Main Problem: In today's increasingly collaborative landscape of software development, the ability to articulate and present one's ideas is essential for creating software that is not only accessible but also effectively communicates its purpose to both oneself and others.

We want to address the ease of documentation and reduction of the need to take action in order to do so.

As personal passion projects increasingly compensate for limited professional experiences in the workforce, developers require a straightforward method to showcase their projects. Simultaneously, recruiters seek an easily digestible format to grasp the essence of these endeavors.

Our aim is to narrow the comprehension gap between these two user groups by offering a single, user-friendly software solution that is accessible to all.

Research:

Existing Solutions (competition) & Features:

- Notion (lot of work on the user i.e. copy pasting and creating structures)
    - i.e. https://lahrry-jeong.notion.site/Bella-s-Workspace-4e43dfc32c3c47c4a0777dcd012455f2
- Command line apps: https://opensource.com/article/18/3/command-line-note-taking-applications (not sophisticated enough)
- Github (lacks workflow and personal thought process)
- Personal Websites (Takes time to create and host)

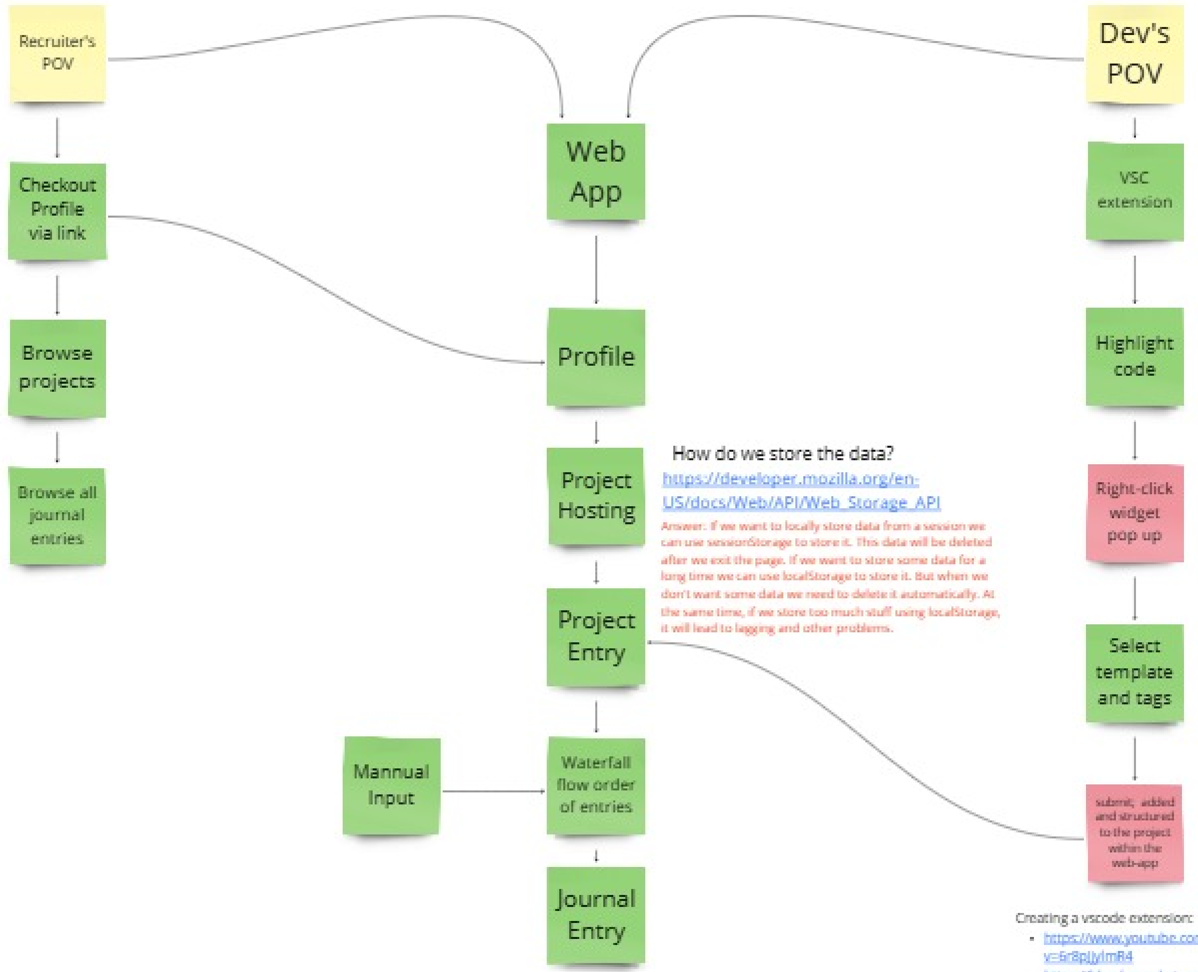
Solution Ideas:

- Easy exporting (pdf, md, etc.) to others
- Co-Pilot type of widget/extension that works alongside your coding environment
- Templates (retrospective, sprint review, decision logs, bug review, etc.)
- Place to swiftly store temporary, or commentable code
    - i.e. Highlight the code, press a button, automatically disappears and is saved in a dropdown in the IDE in which you can press it and it appears.

- highlight code, click a button to open widget, use widget to select tags and add text to a journal entry, generates a journal entry based on preset templates, is presented by the software to be easily accessible and understandable by practitioners and recruiters (**project resume in the form of legible journal entries**)

Feature Selection:

- Web App
    - Hosts projects (project homespace)
    - Each project features a waterfall flow of journal entries
        - with it showcasing a developers progression through their development stages
        - i.e. Templates (Code/feature additions, research, design, retrospective, sprint review, decision logs, bug review, etc.)
            - can mannually input (without code from ide) into the web-app
        - Formatted with templates, that are legible to recruiters and always editable by the input user
- Developer User's POV
    a. Dev highlights code snippet on IDE
    b. right clicks and selects our widget
    c. widget pop-up appears
    d. allows for user to select template and tags (similar to labels on github) *this will determine where it falls within the workflow*
    e. user is prompted with a selection of user input textboxes (such as "What we could improve on", "why we made this change", etc.)
    f. clicks submit and is automatically added to the project within the web-app
- Recruiter's POV
    - click on profile link
    - browse (simple)
    - Get's to see your development process and thinking process as well as all the necessary documentation/code


Recruiter's POV
Checkout Profile via link
Browse projects
Browse all journal entries
Web App
Profile
Project Hosting
Project Entry
Waterfall flow order of entries
Journal Entry
Mannual Input
Dev's POV
VSC extension
Highlight code
Right-click widget pop up
Select template and tags
submit; added and structured to the project within the web-app

How do we store the data?
https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API
Answer: If we want to locally store data from a session we can use sessionStorage to store it. This data will be deleted after we exit the page. If we want to store some data for a long time we can use localStorage to store it. But when we don't want some data we need to delete it automatically. At the same time, if we store too much stuff using localStorage, it will lead to lagging and other problems.

Is it possible to make a pop-up widget in vscode? (vscode extension?)
Solutions:
- add line in context menu (right-click menu) to generate pop-up
- runs script to open web-app page for editing instead of a pop-up

Answer: yes, we can create an extension on vscode and add the pop-up widget
(need to evaluate if this is doable in less than 5 weeks)
https://www.freecodecamp.org/news/making-vscode-extension/

How can we display choices within the pop-up?
How can we develop the pop-up? (What language?)
Answer: We can use html/css/js and json

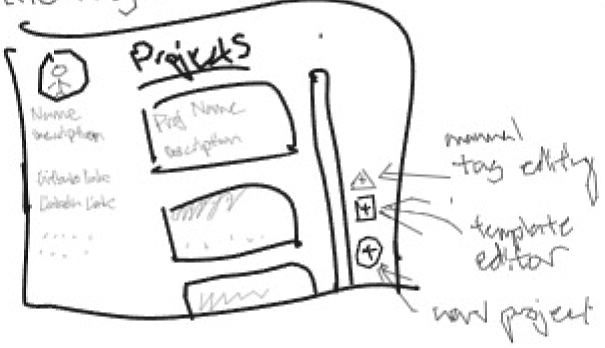
In what way can we make it so that the widget can directly communicate with the web-app?

Creating a vscode extension:
- https://www.youtube.com/watch?v=6r8pJjyImR4
- https://blog.logrocket.com/writing-vs-code-extensions-in-javascript/
- Can be developed using JavaScript or TypeScript

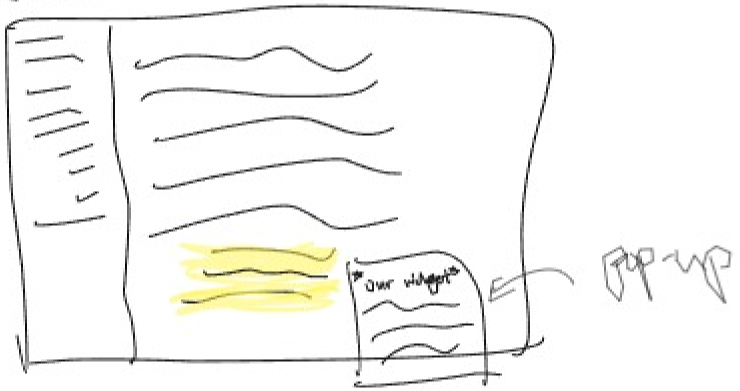
web-app

IDE

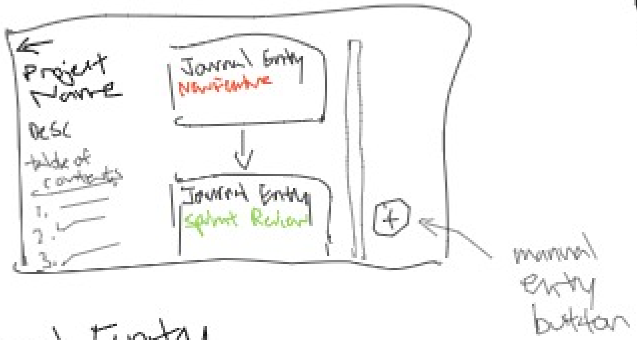
Profile Page



VS code

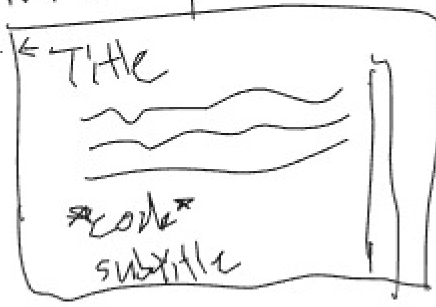


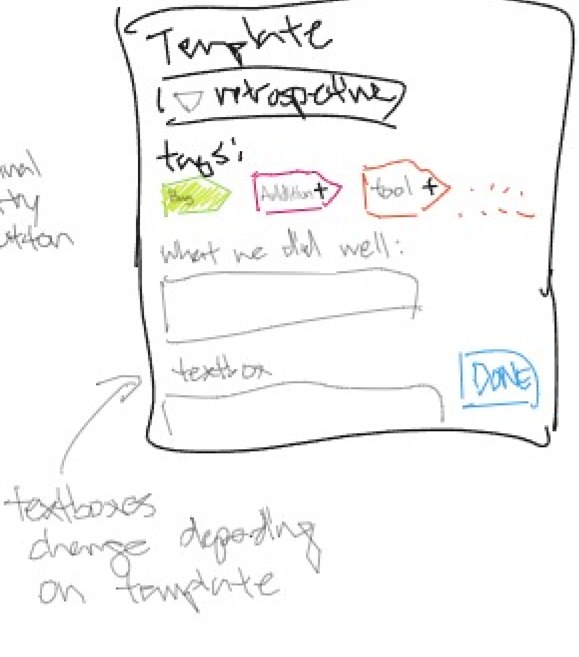
Project Page



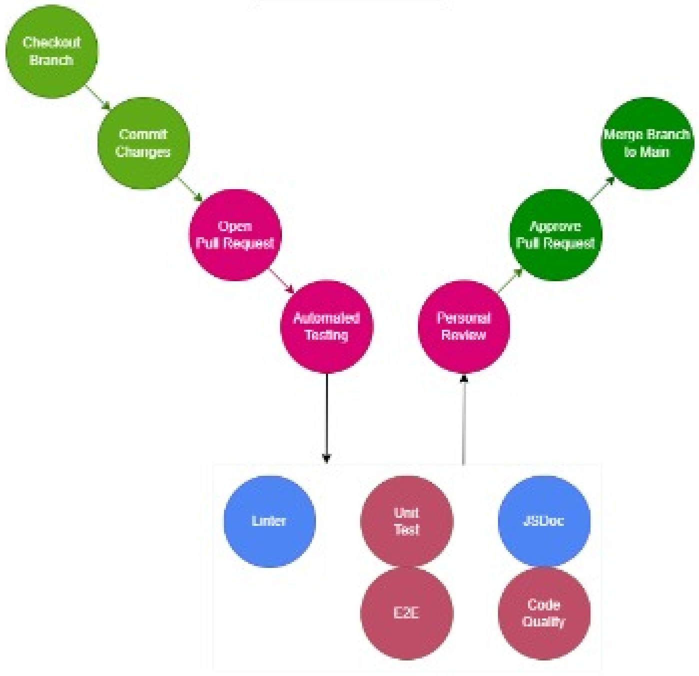
Pop-up Widget

Journal Entry





WORKFLOW



Super Linter

The goal of super-linter is to help you establish best practices and consistent formatting across multiple programming languages, and ensure developers are adhering to those conventions.

JSDoc

JSDoc automatically generates documentation of JavaScript functions with appropriate comments.

Reviewer

Every pull request requires one review before ability to merge to main branch

CHECKPOINT 1

ADD UNIT TESTING

- Add unit testing of most JS functions
- Add tests progressively, as the functions are developed
- Ask the developers for features and expected results
- Tests are run after every merge to main branch to ensure correct documentation

HUMANISTIC TESTS

- Add unit testing of most JS functions
- Tab Through Accessibility
- Works correctly on mobile, desktop, tablet
- Functionality Without Javascript

ADD E2E TESTING

Tests that covers basic User interactions:

- Journal Entries
    - Add journal entry
    - Edit journal entry
    - Delete Journal entry
    - Public or Private Entry
- Tags
    - Add custom tags
    - Add multiple tags
    - Delete custom tags
    - Change tag color
- View Project journals
- Navigation bar
    - Settings Tab
    - Additional tabs
- Refresh
    - Journal entry tests
    - tag tests
    - edit template tests
- Filter projects
- Navigation bar
- Templates
    - Add Template
    - Edit Template
    - Delete Template
- Update User Profile
- Refresh and test change in data
- Toggle public and private
- Test Multiple Tags

5/1 Project Concept
Chosen option: Template Journal.
Reason: It is something that we would use, it is easier to fill out dev journals when we are given the template. The web app provides a clean and organize place to showcase the projects and workflow.

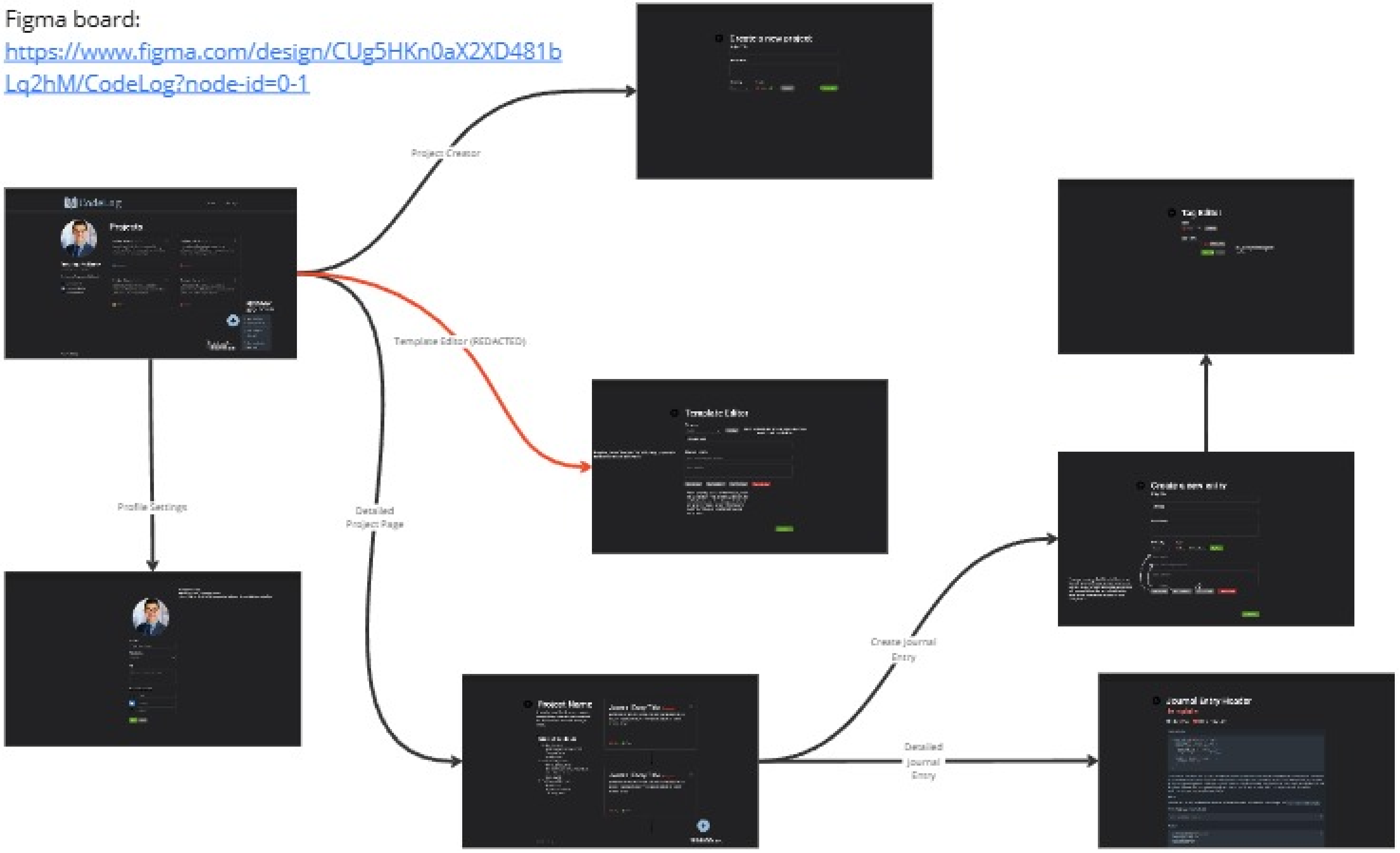
5/8 CI Pipeline
Chosen option: Super-Linter, JSDoc, requires one reviewer, E2E, Unit testing

5/8 Project Features

- main page containing all projects
- ability to add project and journal entry on main page
- ability to view project in detail, that showcases a list of all journal entries
- journal entries are categorized by tags
- ability to add and edit tags
- adding journal entry requires selecting a template or creating a custom template

5/8 Tech Stack
Chosen options: HTML, CSS, JS.
Reason: We will focus on developing with these three, if we ever need another piece of tech we will communicate with our TA and ask for permission.

Figma board:
https://www.figma.com/design/CUg5HKn0aX2XD481bLq2hM/CodeLog?node-id=0-1



5/13 Database

Chosen option: Hugo framework

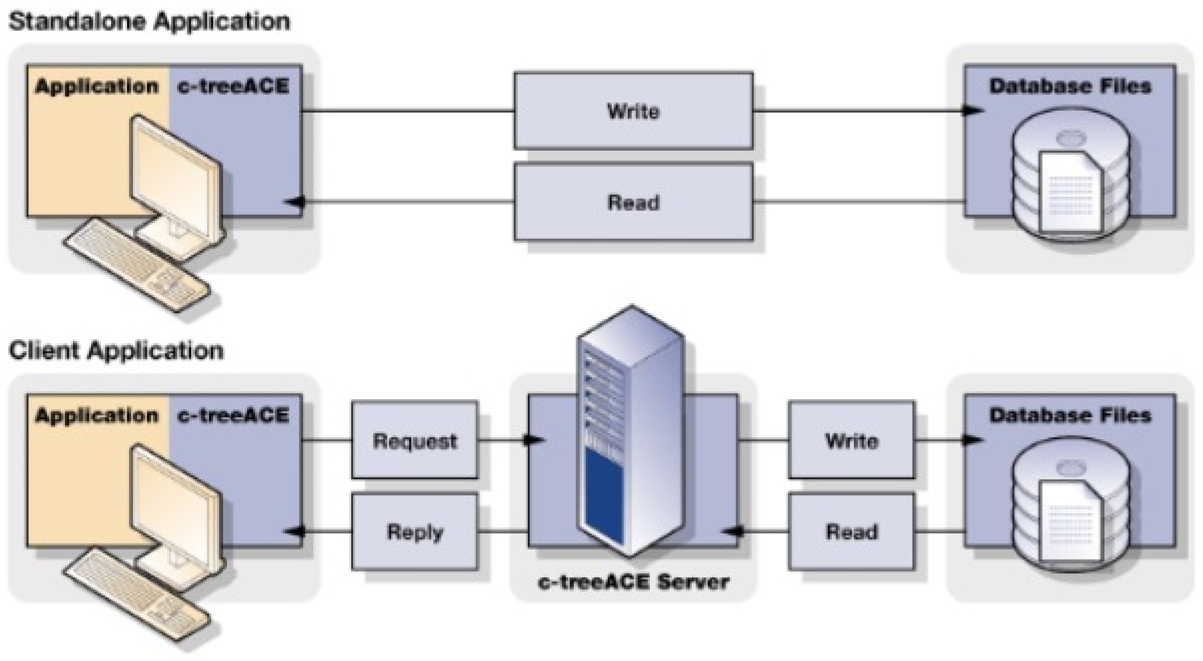
Users can make changes to their journal locally and the data will be stored in localstorage. Once they finished and saved their journal, it will be saved in a JSON file. Then we will run Hugo to convert this site into md and host it with GitHub Pages. This allows local changes to the journal without the need of networks, but also allows other people to browse your journal online through GitHub Pages.

5/13 Data-Structure

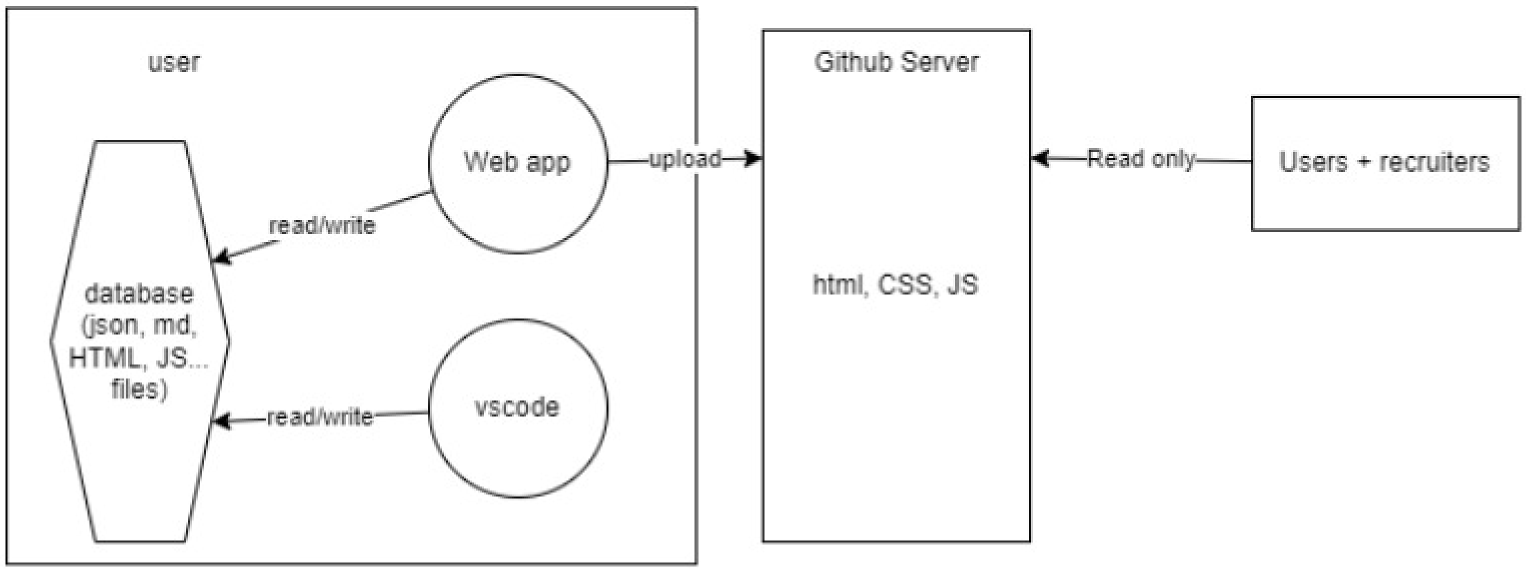
1. Chosen option: Store projects in an array of objects.
    a. Each project object contains an array of journal entry objects.
    b. Each journal object contains tags, content, and other information.

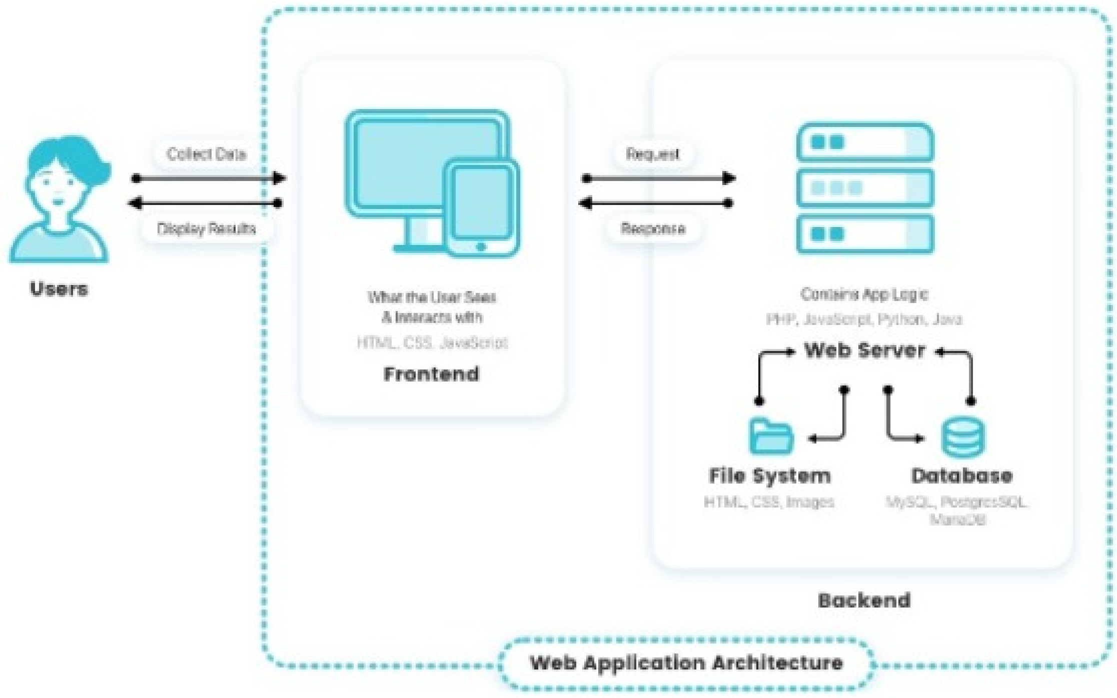
Because this option groups all entries of a project together, so they are easier to access and helps with turning the journal data into a Markdown file to be displayed.

local-only app vs client-server app



Github page + Hugo Framework, good for static websites





typical webb app