Akatsuki 7 Pitch

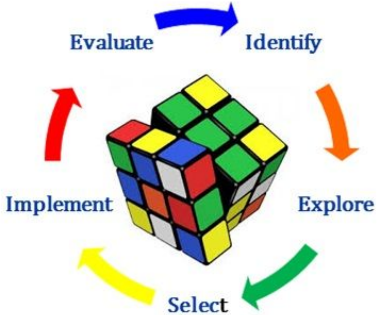
Problem - Identification

Imagine…

- You’re a new developer, just getting onboarded. You’re given a lot of files related to the project and have no idea how to get started or to find the information you need.
- You are following the iron triangle like a good CSE 110 student and want to balance between time, cost, and quality. You want to determine a reasonable timeline for completing the project with optimal quality.
- You’re a developer working on a large project. The division of labor is unclear and there is a lack of communication on what’s being completed.

Problem - Analysis

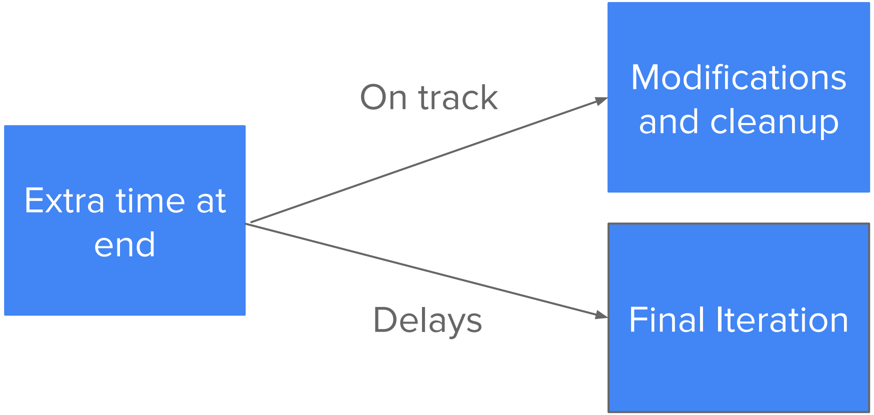
Problem Themes	Possible Solutions
Retention and organization of work information	Search and filter tools, file storage
Time management	Calendar, reminders, dated todo items, chart
Coordination/Division of Labor	Shared todo items, notifications on completion, space to assign team members, diagram
Organizing workflow	Calendar, reminders



Appetite - Time

The first thing we want to do is look at our time constraint - 5 weeks.

Since this is very limited, we plan for new iterations every two weeks. This allows for extra time in case of any unforeseen delays. In the case of no delays, we can make modifications and cleanup, or produce a final iteration.



Extra time at end

On track

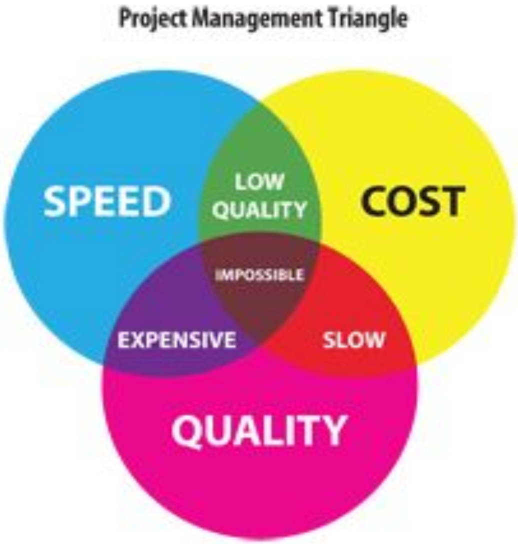
Modifications and cleanup

Delays

Final Iteration

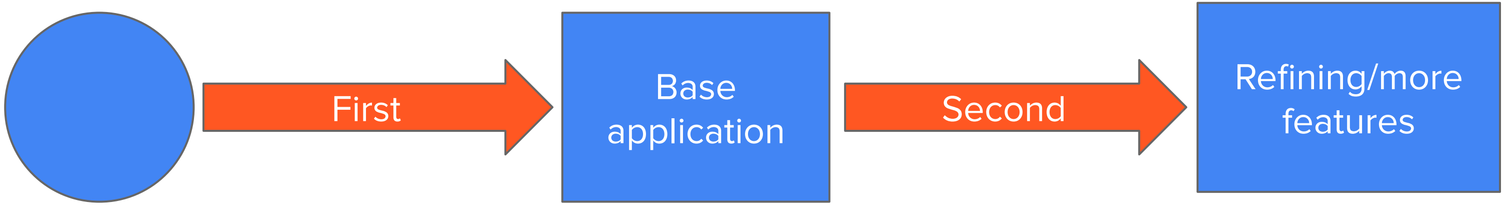
Appetite - Cost

All applications we plan on using will be free, so we do not anticipate any cost in this project.

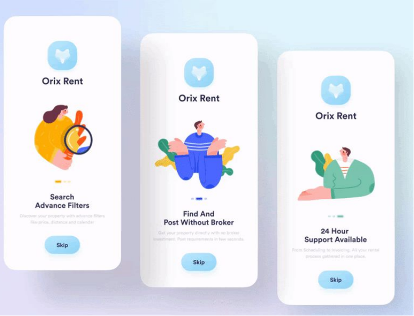


Appetite - Scope

The next thing we want to look at is scope. Due to our time constraints, we want to limit our **first iteration** to fundamental components for our application **(MVP)**. This allows us to add/modify components in our second iteration. This approach enables us to avoid having one monolithic iteration where we try to add all possible features and end up with a poor quality application.

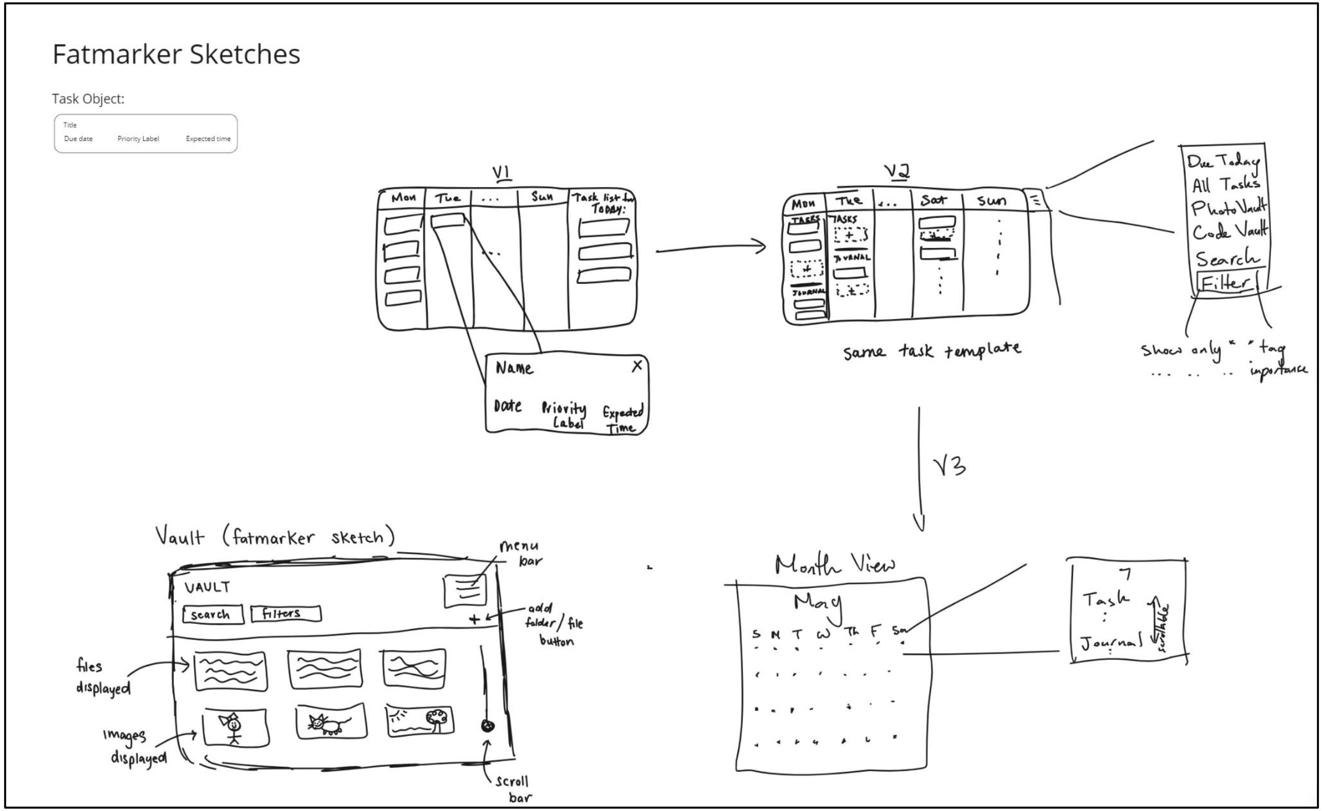


Solution - Brainstorming Features & User Personas



Persona	Attributes	Desires/doesn't want
Freelance Developer	- Fresh out of college, looking for work experience - Working on Projects to learn more	Wants to keep track of code snippets Keep track of how much time current tasks will take Wants an easy way to see how much time they are taking for tasks
New Hire	- Thrown with a lot of onboarding docs - Learning the corporate coding environment	Wants to keep track of API Keys, testing and ssh documentation Wants to reflect on the day to day workflow at their company Doesn't want a complicated interface to find items
Developer/CSE 110 Student	- Learning JS for the first time as well as SWE Principles	Keep track of Professor Tidbits in class Reflect on how the project is going for them throughout the quarter Use planner in tandem with other classes to budget time

Solution - Consolidation

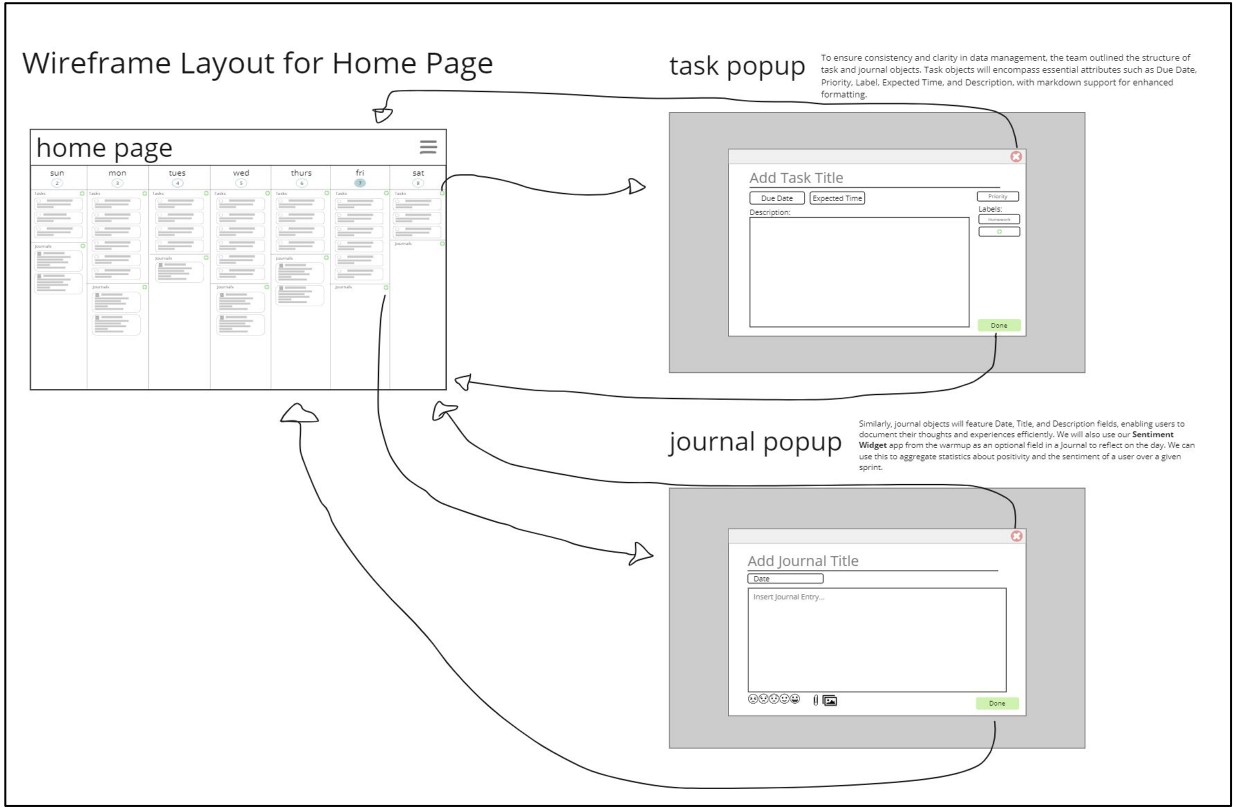


Fatmarker Sketches

Task Object:

Title
Due date Priority Label Expected time

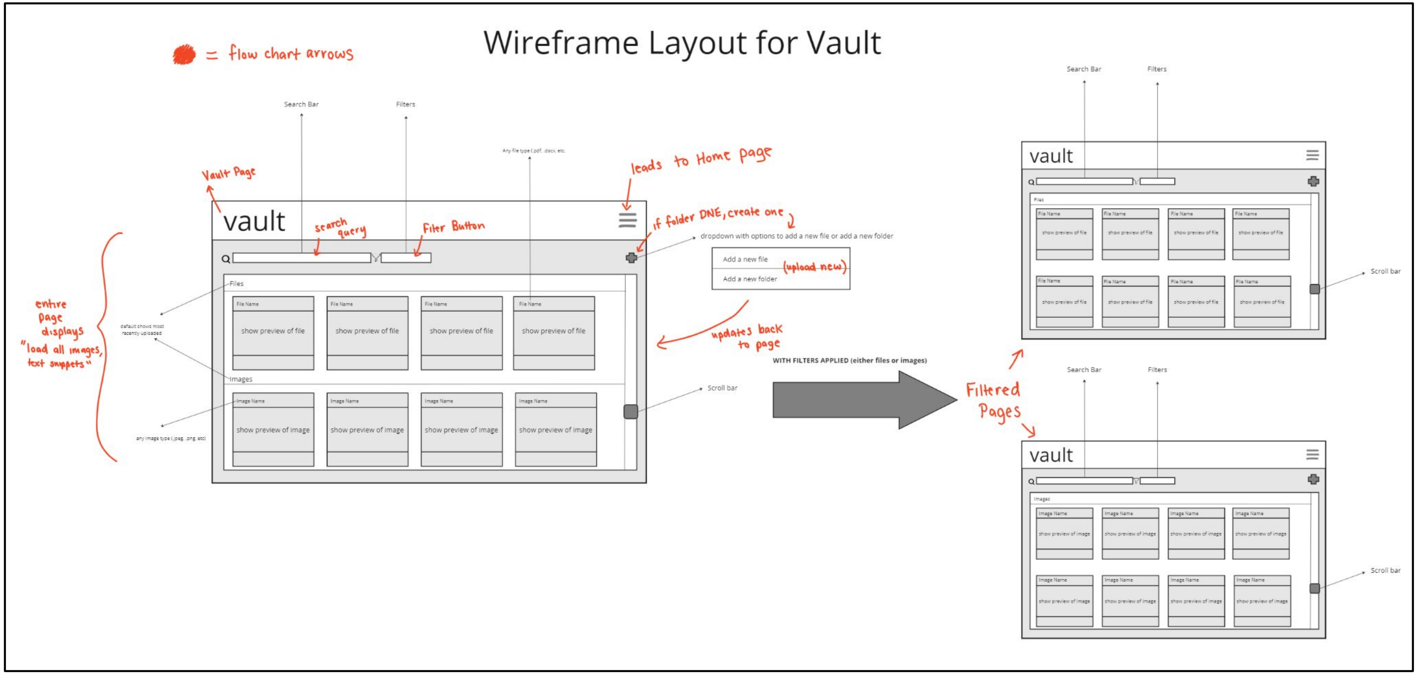
Solution - Consolidation



To ensure consistency and clarity in data management, the team outlined the structure of task and journal objects. Task objects will encompass essential attributes such as Due Date, Priority, Label, Expected Time, and Description, with markdown support for enhanced formatting.

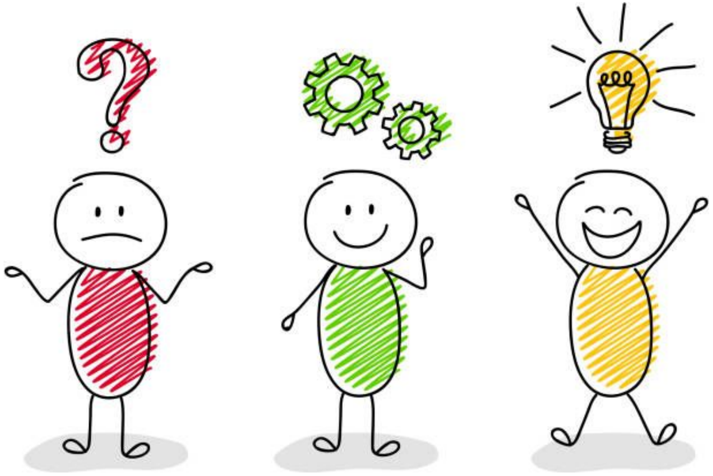
Similarly, journal objects will feature Date, Title, and Description fields, enabling users to document their thoughts and experiences efficiently. We will also use our **Sentiment Widget** app from the warmup as an optional field in a Journal to reflect on the day. We can use this to aggregate statistics about positivity and the sentiment of a user over a given sprint.

Solution - Consolidation



Solution - Statement of Purpose

Our software aims to **streamline and enhance the productivity** of software developers by providing a **comprehensive platform** for task management, journaling, and code snippet storage, fostering efficient workflow and effective documentation of their work processes.



Risks & Concerns

The Vault feature is a risky, complex feature to add since it deals with files stored locally on the client.

Since we've decided that it was a required feature, we downsized the scope of the feature so that it stores and searches **specifically text and image** files in a **separate, designated folder**.

This is an example of an **addressed risk** - we wouldn't want to spend too much time on problems that may arise

Another risk is the dependency feature as it involves multiple users and multiple devices.

We took a different approach with this risk, as it is deemed not to be a required feature. This is **left for the second iteration** to see how much progress can be done on it.

This way, if the feature proves to be too complicated, the product can still ship without the feature being completed.

Limitations

There are three primary features that we aim to avoid.

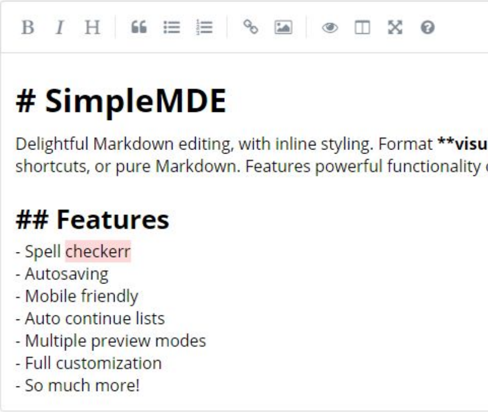
Year view of calendar	● Makes tasks too small to see ● Not much benefit ● Clutters Page
Vibrant text and background color	● Very distracting ● Feels too much
Over-complicated UI	● Calendar should take as little clicks as possible to complete a given task ● Too much would drive users away

Packaging + External Dependencies

Electron.js provides ease of access to a user's **local file system**; without needing to go to the cloud. Local Storage is not **persistent** enough to store files, and the File System API is only supported by Chrome.

This is a possible *risk* that we are taking, since the core functionality of the app can be extrapolated to a website, should packaging via Electron not be feasible, we will deploy our app on the web as a **website** (hosted by Github).

Markdown Editor: we have not decided what framework to use yet. **Since the notes stored should be** in Markdown format, we have decided that it is *infeasible* to develop our own Markdown Editor in a 4-5 week timeframe. We will use an external tool such as SimpleMDE or CKEditor and wrap that in our pop-up for tasks/journals.



SimpleMDE

Delightful Markdown editing, with inline styling. Format **visu
shortcuts, or pure Markdown. Features powerful functionality

Features

- Spell checkerr
- Autosaving
- Mobile friendly
- Auto continue lists
- Multiple preview modes
- Full customization
- So much more!

THANK YOU
THX