge within company
- Solid's metaframework is youngest of all

Recommendation

`React` is the only choice that involves no changes in the current company direction. However, in isolation or if we had the appetite for this my recommendation would to instead utilize either `Svelte` or `Solid`. As mentioned elsewhere, I believe The Python Paradox applies to both of these choices. `Svelte` has many upsides, but requires a lot more transitioning due to how different it is. `Solid` provides an easier migration path from `React` as you can for simple components trivially port them, but suffers from a significantly smaller community which creates more gaps in third party tooling & libraries.

With all of that said, there is likely to never be time or room to completely change to a different framework, so by default `React` wins.

TLDR

Use `React` unless there's desire and time for something better.

Decision

The default choice for new projects will be React. However, there was strong interest in Svelte and if there is a small independent project that is a good testbed for evaluating we'd be open to giving it a try. React Native will not be changed regardless as we do not foresee a second mobile app.