A Collaborative Visual Database

by

Imed Adel

A document submitted in partial fulfillment of the requirements for the degree of

Technical Report

at

Miskatonic University

I Implementation

As exciting as ideas may be, they are worthless if they never leave the books to the real world. In this chapter, we are finally bringing our application to life. We will start, as always, by exploring the possible choice for our implementation, proceed to justify our choice, and finally present our application.

1.1 Technology stack

So far, within the last three chapters, we only made a choice after objectively analyzing the existing possibilities, comparing them, and picking the one that suited our needs the best. This chapter is no different.

In the last chapter, we decided that our application will follow an *n*-tier architecture and a MVVM client. Therefore, we need to start by picking the right technology for each tier. Our requirements, defined in the second chapter, should always be our main focus. That is, speed, performance, accessibility, security, and scalability.

1.1.1 The *n*-tier stack

Client stack

With the MVVM pattern in mind, we have multiple options to pick from. The selection of these options is based on our past-expertise in them.

React An open-source front-end JavaScript library for developing UIs created by Facebook starting in 2013 [22].

It advocates declarative, componenet-based views and relies on a virtual Document Object Model (DOM) to selectively update the browser's DOM based on state changes, thus, achieving better performance and a greater developer experience [19].

Vue.js An open-source front-end JavaScript framework for developing UIs created by Evan You starting in 2014 [8].

In a similar fashion to React, Vue.js also relies on a virtual DOM and advocates a component-based architecture.

Svelte An open-source front-end JavaScript *compiler* created by Rich Harris in 2016 [14]. Contrary to React and Vue.js, Svelte is *not* a run-time library. Instead, it operates like a language on its own that gets compiled down to efficient JavaScript code. Therefore, it does not rely on concepts such as the virtual DOM and minimizes the final size of the application.

While React and Vue.js are similar, Svelte has a completely different approach that can offer better performance. However, speed is not everything—the size of the community matters much more. On top of that, Svelte's performance metrics can be achieved using React too, albeit with more configuration. More concretely, React's monthly downloads were 54 times Svelte's downloads during the same month and 5 times Vue.js' downloads [3]. Library-wise, one can find up to 160,000 open-source libraries for React, compared to 50,000 for Vue.js and 3,000 for Svelte [15].

Taking all of these points into consideration, React seems the perfect choice for our application.

Server stack

Since they have to handle thousands of computationally expensive requests with millisecond-latency, our servers' technology stack has to be as performant as possible. However, while performance is the top priority, it is also important to pick a stack with large community support, otherwise our implementation might have a poor developer experience, and it might face maintainability issues in the future.

The choice of the technology is also limited by our expertise. A performant language or framework with no expertise would as good as or maybe even worse than a slow language with experience in using it.

Node.js An open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine[1] and executes JavaScript code outside a web browser. It has an event-driven architecture that aims to optimize throughput and scalability in web applications, especially ones with real-time requirements [13].

Go A statically typed[2], compiled programming language created at Google. Its syntax is similar to C, but with memory safety, garbage collection, structural typing, and concurrency [9]. Being a compiled, concurrent programming language, Go usually offers better performance than Node.js [6].

[1] The JavaScript engine used in Google Chrome

[2] Type safety is checked during compilation

Rust A multi-paradigm[3] programming language designed for performance and memory safety, which it achieves without garbage collection [5]. It is notable for its extreme speed and low usage of server resources even when compared to Go [20].

When comparing our options, Rust wins by sheer performance, while Node.js wins for its large community and the ability to share code between our front-end and our back-end. However, since our application is technically a database on top of another database, performance is an important metric. This is why Go, with its performance and community, is the best choice for our use case.

Database

Merebase is essentially a no-code collaborative database on top of another low-level database. Aside from being regularly queried by our users, our database will act as the source of truth for real-time collaboration. Therefore, the choice of our application's data store must primarily rely on the performance of such store.

MongoDB A source-available[4] cross-platform NoSQL database program [21]. MongoDB uses JSON-like documents with optional schemas. It is developed by MongoDB Inc. and licensed under Server Side Public License (SSPL) [21]. It was initially released in 2009 [10].

MySQL A widely used open-source Relational Database Management System (RDMBS) initially released in 1995 [11].

PostgreSQL A free and open-source RDMBS emphasizing extensibility and SQL compliance initially released in 1985 [1, 16, 18]. It offers various features, including transactions, automatically updatable views, materi-

[3] It supports object-oriented, and functional programming, among others

[4] A type of a licensing in which the source code can be viewed, but with no or restricted modification

alized views, triggers, and stored procedures [1]. It is designed to handle a wide range of workloads, from single machines to data warehouses or web servers with numerous concurrent users. Some notable products built on top of Postgres include Amazon Redshift, a data warehouse service.

Since each user can shape their database however they like, it might be tempting to choose a NoSQL database, such as MongoDB, for our application. However, benchmarks show that SQL databases, such as Postgres and MySQL, perform significantly better [2].

Finally, when it comes to choosing between SQL databases, MySQL is the most taught one in our schools, and it is the most used in enterprise software. However, Postgres or PostgreSQL offers more concurrency features [12, 17], and it was demonstrated to be more performant than MySQL [7]. Furthermore, the latter's open-source, non-commercial, liberal nature makes it more popular among developers online, thus, providing it with more libraries and extensions.

Based on these points, we decided to choose Postgres as our database.

Pub-Sub

Contrary to the choice overload of the other tiers of our stack, the pub-sub tier is easily dominated by Redis.

Redis is an in-memory store [4]. It can be used as a distributed, in-memory key–value database, cache, or message broker. It supports multiple data structures, such as strings, lists, maps, and sets.

1.1.2 Detailed technology profile

TypeScript

Yarn

Next.js

Zustand

Reakit

TailwindCSS

Virtualization

Websocket

Gin

SQLC

Dbmate

Docker and Docker Compose

Git

VS Code and VS Code Remote

Jetbrains Datagrip

XeLaTeX and Tikz

Acronyms

CmRDT Commutative Replicated Data Type

CMS Content Management Software

CRDT Conflict-free Replicated Data Type

CvRDT Convergent Replicated Data Type

DOM Document Object Model

FDD Feature-Driven Development

JSON JavaScript Object Notation

MVC Model-View-Controller

MVVM Model-View-ViewModel

OT Operational Transformation

P2P Peer-To-Peer

RBAC Role-Based Access Control

RDMBS Relational Database Management System

SaaS Software as a Service

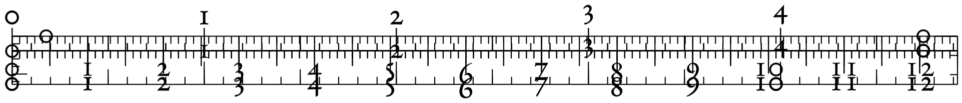
SSPL Server Side Public License

UI User Interface

UX User Experience

Acronyms

WOOT	WithOut Operational Transformation



Bibliography

[1] *1. What Is PostgreSQL?* en. May 2021. URL: https://www.postgresql.org/docs/13/intro-whatis.html (visited on 06/06/2021).

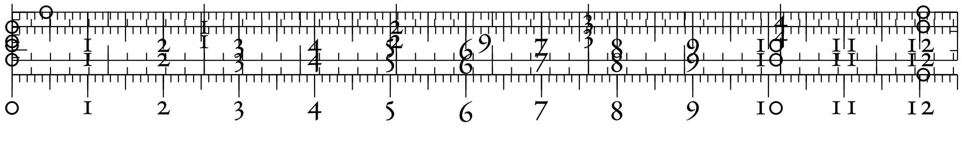
[2] *Benchmark: MongoDB, PostgreSQL, OrientDB, Neo4j and ArangoDB*. en-US. Feb. 2018. URL: https://www.arangodb.com/2018/02/nosql-performance-benchmark-2018-mongodb-postgresql-orientdb-neo4j-arangodb/ (visited on 06/06/2021).

[3] *Compare npm downloads for react, vue and svelte - npmcharts* □. en. URL: https://npmcharts.com/compare/react,vue,svelte (visited on 06/06/2021).

[4] *FAQ – Redis*. URL: https://redis.io/topics/faq (visited on 06/06/2021).

[5] *Frequently Asked Questions · The Rust Programming Language*. June 2016. URL: https://web.archive.org/web/20160609195720/https://www.rust-lang.org/faq.html#project (visited on 06/06/2021).

[6] *Go vs Node js - Which programs are fastest? | Computer Language Benchmarks Game*. URL: https://benchmarksgame-team.pages.debian.net/benchmarksgame/fastest/go-node.html (visited on 06/06/2021).



[7] M. Khushi. “Benchmarking Database Performance for Genomic Data: DATABASE BENCHMARKING.” en. In: *Journal of Cellular Biochemistry* 116.6 (June 2015), pp. 877–883. ISSN: 07302312. DOI: 10.1002/jcb.25049. URL: http://doi.wiley.com/10.1002/jcb.25049 (visited on 06/06/2021).

[8] C. Macrae. *Vue.js: up and running: building accessible and performant web apps*. First edition. OCLC: on1002834117. Sebastopol, California: O’Reilly Media, 2018. ISBN: 9781491997246.

[9] C. Metz. *Google Go boldly goes where no code has gone before*. en. URL: https://www.theregister.com/2011/05/05/google_go/ (visited on 06/06/2021).

[10] *MongoDB System Properties*. URL: https://db-engines.com/en/system/MongoDB (visited on 06/06/2021).

[11] *MySQL System Properties*. URL: https://db-engines.com/en/system/MySQL (visited on 06/06/2021).

[12] *MySQL vs PostgreSQL – Choose the Right Database for Your Project*. en. URL: https://developer.okta.com/blog/2019/07/19/mysql-vs-postgres (visited on 06/06/2021).

[13] Node.js. *About*. en. URL: https://nodejs.org/en/about/ (visited on 06/06/2021).

[14] R. H. S. Nov 26 2016. *Frameworks without the framework: why didn’t we think of this sooner?* en. URL: https://svelte.dev/blog/frameworks-without-the-framework (visited on 06/06/2021).

[15] *npm*. en. URL: https://www.npmjs.com/ (visited on 06/06/2021).

[16] *PostgreSQL System Properties*. URL: https://db-engines.com/en/system/PostgreSQL (visited on 06/06/2021).

[17] *PostgreSQL vs MySQL*. en-US. URL: `https://www.sumologic.com/blog/postgresql-vs-mysql/` (visited on 06/06/2021).

[18] *PostgreSQL: History*. Mar. 2017. URL: `https://web.archive.org/web/20170326020245/https://www.postgresql.org/about/history/#` (visited on 06/06/2021).

[19] *React – A JavaScript library for building user interfaces*. en. URL: `https://reactjs.org/` (visited on 06/06/2021).

[20] *Rust vs Go - Which programs are fastest? | Computer Language Benchmarks Game*. URL: `https://benchmarksgame-team.pages.debian.net/benchmarksgame/fastest/rust-go.html` (visited on 06/06/2021).

[21] *What Is MongoDB?* en-us. URL: `https://www.mongodb.com/what-is-mongodb` (visited on 06/06/2021).

[22] *Why did we build React? – React Blog*. en. URL: `https://reactjs.org/blog/2013/06/05/why-react.html` (visited on 06/06/2021).