National Badger Meeting Notes

Meeting #1

Agenda

- We've been meeting each Tuesday as a group for a few hours
- Karleigh and Val worked this weekend to implement schema and routes
- Current problem — how to test the database and deploy (having problems running the server)
- update from Denis and James
- Created chart model, view, and controller

Questions for TA:

- client-side object while user is making chart → fine
- next Wednesday: "revised design" = ??? → design doc!
- do we have to write test suite? → yes! For mvp too! (test that we can get stuff and put stuff in db. Test validations)
- have to write documentation also
- readme

Progress

- Implemented schema and routes
- Made mockup chart editor UI

Minutes

What we need to do on client side (tl;dr, pretty much most things except for displaying and editing a chart)

1. Page where a user can decide the size of the chart
// haven't told chart_editing.html
2. For remixing, we can do one of two things:
   - send a chart over to chart_editing.html
   - or send an ID to chart_editing.html so that he can fetch the chart from server
3. For creating new chart either:
   - send a boolean across pages so that chart_editing.html will know to create a blank chart
   - over have chart_creating.html, which is a separate page
4. Actually fill in the requests so that we can populate the chart.
5. Create a profile page for charts

Feedback:

- Mongoose validators for values (model or client)
- Make UI more pretty!

==========

Meeting #2

Agenda

Questions for TA:

- Do we need tests before feature freeze? (nope! Final turn in)
- “Not also design choices have choices - you should make choices in your design instead of punting them until implementation.” (What does this mean?) (need to list options and eval them)
- “Hashing and salting passwords is not enough to protect user information” (What do we need for sufficient protection?)
- “You should also put every non-public endpoint (including JSON endpoints) behind authentication. (I have no idea what this means)
- “Immutability errors, e.g., the chart a comment is made on does not change.” (Is it sufficient to just explain that a comment can be changed?)
- When will we get feedback on MVP? (by early next week)

Progress

-Completed the MVP
- Core chart making functionality is there
- Backend has routes and schemas for charts and remixing them
- Frontend has the UI for making charts, remixing, and viewing a feed

Minutes

- Flesh out what report button is for-- populate database w/ administrators + email? Hide button to prevent abuse.
- Tests not needed until final turn in
- In revised design, design choices: write down what options were, make choice
- Need authentication for POST/DELETE/GET requests that require user logged in, etc. Things of that sort.

Meeting #3

Agenda

- Demo to TA

Progress

- Most of the functionality is present
- We’ve added all the backend (routes and schemas) for user accounts, login, following, liking

- UI in place for user pages, following, liking, viewing liked charts, etc.
- Several big bugs to work on

Minutes

-we took a stab at things, lots of bugs
-commenting,liking, parent find, nsfw, etc→ priority
-secondary: polishing (margins, standardized colors/fonts/spacing)
-maybe remove about page? Unless we have nothing to do
- START UNIT TESTING
-testing! Test every public function on server (edge cases!)
    -could also think about testing routes instead (using route testing framework-- may be slightly harder)
-if front end has significant logic, test it
DOCUMENT PUBLIC FUNCTIONS
- error handling, authentication (make sure that people can't edit desc/tags of charts not theirs)
        - Also. If these fields are editable, there should be some sort of indicator to the user that editing is possible!
-fix non-RESTful routes
 - /charts/43943282
 - req.params.id ←use in route
/GET users/
POST users/:myid/following to follow (pass in data about user to be followed)
    - Instance + static methods on mongo
        - Avoid including logic in routes
        - Put logic in model
        - this.model("freet") (see static and instance methods in mongo documentation
        - Ensure request is valid in route, if so: pass into model
        - In model callback function, send response to user
        - If all logic is in models, we only have to test our models