MASHED

PROJECT BRAINSTORMING

CUSTOMER: The professor.

> "*As a professor of software engineering my needs are that I would like to develop a project with a high enough quality that it can be used as a demonstration of what should be done.*"

PURPOSE: control.
STAKEHOLDER: Professor wants to use our project as a potential project he could assign to a class of undergraduate students.
NEEDS: Project of good quality to be used as a demonstrative example for an undergraduate class, and maybe make undergraduate students recreate it.

Team Skills Accumulated

Frontend	Server/REST	Backend	Databases	Hosting	CI/CD
HTML	Node.js	Java	Mongo SQL	GCP	Github Actions
CSS	Flask (Python)	Python	Google Firebase / Firestore	AWS	Travis CI
JavaScript		C++	Heroku PostgreSQL	Azure	Git Pull Requests
React.js				Heroku	
Angular.js				GitHub Pages	

Observations:
- Backend can be any language. What matters is how many components we want to include into the project.
- SWE Tools (CI/CD pipeline) learning curve for everyone since we don't have experience.
- Specific technologies TBD once we finalize a project idea
- As long as there is someone on the team that knows the language, everyone else can pick it up fairly quickly
- Possible limitations: not everyone might feel super comfortable with every tool

Time Constraints

6 person team
- Willing to spend 5-10 hours per week
- Around 7 Weeks of development
- Total number of hours to develop → from 35 to 70 hours per person
- Total team hours → 210 to 420 hours → 1 week to 2 weeks of development time

MASHED

General Timeline (from Syllabus)

Week 3 - Initial Tooling, Group "Storming Phase"

Week 4 - Initial CI, Design Docs

Week 5 - Idea Presentation,

Week 6 - Second CI phase, Mid-Quarter Retrospective

Week 7 - Alpha Presentation and Critique #1

Week 8 - Third CI phase,

Week 9 - Beta Presentation and Critique, Final Push Plan,

Week 10 - Final Check-in and opportunity for change

Project Management

Requirements: clear constraints.
Artifacts:

- User stories/requirements
- Design documents
- Build pipeline
- Clear set of tools
- Clean repository
- Internal and external documentation
- Pyramid testing

[Link Trello Board] -- Kanban, for project management

When we create a new feature:

- Updated documentation as we create the new feature

Code review:

- Pull requests
- Need at least 1 reviewer for the code request
- Linters?
- How to document methods and comments?

MASHED

PROJECT IDEAS:

CS Education - Algorithm Visualization App

High Level Description:

- A web application for visualizing how different graph traversal (mvp) and other algorithms will work.

Motivating example:
https://clementmihailescu.github.io/Pathfinding-Visualizer/

Feature Set:

- Virtualize path-finding algorithms in a 2-D grid world (BFS/DFS/A*/Dijkstra's)
- Virtualize search/traversal algorithms in a 2-D tree (Binary Search, Pre/Post/In order)
- Virtualize sorting algorithms (Quick Sort, Merge Sort, Bubble Sort...)
- Virtualize tree building algorithms (Binary Search Tree, AVL, RBT)
- Show progress in different timesteps (per loop, per unit exploration) using a slide bar
- Random quiz (time complexity, next step)
- Randomly generated worlds

Reasons why this would be a good control project:

- For students, it will be a good tool to learn and review the fundamental CS algorithms
- For TAs, it will be a great teaching tool to educate students
- For implementers, it will be a good practice to revisit the algorithms and build tools (website applications) for CS educators
- With a constrained set of data structures, the feature set is constrained

Type of testing we would want to have:

- Unit testing
- Integration testing
- UI Testing

Risk assessment:

- Have to learn some visualization libraries like D3.js (Or use graphviz to generate pictures and just render these pictures one by one)
- Have to write complex animation CSS code to render these data structures
- Have to know all the algorithms in and out.

Project Timeline and milestones to meet:

- Week 4 - Setting up development environment, starting thinking and coding
- Week 5 - Implement one or two types of visualizer of simple algorithms with basic UI
- Week 6 - Testing (Milestone ONE)
- Week 7 - Implement one or two more types of visualizers of other algorithms with additional features; improve UI/UX

MASHED

- Week 8 - Additional testing, prepare for deployment (Milestone TWO)
- Week 9 - Implement additional features with testing, ALPHA/BETA tests
- Week 10 - Final testing and project deployment (Milestone THREE)

Type Racer

High Level Description:

- It's a typeracer website that lets users measure how fast they can type in terms of words per minute (WPM). We plan to provide customizable settings and UI to give users a good experience.

Motivating example:
https://play.typeracer.com/

Feature Set:

- Type selected paragraphs and record the time you use (show the WPM)
- Settings for users to control what they want
- Maybe different languages?
- Custom UI (dark mode)
- A ranking board with users' name and speed

Reasons why this would be a good control project:

- Frontend components are simple (Textarea, Card).
- No complicated backend.
- It's a fun website to build.
- Building a real world website is probably something they haven't done yet.

Type of testing we would want to have:

- Unit testing
- Integration testing
- UI Testing

Risk assessment:

- Have to get a library of texts (or multiple libraries for different languages)
- Have to create a simple, clean and beautiful UI
- Have to provide customizable settings on the UI and typing.

Project Timeline and milestones to meet:

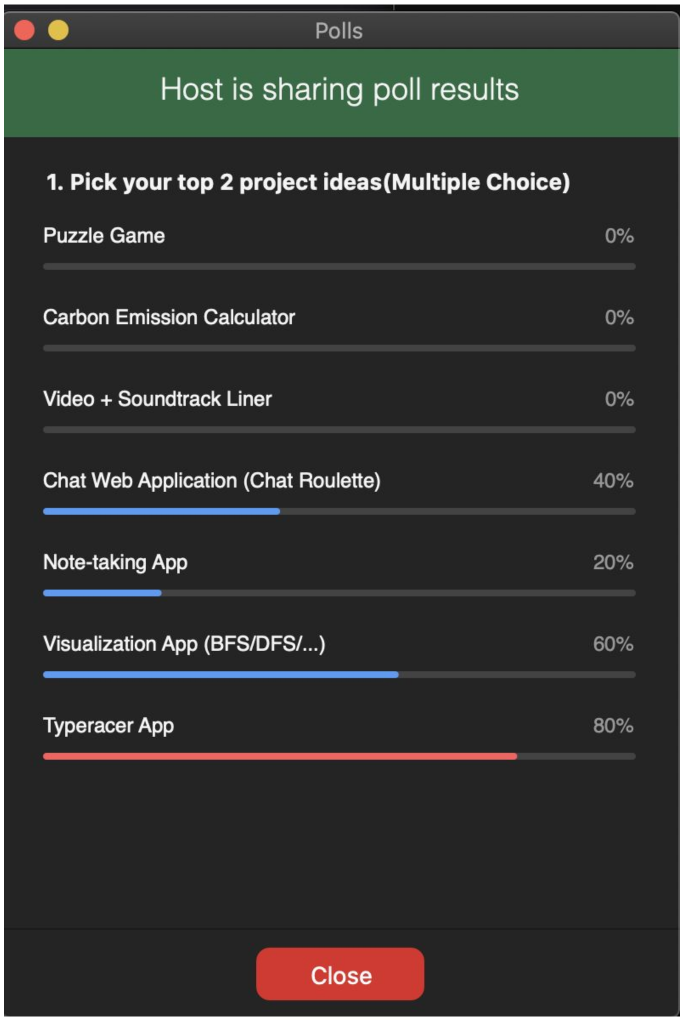
- Week 4 - Setting up development environment, starting thinking and coding
- Week 5 - Create a basic UI that shows the texts and implement basic typing/recording.
- Week 6 - Testing and improving the UI (Milestone ONE)
- Week 7 - Implement customizable settings on the UI
- Week 8 - Implement customizable settings on the typing (Milestone TWO)
- Week 9 - Add support for multiple languages? Additional features

MASHED

- Week 10 - Final testing and deployment (Milestone THREE)

Backups

- **CS Education - Puzzle Game**
    - Heavier on the front-end, back-end relatively easier
- **Carbon Emission Calculator**
    - Dependency on external APIs
- **Automated Video + Soundtrack "Liner"**
    - https://www.youtube.com/watch?v=0ZeO0IQaJ-A
    - More complicated than others
- **Chat Web App (Chat Roulette)**
- **Note-taking App**



Host is sharing poll results

1. Pick your top 2 project ideas(Multiple Choice)

Puzzle Game	0%
Carbon Emission Calculator	0%
Video + Soundtrack Liner	0%
Chat Web Application (Chat Roulette)	40%
Note-taking App	20%
Visualization App (BFS/DFS/...)	60%
Typeracer App	80%