Final Project

# Overview

For the final project, you will be creating your own web application that you can hopefully show off to employers. You'll be working in groups of 2 to 3. Ideally, you do something you're passionate about, but if you're not one to come up with ideas, I have a few that you and your group can ask me about after class or over email. The restrictions are mainly in the scope of the project since it's near Finals time,

# Due Dates

* Tuesday of Week 7 - Project Proposal
* Thursday of Week 9 - Backend designed and queryable. Demos will happen in class
* Thursday of Week 10 - Final Project due. No extensions given.

# Project Requirements

**Credit:** Most of these requirements aren't my own. They're from a [Stanford course](https://web.stanford.edu/class/archive/cs/cs193x/cs193x.1176/final-project/). I tried to shove the things that we learned in class into their structure and used their verbiage so if things don't make sense, please ask me.

Because the project is open-ended, this is the one assignment that's not autograde. You'll probably receive full credit if:

1. Your project meets the Technical, and Style requirements
2. You successfully deploy your site to "the server"
3. You turn in the Proposal (Week 7) and Demo (Week 9) on time.
4. You turn in the Final Project on time.

## Technology Requirements

To receive full points, you must include the following technologies:

### [50] FrontEnd: HTML/CSS

1. Use classes to style your content
2. Use display: flex for Laying out content.
3. Deep-Link to your content by adding ids to an element (like having a "Go to Top" button)
4. Change 2+ default font properties
5. Use a font from <https://fonts.google.com/>
6. Change 2+ box model properties (border, padding, margin)

### [50] FrontEnd: JavaScript

1. Use fetch() to talk to the backend
2. You must use Promises where available

### [75] Backend

1. Must be an HTTP server
2. Must use persist data either to MongoDB or a file (for half credit)
3. Include at least 1 GET route (not including static content)
4. Include at least 1 POST route that takes in a route parameter and serves JSON data.

### [25] Publishing

1. Your site must be deployed somewhere public and available on the Internet.
   1. You can use a class server that I've spun up and will give instructions on tying that server to your group.
      1. Since everyone's content will live on a single physical machine, don't try to do anything destructive to other people's work. I've tried to make sure things are as isolated as possible, but accidents happen. Keep a backup.
      2. Everyone has access to the exact same MongoDB instance. Try not to bring things down, but be sure to keep everything locally as well.
   2. If you'd like to use your own server, that's okay, please put it in the project proposal.

## Style Requirements

The following style requirements are where you might lose points where the technical requirements are where you will gain points.

### HTML & CSS

1. Use semantic tags, only use <div> and <span> for styling
2. No deprecated tags (like <i> and <em>)
3. No custom tags, only tags allowed are from [MDN](https://developer.mozilla.org/en-US/docs/Web/HTML/Element)
4. Use descendent selectors to reduce redundancy in CSS and HTML
5. Only write raw HTML and CSS
   1. No React/Vue/Angular/whatever, no CSS preprocessors

### JavaScript

1. Global variables are okay for constants and loading scripts, nothing else.
2. Toggle classes using classList, not using the style property
3. Raw JavaScript only
   1. No jQuery, no frontend frameworks, no utility libraries (like lodash)
   2. You can use 3rd party libraries that are necessary for your final project.

### Backend

1. Your backend must be written using Node.js and **not** using Express.js or any other server frameworks.
2. You can save data to the filesystem for partial credit if you can't get MongoDB to work
3. HTTP methods and Status Codes must follow convention
   1. Return appropriate status codes (like 200, 404)
   2. GET is only for retrieving data. Don't write data in a GET
   3. POST is only for saving data, not for retrieving a page of content. The return value of a POST should have any updated content from the backend.

### Project

1. All technologies must be used in a single cohesive web application
   1. I'm not grading on realism, interest, or originality but your project can't be a jumble of ideas that fit the requirements.
   2. You'll most likely have completed this after submitting your Project Proposal
2. Your project should be relatively bug-free
   1. Polish and trivial bugs aren't going to matter, but if your project doesn't fulfill one of the requirements above, then that'll be a problem.
3. **If you submit your node\_modules folder, I will deduct 50 points.**

# Contest

Just like in the Stanford course, there will be a contest for final project submissions. As a class, you'll vote for any project (but your own) on these categories:

1. Most beautiful
2. Most fun
3. Most interesting
4. Best overall

If you win a category, you get +20 points per category won. This is equivalent to dropping 1 homework. It can be applied to either the lecture or the lab.

# Project Ideas

Here are some guidelines lifted from the Stanford course.

1. Keep the idea small.
   1. Although you have a total of 4 weeks to finish, you have other classes, Thanksgiving, and finals.
   2. Don't go overboard in features, if you want to explore a specific technology, you can but don't feel pressed to. Your TA and I may not be experts in a specific technology but we'll try to help you as much as possible.
2. Somewhere between 2-4 distinct pages for your application is ideal.
   1. Do not build more than 5 pages.
3. Use the Labs as a guide for how long this should take. This should roughly be the size of Lab 5 + 6 combined.
4. If your application requires authentication, you'll need to use a separate MongoDB instance.
5. Make something you can show off or something you can use after the class. I'll keep everything up for about another month, but we'll go over in class how to provision your own website.

If you're looking for concrete ideas, come to me as a group after class or make an appointment and we can talk more then. Ideally you have some common interests, but if not I have some generic ideas. Don't look to be original if you don't want to. If there's an application that you think is fun and want to copy, that's a great starting place.

# Project Proposal Assignment

The Project Proposal assignment is to be turned in on Tuesday of Week 7. It takes the place of 1 Coding Assignment so you can *focus on finishing your lab*. When you turn your project proposal in, you'll need to submit the following information:

1. Who's in the group?
2. What is the overarching application you'll be creating?
3. What are each of the individual pieces of the project you'll be creating?
4. How will you be deploying your website (on my hardware or your own)?

Submit this as a text submission in Camino on Week 7

# Backend Demo Assignment

This is graded as a group and takes the place of your coding assignment during Week 8. Submit your code for your backend by the start of class on Thursday. To get full points, you **must**

1. Complete at least 1 GET endpoint
2. Complete at least 1 POST endpoint
3. Demo to me in class that both endpoints work

**If you submit your node\_modules folder, I will give you a 0 for this Homework assignment.**