Industry Study 1 (31489) Report - Part 2

Assessment Task 2

Leon Salsiccia (13550494)
Placement: Venntifact

Abstract

Venntifact's 'Idea Wall' fails to provide an efficient platform for managing internal projects that will benefit the company. Inconvenient to use, untrackable progress and maintenance issues have lead to its complete deprecation with many ideas left unactioned. Therefore, through investigation and user input, this report will explore the organisational value within the Idea Wall's complete digital redesign created with a cloud-enabled architecture. This report aims to identify a solution that not only replaces the deprecated system, but also extends it, better enabling Venntifact to adhere to its core values of introspection and transparency amongst employees.

Introduction

Context

I am currently interning at Venntifact- a start-up digital analytics consulting agency. Being a consulting agency, it is Venntifact's responsibility to provide leading and innovative solutions that not only meet client expectations but rather exceed them. Typically, work is split into billable work- usually, client-related- and non-billable work, which can involve certification and training. Additionally, Venntifact values introspection and transparency amongst not only its clients but its employees as well. As a rapidly growing start-up, only recently have Venntifact had the resources necessary to reflect upon internal systems and processes.

What is the problem?

Venntifact's method for proposing and managing internal projects is too disorganised and tedious to incite any meaningful results. Previously, this was accomplished through sticky notes grouped on a wall which allowed any employee to suggest ideas and improvements. The Idea Wall was introduced to increase employee engagement, reflect on flawed solutions and encourage company growth. Employees write proposals onto sticky notes and place them on the wall for others to see, and hopefully contribute.

The Idea Wall was introduced nine months ago and has become increasingly difficult to manage. Most proposals are overlooked, and collaboration is convoluted, making large scale projects unfeasible. Consequently, the Idea Wall has been heavily neglected with most ideas being dropped, ignored or forgotten. It can be seen that Venntifact's implementation of the Idea Wall does not scale and needs to be replaced in order to drive developments that can benefit the company.

Current implementation of the Idea Wall

Investigation

Why is it a problem?

As a start-up, suggesting innovative and impactful projects that will benefit the company is crucial for continued growth; however, issues presented by the previous system impacted this and Venntifact's core values of transparency and introspection.

Transparency has been a goal of Venntifact since its establishment, which the Idea Wall was intended to complement. Failing to provide an efficient platform to suggest ideas within the company, limits transparency amongst employees and executives, which reflects outwardly to clients. Venntifact lacks a structured and regulated method for employees to voice opinions and provide constructive criticism within the company. This may lead to a decrease in morale or staff feeling incapable.

Introspection is fundamental for a progressive company's growth, more so when that company is a start-up. Venntifact has had successful development over the last year; however, as they grow in size, introspection will become increasingly difficult.

A higher number of employees- with many working remotely- makes sustaining the Idea Wall impractical and creates additional demands on management. The sticky notes also do not provide enough space to record idea development progress or staff involvement, further hindering management and making large scale or long term proposals impossible.

The Idea Wall failed due to its lack of ability to scale and support large projects or provide a means to track staff involvement and progress. This disables Venntifact's growth and its ability to achieve its goals of transparent and company-wide introspection.

So what?

If Venntifact continues to use the Idea Wall, many issues will worsen and new issues will arise, including:

- Impact on management - As staff size increases, the Idea Wall will become cluttered as wall space is finite. Ideas would need to be reorganised by hand, or stored in multiple locations, both of which are undesirable for staff who may work remotely. This will complicate the tracking of progress and lead to the desynchronisation of work.

- Disrupted workflow - Employees express concerns regarding the lack of non-client related work. Often, staff undertake training modules or certifications when they have spare time. However, this can become exhausting when performed in succession, decreasing morale. Employees do not have other means to contribute to the company once they complete all training available to them. This, in turn, delays management with finding work for staff who have completed training.

- Financial burden - As a result of impacted workflow, deliverables to clients may suffer, and the overall quality of the service may drop. Staff may

become apathetic from training and produce lower quality work, and management will need to spend valuable resources supporting both the Idea Wall and employees. Consequently, Venntifact will suffer financial burden through lost leads, inefficient workflow and wasted resources.

Who cares? (benefits)

Many individuals will benefit from the change, including:

- Clients - Clients will benefit from improved workflow and efficiency amongst staff, resulting in higher quality deliverables and overall service.

- Employees - Employees will now have a regulated and effective platform to voice concerns and suggest improvements to the company. They will be able to refine their skills by taking on smaller projects or collaborating with others on larger projects. This benefits not only the company but also the individual as they can utilise these skills throughout their career. They will experience less fatigue from prolonged certification training, resulting in increased morale.

- Management - Management will be better equipped to lead staff working on internal projects through the digital platform. They will be further enabled to assign meaningful work that will have a firsthand impact on the company. The onboarding process will also improve as management will be able to assign smaller tasks to new staff, facilitating collaboration with peers in order to ease them into their role.

Why hasn't it been done yet?

After much rapid development and company expansion, only recently has Venntifact had the time and resources to support such a change. Before, resources were spent building and maintaining client relations while Venntifact was still

establishing itself within the market. Additionally, there was not enough technical staff to devote to internal improvements.

Existing Knowledge

What is known to work

The Idea Wall's most significant benefit was in its simplicity. The action of proposing an idea was intuitive and efficient, requiring only a short amount of time to complete and little prior knowledge regarding the use of the system. It is a universal concept that works well for quick notes and ideas, especially amongst small groups.

What is known not to work

Where the Idea Wall disappointed was in its maintainability and management. Having a physical implementation severely inhibited its ability to scale as wall space availability became a restriction. As a result, employees had to be assigned to maintain the wall, expending company resources.

This, in turn, affected the tracking of progress, becoming increasingly difficult as the Idea Wall grew. The sticky notes failed to provide enough information such as progress nor allow said progress to be updated effectively. Having a physical implementation also confined the use of the system to a specific location within the office, which was undesirable as employees are often out of office supporting clients on site. This impacted the proposal of new ideas, and by a combination of the above problems, idea recognition, resulting in many proposals being unseen, forgotten and generally unactioned.

What is not known and why this prevents progress

Prevention in progress towards a solution was due to it being unknown whether solving the problem would result in its use. The current Idea Wall was rarely used,

making it difficult to judge the significance of developing a solution. Would the new solution even be used? Would the benefits of a custom solution outweigh the simplicity of implementing an existing third party solution?

Other factors that prevented progress include determining the feasibility of using a Backend as a service (BaaS) like Firebase. Would this service support our requirements? How long would it take to learn the platform? Questions like these were apparent, yet challenging to answer having limited experience with these technologies. Although there was a surface level understanding of the concepts that would be used during development, it was clear that further research was required for the solution to be successful.

Existing Solutions

Other potential solutions included the use of an online spreadsheet, Atlassian's Jira Software and Trello; however, these failed to provide a solution as suitable as custom software:

- Excel - Proves challenging to manage on a large scale. Is less intuitive for the required task and does not provide an efficient means for providing feedback such as a comments section.

- Jira - Although similar to the proposed solution, Jira is an external product which adds further complexity. Additional accounts will need to be created and managed, and lack of integration into existing internal systems- such as Venntifact’s custom timesheet software- makes it less desirable.

- Trello - Does not succeed as a solution due to the same reasons described above for Jira.

Requirements elicitation

Two methods were used to gather requirements for the project; Introspection and Interviews.

By analysing both the previous solutions and past attempts at addressing its flaws, It was decided that a physical implementation was insufficient, and the new solution would benefit significantly by transitioning to a digital one. Since the current Idea Wall was unused, observation elicitation techniques were impossible. Instead, interviews were held to help gather information about what employees expect from the new digital Idea Wall. Three separate individuals from different teams were questioned about the previous solution and issue tracking tools in general. Questions were focused on their experience with the Idea Wall and other issue tracking software, such as Jira and Trello. They were asked what they liked and disliked, and what they would change.

By the end of the elicitation phase, it was concluded that the proposed solution would be one that provides:

- Trackable idea progress
- Extensibility and scalability
- Support for large scale projects
- Auditing capabilities and management tooling
- Centralised accessibility via Online Web Application
- Categorising and organisation of ideas
- Integrations into existing solutions

Designing the Solution

Researching

This research section aims to provide backing to the assumptions made during the investigation and development of this solution. Additionally, it will act as a guide to identify the solution that best fits the requirements. These findings supported the decisions made regarding the architecture and technical foundation of the developed solution.

Issue Tracker

When first designing the solution, it was important to understand the main functionalities of issue tracking software. This led to researching existing examples such as Jira and Trello. An issue tracker appears simplistic in concept but requires a significant amount of thoughtful planning and consideration. The before-mentioned examples served as inspiration regarding the journey flow of the application: the logical progression a user should follow to get from ideation to proposal and finally to contribution.

Interface

I would also need to develop the user interface (UI) and by extension, the overall user experience (UX). Pinterest.com served as the primary source of inspiration regarding the aesthetic design of the final solution as it offered many examples made by other users. However, much of the rationality behind design choices and the fundamental concepts of designing a successful UI/UX came from sources such as tailwindcss.com and material.io. These sites provided information regarding the theory behind designing an application interface. Designing an interface proved more extensive than first anticipated, and a considerable amount of time was spent learning these concepts.

Architecture

Multiple web application architectures are applicable for the intended solution; however, one stood clearly above the rest in modern web development: component-based architecture. For frontend component-based architectures, the most popular choices include Javascript frameworks; React, Angular and Vue. These options provide the tools and foundation for developers to create powerful user interfaces and single-page web applications that are modular and extendable.

Components are reusable pieces of functionality that are abstracted and exposed using an interface. They are designed to interact with other components, passing data amongst themselves whilst conforming to regulated and predictable behaviours. Essentially, it acts as a way to breakdown the frontend into smaller, more manageable fragments that form the final application. A component-based architecture is one which enables the following characteristics:

- Reusability
- Maintainability
- Extensibility
- Reduced costs
- Functional independence

Choosing amongst the three required further insight into each of their capabilities. Multiple resources, including stackoverflow.com, youtube and the respective framework websites provided an analysis of their strengths and weaknesses. Angular was the oldest and React was the most popular, however, Vue- although the newest of the three- provided powerful capabilities that not only matched the other options but in many cases, superseded them. Additionally, Vue was the most user-friendly of the three with a relatively small learning curve, making it the easiest to learn and apply. This is also beneficial for future maintenance of the codebase.

Backend

Typically, websites and web applications have a physical backend server to serve static assets or perform any computational tasks necessary for the functionality of the application. Since managing a physical server is resource-intensive, it was decided that the application would use a Backend as a Service (BaaS), which typically provides:

- Database management
- User authentication
- Hosting
- Cloud storage
- Push notifications

Well known cloud service providers Amazon and Google both have their own BaaS, Amplify and Firebase respectively. Both are excellent choices, though Amplify takes longer to set up and is intended for medium to large-scale applications. For example, enabling real-time subscriptions to Amplify's database requires writing lengthy queries and defining explicit schemas. For Firebase, the flexible, schemaless database allows for real-time subscriptions from the frontend, enabling a more affordable ad-hoc workflow. As Venntifact is still a start-up with limits to the number of resources they can devote to internal projects, as well as time restrictions, Google’s Firebase was used as the application backend.

Constraints

Various constraints applied to this project, including:

Time constraints

The project had to be completed by Friday, 13th of December, the last day of my placement. By then all work had to be handed over to Venntifact.

Budgetary constraints

The previous implementation of the Idea Wall, although unused, was inexpensive. The new solution also had to be relatively inexpensive, however additional costs were expected. Ideally, the solution would be self-sustaining and require limited company resources to operate.

Coding standard

A self-imposed constraint of adhering to a standardised coding pattern to ensure the final solution is both maintainable and extendable by persons other than myself. This was necessary as the solution will move to new ownership once I complete my placement, and will be maintained by Venntifact.

Dependencies

The solution would also rely on external dependencies as little as possible. This meant only using core frameworks and APIs during development, such as:

- Vue JS - A component-based model-view framework for dynamic web applications.
- Firebase API - For interfacing with the Firebase BaaS platform within the application's frontend.
- Tailwind CSS - A utility first CSS framework used to build the applications visual design language.

It was necessary not to introduce any more dependencies than needed to ensure that the application bundle size is small and not subject to any further vulnerabilities. This will be further explained in the next section of the report.

Risk Assessment

The risks associated with this solution primarily lie within its development rather than its implementation.

All unforeseeable circumstances and edge cases had to be taken into consideration when the proposed solution was being developed. Bugs and unintended results are inevitable, however, developing in a component-based and modularised approach ensured that defects within one module of the application did not impact others. This form of risk mitigation was intended while designing the solution and is one of the benefits of using a component-based programming architecture. If an issue is detected, creating a fix will be quick and inexpensive as the issue will be isolated and identifiable. The component-based framework of choice, Vue, also provides additional tooling to help debug components in the form of the Vue.js devtools extension, accessible from the developer console within Chrome.

Arguably, the most significant risk is data privacy. As the application captures both personal and confidential information, in the form of employee details and internal company projects respectively, it is detrimental that the transfer, storage, and access of information is secure. Google's Firebase platform provides excellent security regarding the transfer and storage of data via its provided API; however, permissions to access this data must be configured manually. This was achieved through Firestore (Firebase's database) security rules. They are expressions defined by the developer that determine who should have access to data within the database. By default, Firestore has rules that prevent all access to the database, meaning rules had to be set up before it could be used. Additionally, the Firebase platform will automatically email the project owner and inform them of insecure

rules, should it detect any. Overall, this made for a robust and refined security system that protects both the end-users details and any confidential information.

As mentioned in the constraints section above, introducing unnecessary external dependencies increases the potential for the application to contain vulnerabilities and insecurities. Should any of the dependencies become compromised or cease to function, so too would the solution. This also means additional maintenance is required if any dependency is deprecated or altered. By reducing the number of external dependencies, the security of the application is improved and the codebase will require fewer updates.

Solution Development Method

What is the solution?

A complete re-implementation of the Idea Wall into a digital platform. The new system is a custom-built web application that can integrate into existing internal systems and processes. Moving to a digital platform addressed all current issues while providing additional functionality.

The new system uses existing Venntifact accounts requiring no additional setup. The interface and functionality is intuitive and straightforward, akin to software already used widely within the company; Jira by Atlassian. This ensured that minimal training is required for staff to use the solution.

For creating the application, a prototyping development approach was adopted. This methodology allows for continuous learning and research across the development cycle. Furthermore, an iterative approach ensured that a working prototype was created at the very least, should the project exceed its due date, which was a recognised risk. Other methodologies were considered prior to development as well. The solution was to be custom-built, exempting the end-user

approach. Waterfall or structured methodologies were also not adopted since they increased the risk of time constraints and project costs, providing little flexibility should the requirements change. Prototyping allowed for a balance of rapid development pacing whilst remaining affordable.

The application was developed using Visual Studio Code as the text editor of choice. Visual Studio Code provides tooling in the form of extensions to help with tasks such as version control; for which Github was used as the host.

As previously mentioned, the application itself was developed using a component-based programming architecture with the Vue framework. More specifically, Vue + TypeScript was used for the frontend. TypeScript is a superset of JavaScript developed and maintained by Microsoft that introduces ‘strong typing’ to the language. It is intended for large applications such as web apps, and is also used as a form of intrinsic and internal documentation to improve readability and maintainability.

As for the design and aesthetic look of the application, the Tailwind CSS framework was used to design a unique user interface. Many CSS frameworks provide pre-built components that can be placed into any web app, however, these lack authenticity and look generic. Mock-ups were created using Photoshop before any CSS was written in order to establish an acceptable design. Creating a unique and consistent design language proved more time consuming than initially thought. In hindsight, it may have been preferable to use a more generic, pre-built framework such as Material IO.

Rather than developing a backend, a "serverless" approach was taken and Firebase was used as the BaaS. A serverless approach allows for flexible, cost-effective technical architecture that scales dynamically with the application’s use. They are

excellent for prototyping and beneficial in Venntifact's case, where a "pay for what you use" plan is ideal.

Firestore was used for the solutions database. A schemaless NoSQL database suits the proposed solution as it provides the following benefits:

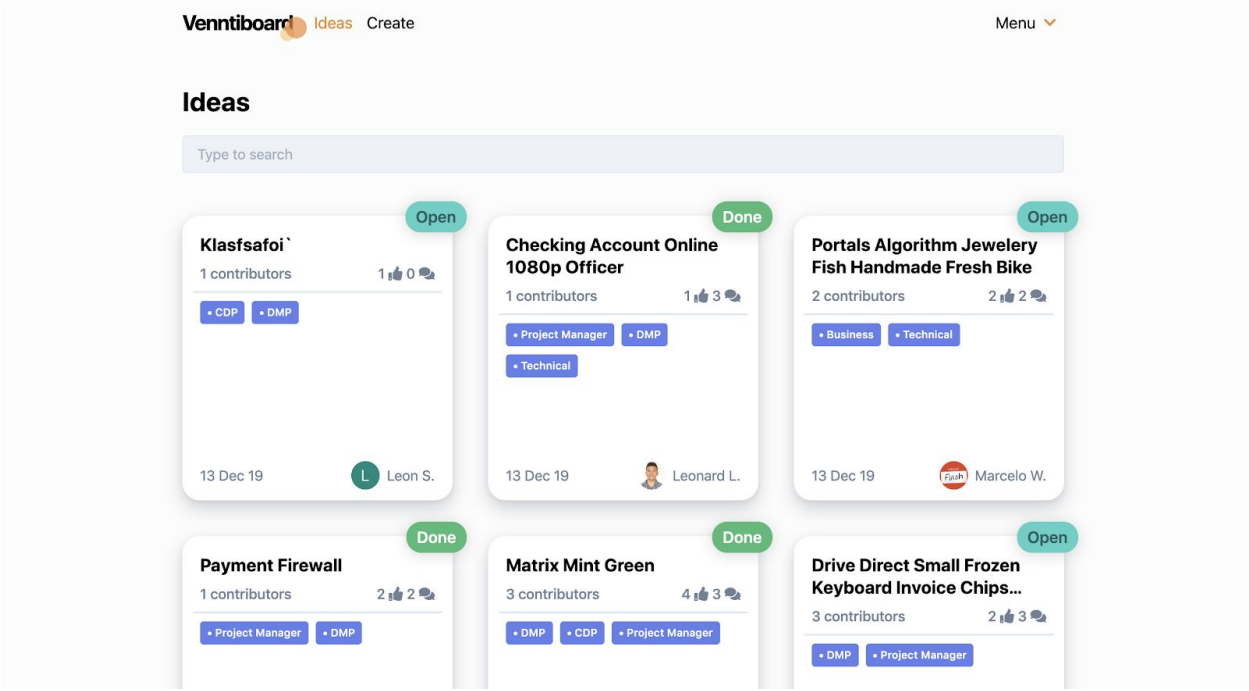
- Flexibility - Since NoSQL databases do not enforce schemas, the underlying data structures can change and adapt to the flow of development. As NoSQL databases are adaptive, initial time spent to define the structure of the database is reduced significantly. Using a flexible, NoSQL database enables inexpensive development approaches such as prototyping.

- Realtime - Many NoSQL databases such as Firestore, allow for real-time bindings between the database and the application view. This provides real-time reporting and updates within the application.

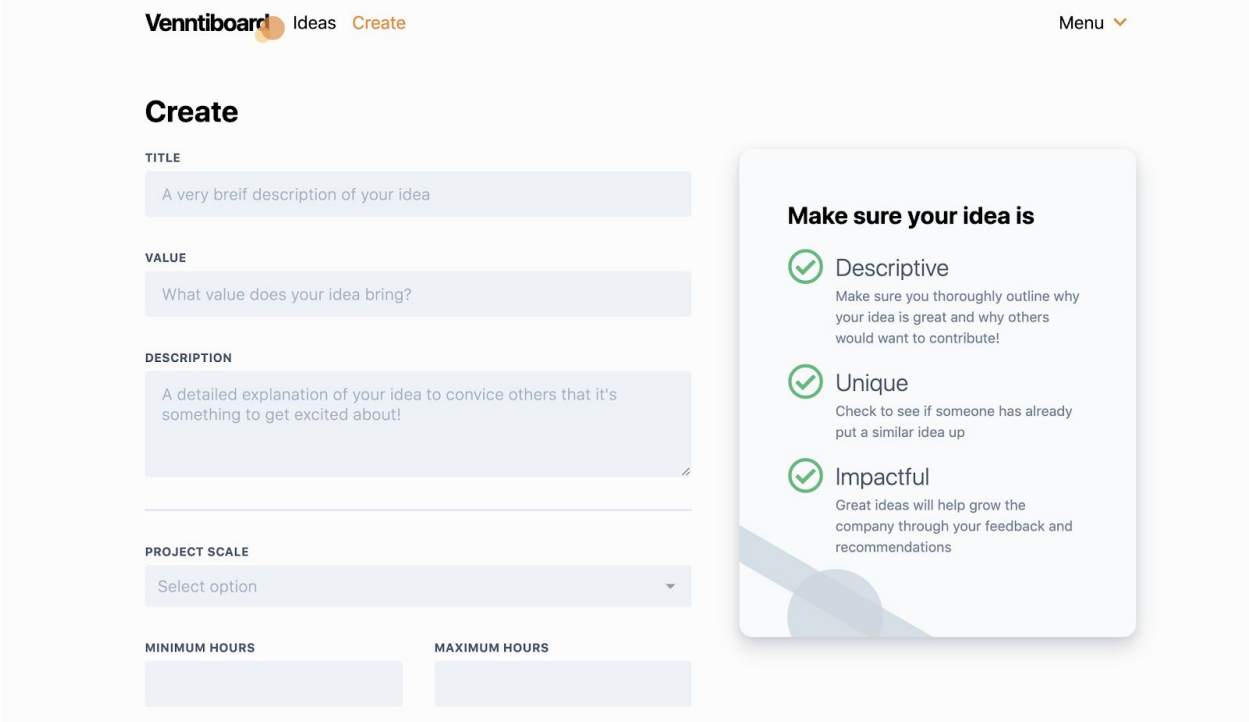
- Transactional - A key feature of Firestore is its transactional capabilities. Transactional databases enable applications to be fault-tolerant in their data processing. A transactional database is one that can revert a database operation, such as a write operation. The means that if any information within the application is incorrectly modified or an action is completed inappropriately, the database can roll back to a previous state. This is necessary as many team members will be accessing and modifying the same information. Using a transactional database ensures information is in its most correct state for all users, and ensures information validity.

By the end of the development cycle, the following were planned:

- Application front end (Interface, UI, UX) is completed
- Journey flow is completed
- Main issue tracking software functionality is completed, including:
    - Creating issues
    - Editing issues
    - Updating issue progress
    - Searching and filtering issues
    - Assignments
    - Likes
    - Comments
- Management tooling is completed and accessible to those with the authorisation to use it
- Serverless functions that act as the application backend are created
- Authentication and authorisation to support permission-based CRUD operations are completed.



Dashboard of the finished application



Idea Creation page

Implementation

Common concerns when proposing a custom solution include resource allocation such as time and funding, but most notable is implementation risks. However, since the Idea Wall is rarely used, a direct cut-over was performed for the implementation method, yielding its benefits of being fast and inexpensive with minimal repercussions. The staff were able to use the solution immediately and development then shifted to an agile workflow in which features can be introduced and modified gradually.

As for the implementation itself, the application was deployed and hosted on Firebase, free of charge. This was achieved through the Command Line Interface (CLI) provided by Firebase. Along with the deployment of static assets and the application itself, serverless functions created during development were also deployed and hosted on Firebase. These functions were managed and provisioned by the platform automatically, scaling dynamically according to the usage of the application. Extensive use of these cloud functions may incur further costs in a "pay for what you use" plan, however, quota caps were set beforehand to avoid unexpected costs. The user authorisation service and the database were also maintained entirely by the Firebase platform, requiring little effort on our behalf. Like other Firebase services, so too will authorisation and database functionalities scale automatically, though additional setup in the form of user permissions and database rules were required. Additionally, in spirit of Venntifact, and the digital analytics services it provides, web analytics was also configured to enable powerful reporting in the form of Firebase analytics. This service was simply enabled like any other from the Firebase console. Finally, the Firebase project was migrated from my Google account to Venntifact’s Admin Google account, transferring complete ownership.

The implementation of the final solution into a production environment was inexpensive, fast and without notable risks. By the end of the implementation phase, the following were planned to have been completed:

- The web application is deployed publically via Firebase hosting, replacing the existing physical Idea Wall in its entirety.
- Permissions and database rules are established and approved.
- The database rules allow for CRUD (Create Read Update Delete) operations.
- Monitorisation features are enabled and configured for the web application via Firebase analytics.
- Reporting is enabled and accessible from the Firebase administration console.
- The Firebase project is wholly migrated under the Venntifact Admin Google account, providing complete access to any element of the solution.

Evaluation against requirements

Unfortunately, not all requirements could be met within the established time frame as a result of an increased client workload before the Christmas holidays. This was an identified risk and partially mitigated through the fact that a prototyping development methodology had been chosen, allowing for the core functionality of the application to have been completed.

The requirements which were not met include the Auditing capabilities and Management tooling. These were not core functionalities, but instead, features that would have provided additional management capabilities. Furthermore, custom integrations were not prioritised as it was determined to be an added benefit that would be nice to have, but required a significant increase in workload to implement.

Many of the core features were completed on time and to the standard that was expected, including:

- Trackable idea progress
- Extensibility and scalability
- Support for large scale projects
- Centralised accessibility via Online Web Application
- Categorising and organisation of ideas

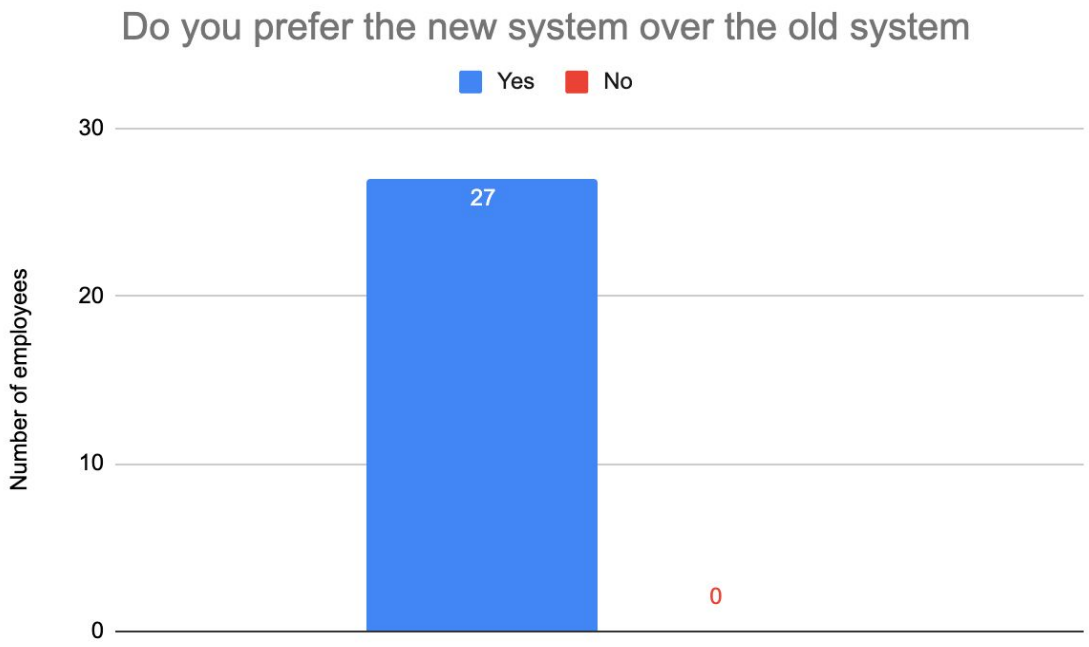
Some features were improved upon from initial conception, such as the progress tracking, which now includes creation dates, due dates, minimum and maximum hours and a progress bar, rather than its initial "Done" status.

Analysis and Discussion

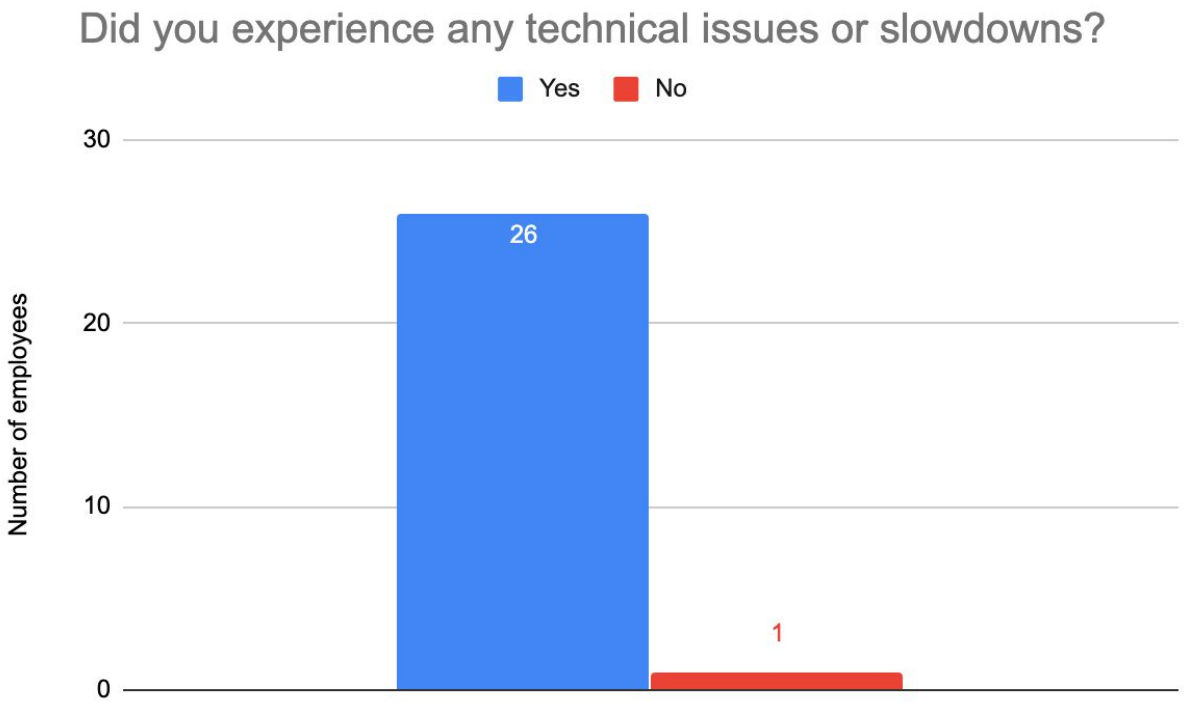
Validation Technique

To assess the performance of the new solution, a survey was conducted five days after the solution was implemented. This allowed for sufficient use of the solution before feedback was gathered. The survey was created using google forms, and results were collected anonymously to ensure the analysis was without bias. The survey was sent to all 31 employees of Venntifact via email, of which 27 completed. Additionally, a comment section was attached to the end of the survey where the user could provide further feedback. Results from some of the questions in the survey are discussed below.

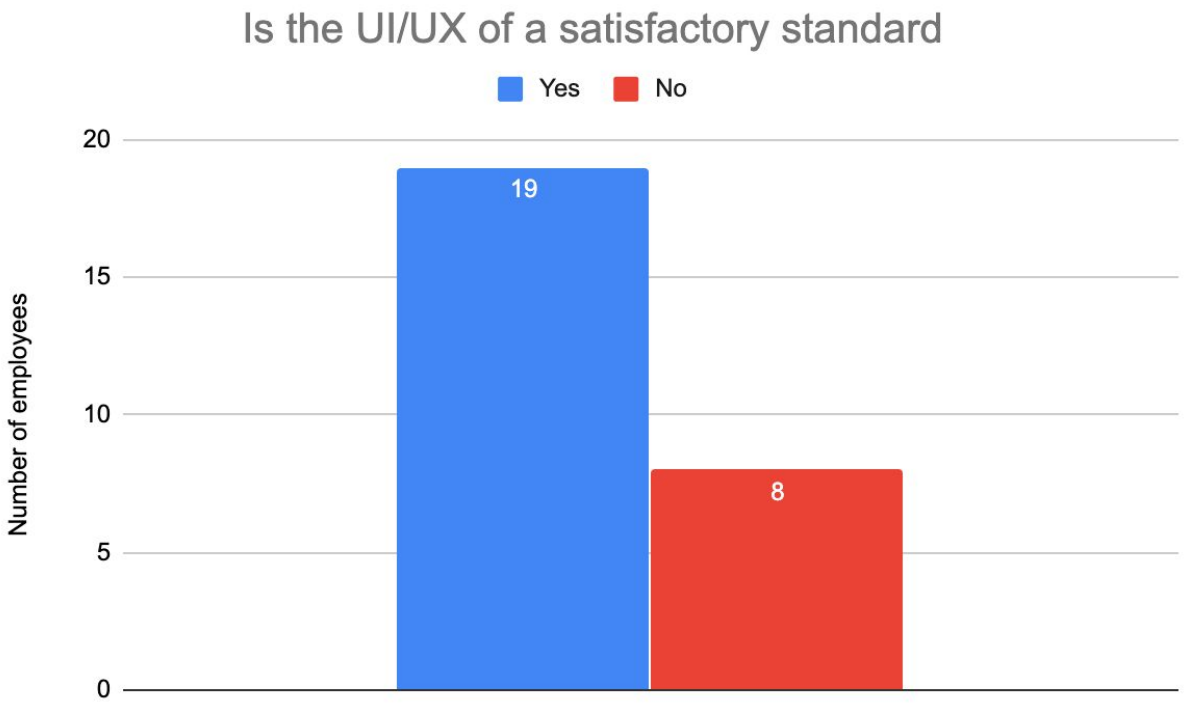
Outcomes



The first question of the survey was to determine whether the solution was a success overall. The results were overwhelmingly positive, with 100% of employees preferring the new system. This result was expected, seeing as the old system was unused and deprecated.



The next question was used to determine the capabilities of Firebase. Predictably, it performed exceptionally well with only one user experiencing technical issues or slowdowns. This was reassuring to know that Firebase's free tier was capable of supporting the solution requirements and size of the user base. No comments were found related to technical issues or slowdowns, so the reported technical issue has been ruled off as an exception.



This question garnered the most mixed responses with 8 out of the 27 employees disagreeing. Upon looking at the comments, many users stated that they felt the UI was to "cramped" or felt "unfinished", which was completely valid and true. This was one of the main concerns discussed above, as I realised the challenges that come with creating a robust design language.

What could have been done better

Time management was one factor that impacted the final solution. Creating the user interface and user experience for a full-fledged web application proved much more complicated than initially anticipated, as mentioned in the development method section of this report. Using the Tailwind CSS framework allowed for flexibility when it came to designing the application, but, as it was my first time developing an interface, a lot of effort was put into researching user interface

design and creating a practical design language. This negatively impacted the delivery of the solution, preventing me from completing the application.

Significant time was also spent learning typescript. It was the first time I had used a strong-typed language for a codebase of this size. However, I believe the benefits that come with a strong-typed language, especially for large scale projects such as this, greatly outweigh the time it took to learn. Typescript ensured the application is maintainable, which is of far greater organisational value.

Conclusion

The project failed to reach the final build due to time constraints. Though not all requirements were met, the function of this report was to perform and document the process of requirements elicitation, solutions research, prototyping and analysis, rather than build the final solution. The continued evolution of the prototype, once ownership is handed over to Venntifact, will hopefully address any missing features mentioned above. From the investigation, a clear solution was identified by analysing the old system, gathering user feedback and additional research of related concepts. These helped develop the digital web application that is now used as Venntifact's custom-built, internal project tracker.

Now that the prototyping development cycle of the project is completed, Venntifact should shift to an agile development methodology and gradually build upon the solution, adding additional functionality and increasing the overall quality of the solution. This can be achieved by assigning a small team to maintain the codebase. Should Venntifact wish to build upon this project, they should consider building custom integrations with other third-party applications, or incorporating other services Firebase offers, such as notifications and file storage.

The previous Idea Wall failed to provide an effective method for staff to propose innovative ideas and present constructive criticism. With many benefiting from its replacement, Venntifact's decision to convert entirely onto a custom digital platform that supports scalability and further development, has better enabled internal growth and their ability to adhere to their core values of transparency and introspection.