Project Management

Wireframes, Mockups, and Sitemaps

So, yes, we have been calling them wireframes, when maybe they are mockups. For this project it would be a good idea to have both. You will need to plan out the path through the app, and also have some high definition 'mock-ups'. For those we will still use Figma:

- https://www.figma.com You can also use figma for the lofi wireframes.

For the site map this program looks pretty interesting (although I have not used it):

- https://mockflow.com/apps/sitemap/

Database designers

For the database stuff, I like this one:

- https://app.dbdesigner.net It's been enough for me in the past, but yes, it is not an offial ERD.

I'll try to have a play around with some of these ones too:

- https://erdplus.com/#/
- https://www.lucidchart.com/pages/examples/er-diagram-tool They seem to have more of the technical quality that some might prefer.

Trello (kanban)

Trello is a useful tool for keeping track of tasks. If you kick it off with 'To do', 'Doing', and 'Done', and then make some new column as they come up organically. When you have a heap for one aspect of the app you might make a new column. You can also assign priorities (should, certainly later as time gets tight), and link certain tasks to individuals.

- https://trello.com/

User stories

User stories help define what you hope your app will achieve for particular users. They have the pattern:

As a **user-type** I should be able to **do-something** (or similar)

You also can write up some detailed ideas of what types of people you expect to make use of your app. These are decent resources:

- http://www.agilemodeling.com/artifacts/userStory.htm
- http://www.mountaingoatsoftware.com/agile/user-stories
- https://www.atlassian.com/agile/project-management/user-stories

Git Collaboration

Some info (solid article): https://medium.com/@jonathanmines/the-ultimate-github-collaboration-guide-df816e98fb67 The challenge:

- https://gist.github.com/anharathoi/9fe9bd883831dbf167a99de1c1010e73

Example

This is an example app, and the README might be useful as something to aim at. It's a very thorough job.

- https://github.com/JoshTeperman/rails_coding_tutor

Some Extra Blah..

A few things that I'd love you to keep in mind.

- **Work on your weaknesses**. This doesn't mean that you must take over the writing of all of the models if that's something you find baffling. But you certainly can do some pair programming, or get a team mate to talk your through some of it. As things become clearer you might step in to take over for some of this. Of course you are welcome to play to your strengths at times, but the project is a learning tool above all else.
- **Include all of the members**. Obviously people are at different stages as we go through the course. But I would really love the strong members to work on including people in all of the discussions, and trying to bring people along as the project progresses.
- **Clean up your data towards the end**. Of course you can put in nonsense while you are getting everything up and barking. But later I strongly recommend you load the database with good images and data details. It makes a huge difference to the presentation of the app. Doesn't take too long to write up some solid seeds (or use a gem like Faker).
- **If you have decent functionality down, spend some time on styling**. Understanding front and backend is the main game here. But styling is important too, and a couple of hours to make things look a bit slick will take it to the next level.
- **Put a little time into the presentation**. I would plan out how you want to best show the app - the path you'll take, the login details you'll use, the aspects of the app that are most interesting. Have a run through and time it to get the feel for the timing of things, and also be careful to prioritise the information that you are presenting.
- **Collaborate across teams**. Help each other out. When you find some great NPM package, or a useful article, keep posting them, and also help each other out directly with code.