Technology Stack

Our technology stack consists of an implementation involving Vue.js, Vuetify, Node.js, Firebase Cloud Firestore, Firebase Hosting, Firebase Cloud Storage, Firebase Cloud Functions, and Firebase Authentication.

Technology Services Used:

Vue.js:

Vue.js is a JavaScript framework that allows dynamic binding of data to help create interactive web applications with HTML, CSS, and JavaScript. Vue.js will be used to construct the bulk of the client, allowing our sponsor, Kate to create documents, and search on them with an easy interactable interface. Vue.js will also serve as the core of our project, meaning that all the code related to interacting with the database and manipulating it, including executing queries, will be done in packages that integrate with Vue.js.

Vuetify:

Vuetify is a Material Design component framework that integrates with Vue.js allowing the creation of customized components that can be reused for whatever needs the project might call for. Vuetify also comes along with a preset library of reusable components that come pre styled and the ease of data binding so there’s no need to worry about trivial things like event listeners or spend hours tinkering with CSS.

Node.js:

Node.js is an asynchronous open source server framework for JavaScript that allows our web application to send and receive requests from our database. Node.js is the main way that our Vue.js client will be able to interact with our database, and will be used for all requests that need to be made between the client and the server. The quality review documents can be created and sent to be stored on the server, as well as retrieved at any time due to the usage of Node.js. Node.js is also used to install Vue.js and set up the environment needed for our application, as it contains a package installer that lets us stay up to date with the latest implementation of each part of our tech stack.

Firebase Cloud Firestore:

Firebase Cloud Firestore is a scalable NoSQL cloud database that is developed as one of the products in the Firebase library. Firestore allows us to have a central database to store all of our data related to our web application including the quality review documents, and individual users' information and permission levels. The main appeal of Firestore is that it is a serverless implementation that still allows us to execute queries and interact with our data in flexible ways, while also automatically caching this data on the client so that even if the network was to go offline, the information would still be available from that moment in time. Firestore is also on a pay to scale plan which means that the amount that is paid will be directly related to how much usage Firestore receives, and can even be completely free if Firestore doesn't receive too high of a usage.

Firebase Hosting:

Firebase Hosting is a service offered by Firebase that allows our web application to be deployed onto a live server so that it can be accessed on the web. Firebase Hosting integrates directly with our Vue.js client and deploying changes live is as easy as running a command in the command line. Firebase Hosting offers SSL right out the box without needing any special sorts of configuration, giving the hosted web application a secure hosting solution in an incredibly lightweight and simple fashion. Firebase Hosting can even integrate with our GitHub repository so that we can get a live preview of the app when reviewing pull requests, and even setup pull requests to automatically trigger a redeploy. Firebase Hosting offers free domains as well as offering the ability to use an existing custom domain instead.

Firebase Cloud Storage:

Firebase Cloud Storage allows us to store the quality reviews online in a virtual storage, so that we can access them at any time from our client by calling a Firebase Cloud Function. Cloud Storage will be able to efficiently store all files in a secure place, which we can access at any time, meaning that our sponsor Kate will be able to receive any document stored in the system that she previously created, as well as upload new ones as soon as she's finished.

Firebase Cloud Functions:

Firebase Cloud Functions are serverless back-end functions that are created inside of the Firebase Dashboard and enable our application to trigger these functions based on events that occur. This way we can access any of our other Firebase services as well as trigger the function from within those services based on events as well. An example of our usage would be that when a request is sent from our client to retrieve a specific document, it would trigger a cloud function that would retrieve that document from Cloud Storage, and return it as a response to our client, allowing us to send and receive documents as freely as we'd like.

Firebase Authentication:

Firebase Authentication is an easy and simple implementation of an authentication system that can be used for user sign ins so that only people who are given access to the application can use it. This ensures that Kate will be able to sign in and use her application as she'd like, and other users she invites can as well, without having to worry about people who shouldn't have access being able to view the documents. This also ensures that there aren't any requests being made to our Firebase instance that should not be allowed, meaning a higher level of security against unauthenticated users.

Utilization:

Overall:

Our web application will overall be a simple website that consists of an area with a text form that users can fill out to create quality review documents, and an area to retrieve these documents at a later time to view them and analyze the metrics on these documents. With our current technology stack, the implementation of each would be as follows:

Implementations:

- Vue.js and Vuetify serve as the front-end client that our sponsor and other users will interact with. Both will be used to serve to construct all the events and features that come with this experience, as well as contributing to the design and overall look and feel of the web application.
- Node.js will be used to install all of our necessary packages such as Vue.js, and necessary plugins to implement Firebase with our Vue application. Node.js will also be used as a way for us to send requests from our client to our server.
- Firebase Hosting will host our web application so that it can be live on the web, available for our sponsor and other users to be able to access it.

- Firebase Cloud Firestore will be the database for our web application, storing all important information related to the application such as user data, and document data.
- Firebase Cloud Storage will store all of the PDF documents that get generated by our sponsor, allowing us to access all of the documents from one place. This will let us be able to add documents as needed, and view any of our documents at any time.
- Firebase Cloud Functions will allow our application to interact with all of the different Firebase products that we will be using, as well as allowing certain actions to occur in an event-driven system related to anything occurring in our application.
- Firebase Authentication is our authentication service that will provide a higher level of security, ensuring that all of the information and documents from our web application will only be viewable by authorized users, preventing any of it from being leaked when it shouldn't be.

Reliability:

Overall:
Our whole web application is going to be Google based, stored on a web server hosted by Firebase via Google. The hosting is offered via a global Content Delivery Network meaning that there will always be a location near a user to deliver this data to them in real time. Since the hosting is ran off of multiple servers, and backed by a large company with notoriety like Google, the chances of the application experiencing any sort of down time will be incredibly scarce, and can even be monitored directly from here: https://status.firebase.google.com/

Pricing:

Overall:

Vue.js, Vuetify, and Node.js are all completely free and available to the public, meaning that there won't be any costs associated with these parts of our technology stack.

For our Firebase products, they are all offered on a "pay as you go" pricing scale, where pricing is completely dependent on usage, which is perfect for a small scale lightweight application like this. Every product has a certain limit that is allowed to be used for free, before Firebase starts charging at a metered level, meaning that for some of our Firebase products, there's a low chance that we will even end up reaching the threshold to transition from free to paid, lowering the costs even more.

A full breakdown on the overall costs for each Firebase product can be found here: https://firebase.google.com/pricing

Additionally, I've compiled a small table below consisting of just our products and the expected pricing:

Firebase Product Breakdown:

Products	Free	Pay as you go
Firebase Cloud Firestore		
Stored data	1 GiB total	$0.18/GiB
Network egress	10 GiB/month	Google Cloud Pricing
Document writes	20k/day	$0.18/100k
Document reads	50k/day	$0.06/100k
Document deletes	20k/day	$0.02/100k
Firebase Cloud Functions	(Only for Node.js 8)	(Node.js 10+)
Invocations	125k/month	Free up to 2M/month, then $0.40/million
GB-seconds	40k/month	Free up to 400k/month, then Google Cloud Pricing
CPU-seconds	40k/month	Free up to 200k/month, then Google Cloud Pricing
Outbound networking	Google services only	Free up to 5GB/month, then $0.12/GB
Cloud Build minutes	Not applicable	Free up to 120min/day, then $0.003/min
Container storage	Not applicable	No free usage, $0.026/GB
Firebase Cloud Storage		
GB stored	5 GB	$0.026/GB
GB downloaded	1 GB/day	$0.12/GB
Upload operations	20k/day	$0.05/10k
Download operations	50k/day	$0.004/10k
Multiple buckets per project	Not available	Available
Firebase Hosting		
GB stored	10 GB	$0.026/GB
GB transferred	360 MB/day	$0.15/GB
Custom domain & SSL	Available	Available
Multiple sites	Available	Available

Firebase Authentication	Unlimited Authentications	Unlimited Authentications

Pricing Analysis:

Overall, given the scale of our project being that it will be at most 10 users including our sponsor, and that the application can be expected to create at most, 100 documents a year, the majority of the thresholds to go from free to paying will not be reached.

For the Firebase Cloud Firestore, most of the activity will be coming from our sponsor who will be creating documents using the application, and with the most amount of activity being from 10 users who would be in the app at once,reaching over 20k writes and 50k reads a day is highly unlikely, meaning we can assume the overall price for Cloud Firestore will be free.

For the Firebase Cloud Functions, they will serve a very minimal role in our application, and our implementation will be using a version of Node.js higher than 8, meaning that we are given 2 million invocations a month, an extremely generous amount. If we are to assume that our sponsor attempts to view a document from Cloud Storage 1000 times a day, every single day for a month, they wouldn't even reach 1/10th of our given amount each month, meaning we can assume the overall price for Cloud Functions will be free.

For the Firebase Cloud Storage, given the assumption that each document is going to be at most 1 MB large, (a very generous overestimate), with 100 documents being stored a year, that means at most, there will be 100 MB stored in the Firebase Cloud Storage a year. If we go by this assumption and continue at this frequency into an infinite amount of time, we can theoretically stay on the free plan of Cloud Storage for 50 years. With this information, we can overall assume that the price for Cloud Storage will be free.

For Firebase Hosting it's hard to tell exactly how large our application will be right now during the planning phase since it can widely vary, so it is also hard to gain a good estimate on the price of this service. However, even if our application ends up being 100GB big, (a very generous overestimate), the price would only come out to a very minimal $2.34 (10GB included for free + $0.026 * 100 = $2.34). Overall, we can assume from this information that our pricing for Firebase Hosting will come down to anywhere between $0 - $2.34 in total.

Firebase Authentication offers the same features and functionality on both the free and pay as you go plan, meaning that this service does not cost us anything. Something to keep in mind is that there is a small price that can be associated with Firebase Authentication if we decided to use the phone number authentication service that Firebase Authentication provides, but given that there is likely no need for such a deep level of authentication for a small scale project like that, and it currently is not part of our plans to implement that feature, this is a price we won’t have to consider.

Total Cost:

Overall, with all of the products and services we’ll be using in our technology stack, the total cost of everything will end up being anywhere from completely free, to $2.34. This low price allows us to implement many of the features that our application needs without having to sacrifice any sort of functionality to fit a certain price point.