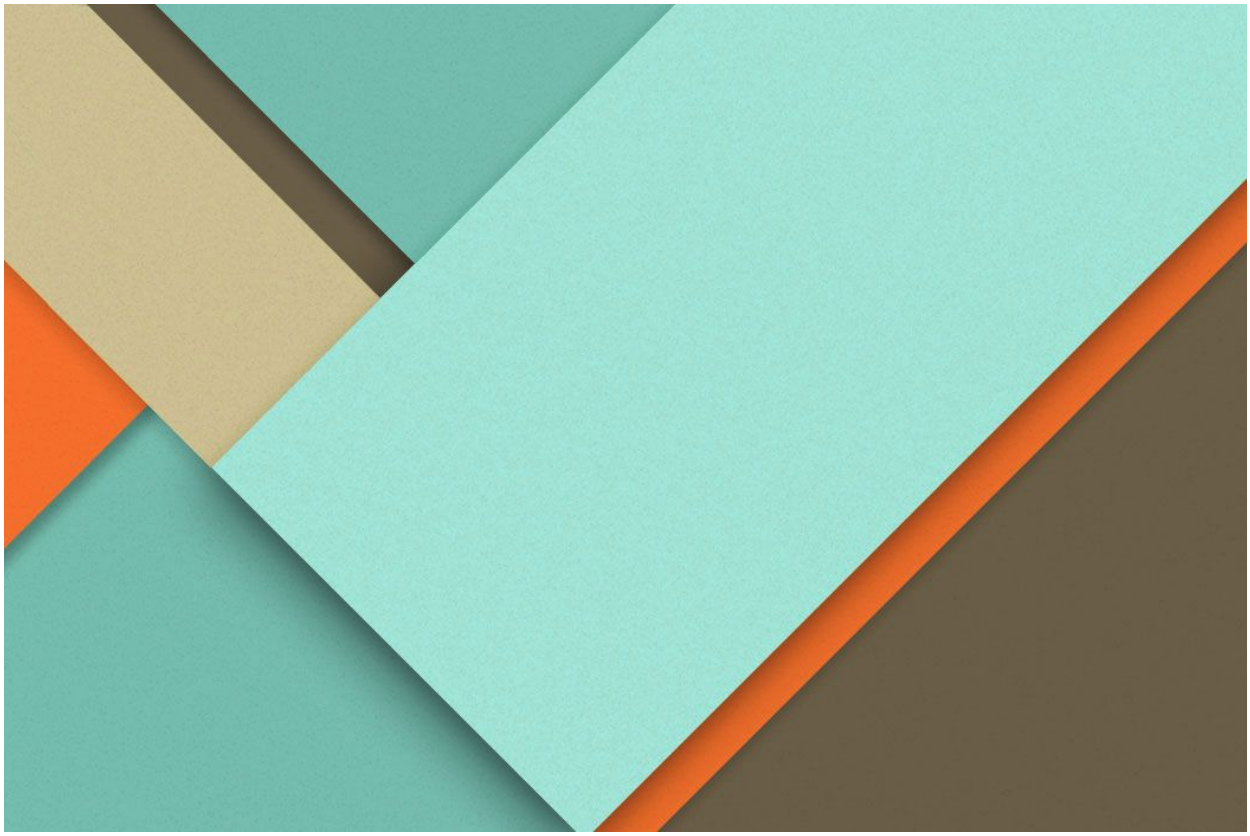

Tokens Lessons Learned

08.09.2019

Sean Midford
ESDC

React & Semantic UI

The tokens web app used React and Semantic UI. React is a framework open sources in 2015 by Facebook. It works well for websites that have many independently updating portions - for example Facebook or Twitter. The tokens app also had many individually updating components, making React a good choice. React is written in JavaScript, very fast, and provides convenient lifecycle hooks to load information from an api independent of the UI.

Semantic UI is a css library for styling the website - similar to Bootstrap. It uses css class libraries reflecting complete sentences, making it very readable. Recently, development on the project has slowed, Recommendation is to stick to Bootstrap or explore another alternative such as Foundation or Material.

NodeJS Express

The web server and API was also written in JavaScript. The api was handled by Express. NodeJS is extremely fast, and since Express is also in JavaScript, we could move to a completely end to end Javascript stack could reduce the need for language and framework knowledge (no more C#, ASP.NET, JAVA, etc.).

Express is lightweight, easy, and written in JavaScript.

MongoDB

MongoDB is a NoSQL database that reflects JavaScript objects. MongoDB contains ‘Collections’ of ‘Documents’ (Javascript objects). MongoDB is extremely fast, but because there are no relations between data, there are constant trade offs between performance and data duplication. For example - storing the user’s name in the Users collection, and their balance in the Transactions collection means two lookups must be performed for a user (cost on time). Alternatively, a user’s name can also be cached/duplicated and stored with the balance in Transactions (cost on space, and makes updating user information difficult).

MongoDB is very fast and scalable for high traffic websites - but because of the tradeoffs, In most cases I’d recommend sticking with SQL (unless MongoDB was uniquely suited).

Docker

Docker was explored for the project, and had the project continued would have been added completely. Docker is a container system for bundling code, software and libraries into individual modules that are then configured to work together to run the project. In tokens, the containers would have been Frontend, Backend, Database, and HyperLedger (had we gone ahead with Blockchain).

The Docker containers are then configured using docker compose. Docker compose wires all the networking, filesystem access, and configuration for how the various components should fit together.

GitHub

- Note -- GitHub is a website for hosting Git repositories. Git is a version control system. Git is not affiliated with GitHub, GitHub is just the most popular remote repository for projects.

The project is hosted on GitHub at https://github.com/sara-sabr/InnovationTokens.

Notes about GitHub usage for this project:

- Each task was added to the project as an 'Issue' (each issue is given an id)
- Developers (me) create a new branch for all work, then check the branch into the main repository (on GitHub)
- On GitHub, developer merges the new branch to master with a Pull Request
- Other developer(s) on the project review the work, request changes, or simply accept the pull request.

There are other standard ways to manage projects on GitHub, we used the merging of feature branches.

HyperLedger (Blockchain)

HyperLedger is Intel's framework for Blockchain. It was decided this project was not a good use for Blockchain. It was not necessary to use the Blockchain to store, for example, transaction data because there was only a single source of truth - the web server. Essentially using Blockchain would be no different than simply using a database, which is what we did instead. Blockchain is best suited for decentralized data systems, involving many participants.