- Emailed us design feedback
- Likes email over SMS decision
- Cost splitting
    - Write up algorithm for cost splitting
    - Tests
    - Worth a section in design document
        - Adds a layer of complexity and something interesting to the app
    - Data model should incorporate monetary contributions
- React bugs
    - Do not set props values in state
    - Callback
- MVP demo
    - Moment library for date parsing
    - Use state less, use props more
    - Core information in one state and passed down through props
    - Styling
        - Bootstrap grid
    - When is a party closed and costs calculated?
    - What if someone doesn't bring a full item?
    - Minus points if looks like Fritter in the end
- Need emails and cost splitting first

Summary of discussions and advice

1. Clearly lay out cost splitting functionality and algorithm in design document since that is a complex and interesting component of the application
2. Use state in parent and props to pass through children; use state as little as possible
3. Think about design misfits
    a. When is a party closed and costs calculated?
    b. What if someone doesn't bring a full item?
4. Needs a lot of styling; shouldn't look like Fritter UI in the end
5. Think about what key components still need to be developed and do those first
    a. Meet to plan out rest of time (9 days)
    b. Email notifications and cost splitting are core components still missing

Summary of new decisions

1. Using email notifications instead of SMS
2. Creating a document to organize features remaining and who will be working on which features
3. Using a backlog to organize all tasks

Changes to problem analysis or design

1. Detailing cost splitting in design document
2. Data model should have cost as a property of item

3. Host determines close of party and when costs are settled after party occurs