Meeting Notes

- Updated data model: resolved confusion over parent match relation
- For each parent match, would want a ? at each end
- Metadata discussion: generalize data other than winner or loser in order to accommodate different types of sports
    - How is the metadata associated with the different users or teams within a match?
- Winners for matches, brackets, and tournaments
    - Is it redundant?
    - In elimination, could just pull the bracket winner from the final match
    - Probably necessary in other tournament types, like Round Robin, because we need to aggregate to find the winner every time
- Only have matches between teams
    - Do we enforce that each match must have teams of equal size? E.g. team of 1 vs a team of 3
    - Interesting solution to this problem would make app more viable, since it would be tedious to have the user enforce it
- For API, don't skim on error model
    - make sure all database responses are caught with appropriate status codes
    - well-defined errors + metadata
- For now, focus on making the app robust!
- UI for tournament display
    - graphics library like fabrics.js or tables
    - D3 - has good charts
- Testing recommendations
    - QUnit is good, forces you to specify expected correct and incorrect outputs
    - make calls asynchronous
- Security scrubbing inputs
    - server should take care of it
    - arrays - look into serialization library
    - e.g. array of winner, loser and parse out desired inputs
    - sidenote: why have array of [winner, loser]? should make separate relations
    - shouldn't need to scrub the payload, because you are creating it
    - have to defend on the server side (e.g. with curl), must use rep invariants
    - security is not a priority for MVP