Guidelines for Web Programming Projects

For the mini project you have to create a web application using the key technologies we have in the Web Programming course. Instead of giving you a specific web application to replicate, you are given the freedom to choose what you'd like to make.

We are giving you some restrictions on what you can make, such as a list of technical and style requirements. If you do not come up with any ideas for a web application you'd like to make, we will provide a *Dashboard Application* project idea that you may use. If you choose to create the *Dashboard app*, it will be graded in the same way the other mini Projects are graded.

The project should be done in groups, Size of group cannot be more than 3 members.

Project requirements

The mini project is mostly open-ended, with some specific requirements as outlined in this section.

Overall, you can receive marks on the mini project if:

- Your project meets the Logistical, Technical, and Style requirements listed below.
- You successfully deploy your site to Heroku.
- You turn in the completed Project Proposal on-time
- You turn in your completed mini Project on-time.

1. Logistical requirements

- This is a group assignment, you have to create a group of 3 students.
- The deadline for the mini Project proposal:
  - Mon, Feb 28 @ 11:59pm: Project Proposal
- No late submissions will be accepted .

2. Technology requirements

In your mini Project, you must include the following technologies:

Frontend: HTML/CSS

- Use classes and ids
- Use display: flex;
- Include a CSS animation or transition
- Change 2+ default font properties
- Change 2+ box model properties (border, padding, margin)

Frontend: JavaScript

- Write object-oriented JavaScript
  - Include 2+ ES6 classes
  - Use at least one callback or Custom Event to communicate between classes
- Listen for 2+ events
  - Can be the same event type, like 2 click events
- Use fetch() to talk to your backend

Backend

- Use Node and/or ExpressJS
- Save persistent data in some way using MongoDB
- Include at least 1 GET route

- Include at least 1 POST route, and it should include a message body
- Optional but recommended: include at least one route parameter

Misc

- Use async / await (or Promise) properly somewhere in your project.
    - This can be in the frontend, backend, or both.
- You need to incorporate all these technologies into one cohesive web app.
    - We will not be grading on things like how interesting your project idea is, how original your idea is, how "realistic" it is, etc.
    - However, your project also can't be a set of totally nonsensical code that happens to hit our checklist of technologies. Your project should be a web app, even if it's a very small web app, or a silly web app, or a dumb web app.
- Your project should be relatively bug-free.
    - It's hard for us to judge whether you are using the technology correctly if your code is very buggy.
    - We will not deduct points for trivial bugs or polish bugs, but we will deduct points for things like if the one CSS animation you included in your project is not actually working.
    - This is another reason why you should aim small in your project scope.

3. Style requirements

These style requirements should be no surprise, as we've enforced them all quarter.

HTML/CSS:

- Use tags semantically, e.g. don't use <div> for every single element on your page.
- No deprecated tags.
- Use descendent selectors to reduce redundancy in CSS and HTML
- You must write raw HTML and CSS, as we have done all quarter.
    - i.e. don't use SASS or compiled CSS/HTML

JavaScript:

- Must write object-oriented JavaScript.
- May include ES6 classes.
- Avoid global variables.
    - OK to use globals for constants, instantiating classes, or other reasonable scenarios
- But you should not put anything in a global variable that could be better encapsulated in a class
- Add/remove/toggle classes from *classList* instead of modifying style directly, unless you have to modify style directly (e.g. if you are calculating the value of a CSS property in JavaScript)
- Communicate between classes using callbacks or custom events.
- You must write raw JavaScript, as we have seen in other assignments.
    - You may or may not use jQuery.
    - You may or may not use any frontend frameworks, such as React, Angular etc.
    - You may query a 3rd party libraries.

Backend:

- Your backend may be written using the Node and Express libraries
- Don't save data to the filesystem: persistent data should be stored in a MongoDB.
- HTTP methods should be used in ways that are compatible with the method definition. For example:
    - Use GET for retrieving data. Do not write data in a GET handler.
    - Use POST for saving data. Do not use POST to display a page.
        - Don't use query parameters with POST

Publishing your site

Deploying:

- You must successfully deploy your website on Heroku for turn-in.
- For backend (MongoDB backend) you must use the mLab Heroku add-on.

Browsers:

- You must verify your site works on the latest version of Chrome.
- You do not have to make your site work on any other browser but the latest version of Chrome.

Project ideas

You need to decide what web app you will build for the mini Project. Here are some suggested guidelines for you to follow when deciding what you want to build:

Small and simple.

- You only have a fixed duration for this project, and programmers tend to grossly underestimate how difficult a task is / how long something will take to implement. Try to choose something very focused and simple for the subject of your mini Project.

No more than 5-6 different pages/screen in the entire app.

If in doubt, ask us

If you feel difficulty in selecting a project, ask us. We will provide you a project.

Project Proposal

The Project Proposal is due Mon 28 at 11:59pm.

1. Decide on a project idea

- Read the project ideas section of this spec and come up with an idea for your mini Project.
- On the turn-in form, you will be asked to write a 1-2 paragraph description of your project idea.

    - This doesn’t have to be too long or descriptive, and it doesn’t have to be a “pitch”; we just want to get a sense of what you want to make.
    - We may reach out to you if we think your project is too difficult or otherwise inappropriate for the mini Project.

2. **Deploy the starter code to Heroku for practice**
    - You are **required** to deploy your mini Project to Heroku. Practice doing so now so that we can work out any issues before the mini Project deadline.
    - **Submit your project proposal**

https://forms.gle/ehkJpqjScbAnkpCo7

<u>Note: Implementation Milestones will be shared separately.</u>