Mentor Meeting 11/18

Progress:
We've talked through high-level implementation and design together as a team and split up the work and which aspects of the project each person will be starting with. This will be subject to change based on how much progress each person makes over the course of the project.

We came up with a couple schemas to include in our implementation:

- Note-pitch, duration, time (possibly using midi)
- Sheet-owner (User), collaborators (list of Users), data (set of Notes)
- User-username (String), password (String)

We also planned out how concurrent editing will work on a sheet of music. Essentially, every time a person is adding, deleting, or editing notes in a certain section of the canvas in the UI, a lock request will be sent to the server. If they don't get the lock, then they will be denied access from editing that section of the music. This resolves the potential issues that could arise from multiple collaborators attempting to edit the same section of music simultaneously.

Agenda:

- Go over progress
- Ask for advice on the following issues:
    - Feedback on schema and database designs
    - UI implementation advice (using canvases vs. SVG)
    - Real-time updating: pulling entire document from server constantly vs. pushing deltas (changes) from server to users
    - Review concepts from design doc

Meeting Minutes:
--general implementation
logging in, general things
notes, sheet - kim & lisandro
user - stuart and jess

database design - similar to fritter
user
sheet (add schema to agenda)

using midi playback

implementation question (UI);

challenge--representation of the sheet

our own rep and then convert for midi only

questions:
real time updating (better to send entire state of doc or every few seconds send updates only)
--should be fine to go with the deltas
--not a huge concern if the app doesn't work perfectly, a lot of bugs isn't great
--UI testing not required, mostly test data models

--don’t need to test concurrency testing, but we can do that if we want
--modular building is probably the best way to test while going
--using canvas vs svg? - NOT GRADING ON UI so we can scale back on how much time is spent on UI
^could also try d3, canvas is probably more straight-forward
--concurrency protocol --protocol of locks & push to all users, could have timeout to stop a lock from lasting forever, but dont get too hung up on UI things
--focus on RESTful API, nice routes (UI-wise go for the simplest thing that works)
--play&share are “action”s better concept-collaborator
--want concepts to be novel and encompasses a core idea we had to create

concepts:
definitely note and sheet
concepts to fulfil the purpose of collaboration:
collaborator is reasonable--- purpose and operational principle are different
locking -- purpose: , op principle:when a user clicks on a locked note, other users can’t click or drag on it

Decisions Made/Alterations to Project Plan:
Decided on using deltas for real-time updating
UI and smaller features don’t matter as much as making an overall well-designed app and API
Concepts to be updated in revised design doc-purpose and operational principle should not be the same