Potluck Design Doc

Group: ARMR

Authors: Anna Frederich, Rachel Rotteveel, Maddie Severance, Ryan Stuntz

Overview

Brief description of system to be built

We are building a party planning application to better organize and centralize the important aspects of a party, namely the location, host, guests, and supplies. Furthermore, we want to ensure that everyone contributes equally to the party, whether that be in terms of supplies or money. This application will notify users of the time and location of the party as well as their individual contributions in order to prevent users from forgetting the party or their assigned supplies.

Key purposes

The main purposes of our app are three-fold. First, we want to make this app an easy way to invite friends and family to attend an event and notify/ remind them of its happening. In other words, the host will input the emails or numbers of all of the guests and then they will be sent an email or text notifying them of their invitation. Second, we want to evenly distribute the bringing of supplies to the event. The host will make a list of all supplies required for the party, and then invited users can logon and volunteer to bring an item. Finally, we want to ensure that the cost of the event is split evenly as well, so if the host buys all of the supplies for the event, they can easily keep track of what they have bought in addition to how much each item costs, and then after the party, each attendee will be notified of how much money they owe the guest.

Deficiencies in Existing Solutions

Some existing solutions for our problem include creating Facebook events or inviting people in a traditional group chat. Facebook events allow you to invite guests, write a description of the event, and post in the event. However, there is no way for people to coordinate who is bringing what or do equal cost-sharing. Group chats are also inefficient, in that all of the coordination happens in a chain of texts, and you have to read and write dozens of texts to make plans. Tilt is an existing crowdsourcing app for events, but it doesn't help people decide who is bringing what, and doesn't divide costs equally. Our app is different than existing solutions because it aims to achieve all three of our purposes in an integrated way.

Design Essence

Concepts

Party:
- *Purpose*: To create the event you will be hosting
- *Operational Principle*: If you click 'Create Party', then you will become the host of the event and be prompted to invite guests who can join your event and to identify supplies essential for your event.

Invite:
- *Purpose*: To notify a person that they have been invited to an event.
- *Operational Principle*: If a host adds a person to their invite list, then the person will be notified via text or email that they are invited to a party which they can then view, RSVP for and sign up to bring items to via a link included in the text/email. The host and guests can see who has accepted their invites to a party.
- *Anticipated Misfits*: Making sure that the invites account for everyone you want at the party.

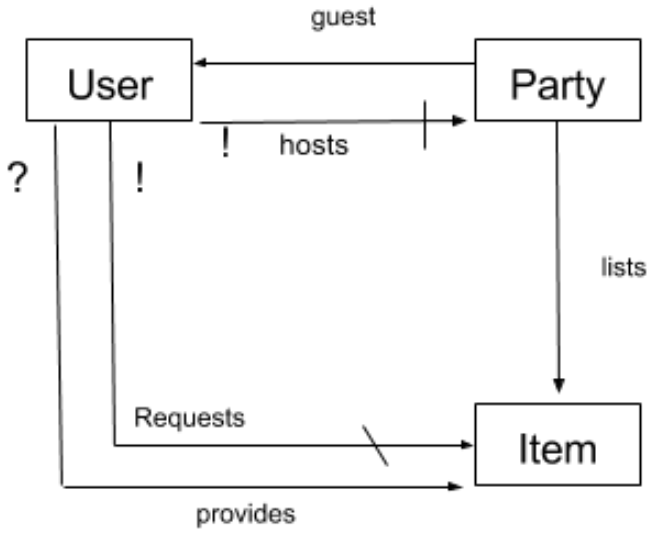
Contribution:
- *Purpose*: To assign how guests will equitably provide supplies or money for a party
- *Operational Principle*: When a guest responds to a party's invite, he or she is either held responsible to a monetary contribution to the host if they host is providing all of the supplies, or he or she must make a contribution to the party by bringing one of the items on the supply list.
- *Anticipated Misfits:* Determining if supplies people are bringing are evenly distributed. Determining whether contributions are defined solely by the host, or whether a guest can choose his or her contribution.

Item:
- *Purpose:* To represent a need of the party in order for the party to be well supplied.
- *Operational Principle:* If an item is added to the supply list, then it will be provided by either the attendee or the host.

Supply List:
- *Purpose*: To keep track of what items are being brought by whom, and what items still need to be signed up for
- *Operational Principle*: The central part of the app will have a supply list that is populated when the host identifies supplies needed for the party and when people sign up for a contribution or add a new contribution to the list. The supply list shows each item for the party and identifies either the attendee bringing that item or that the item still needs a provider.

- *Anticipated Misfits:* Items still being forgotten when creating/adding items to the supply list

Data Model



At the heart of our design, our data model is very simple. All that is included are the sets for user, party and item. The beauty of this is that with the simple data model, more time will be spent on developing the user interface and making design decisions that affect how users will use our app.

Security concerns

In all web applications, there are many points of entry for attackers. In our application, we want to try and limit the number of different attacks that are possible in order to best prevent the attacks. In the attacks that we will discuss, we are assuming that attackers do not have access to the server. The attacks that the user may be able to make would include adding items that may be dangerous to either our database or that would attempt to run code on our server. In order to mitigate these attacks, every call to our database will just pass in parameters instead of being created on the fly. In addition, we will not use eval() so that no arbitrary code will be able to be run on the server, and will make sure that no other parameters will be used in such a way so that they can execute arbitrary code passed in.

Another thing that could go wrong is users getting access to parties that they should not have access to. In order to mitigate that, we will be making sure

that every user is logged in before they ever see a party. By doing so, we can make sure that users never see a party that they should not be allowed to see.

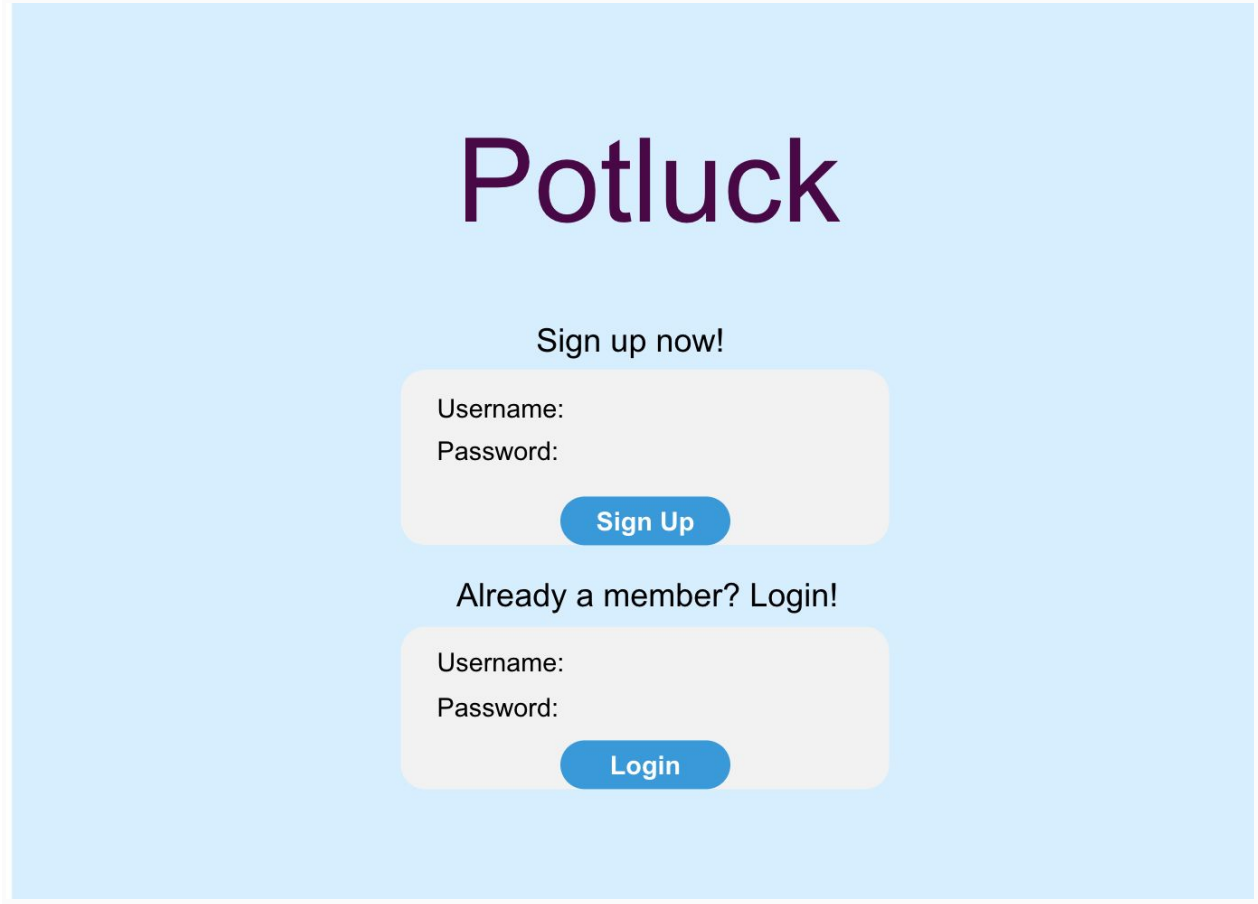
In order to mitigate cross site scripting and request forgery, we will institute a few different requirements. In order to prevent cross site scripting, we will make sure that no input from the user will be used in a context where the html will not be escaped. The way that this will be done is through React. React automatically escapes everything unless you use a "`dangerouslySetInnerHTML`" method which we will not be using in our code.

In addition, to prevent CSRF attacks, we will make sure that we do not allow requests coming from other domains. This will be done through Node by using the csurf package which will allow us to ensure that every request is actually coming from our website. This package will take care of most of the details and implementing it as middleware will allow us to be safe on every page.

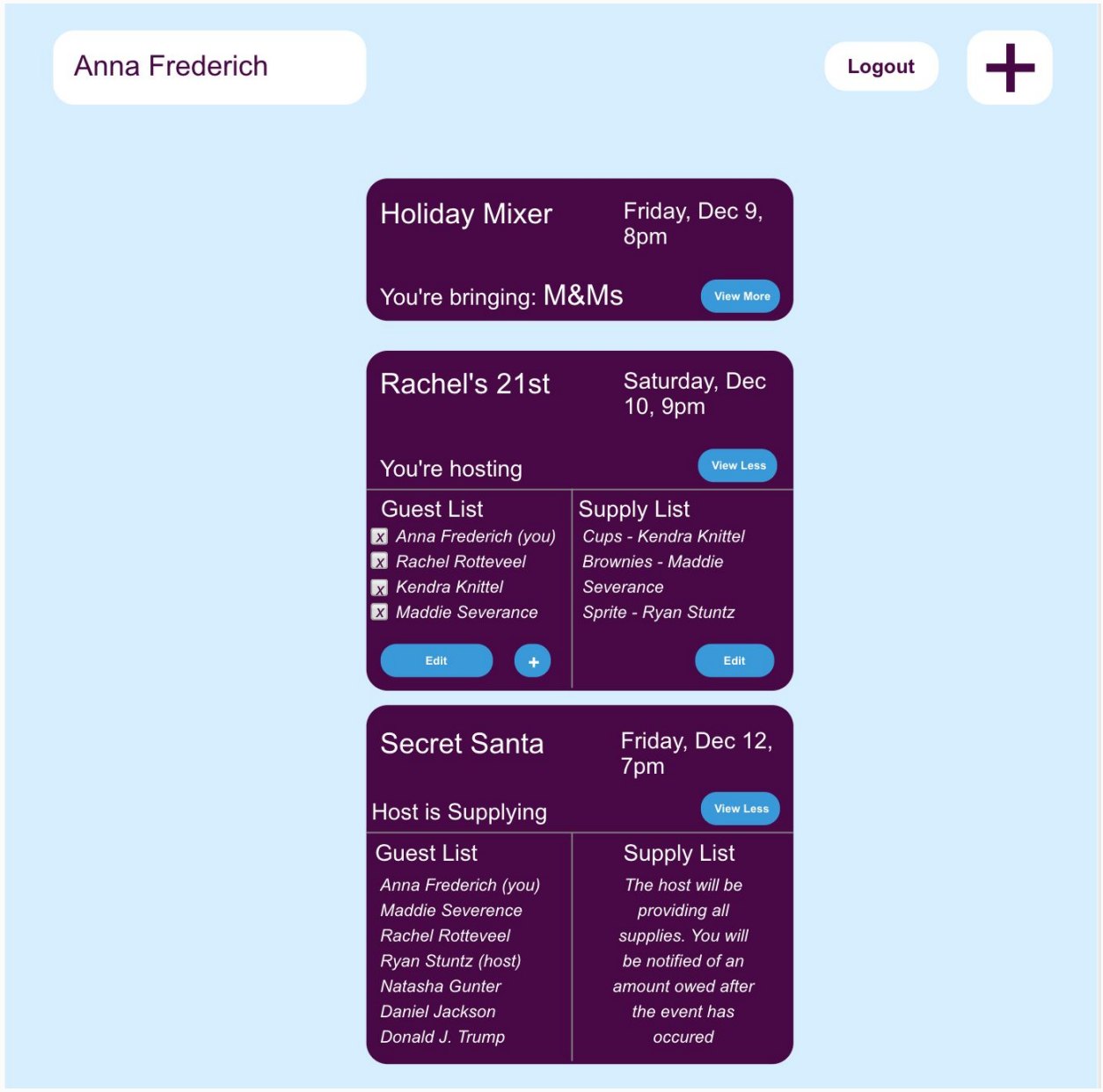
User Interface

Wireframes

Login Page:



User home page:



Create new party:

Challenges

Design Choices

The first design challenge that we came across was based on the method of authenticating users. Specifically, we discussed how we wanted to implement the signup process for new users. When a guest gets invited to a party via text message, should that link go straight to the party page or not? Both sides had valid

arguments, but we decided to redirect to a login/signup page instead. The main reason behind this was security concerns as making sure that each person should have access to the party that they are looking at is paramount for confidentiality. By making each user have to have an account before looking at a party, we make sure that only the users who should be able to see a party can. However, the idea behind going straight to the party had some arguments that were intriguing. For one, by doing so, less users would be lost in-between the text getting to them and actually signing up for something to bring to the party. In order to speed line this process, we will be populating the sign-up field with the information that the host has already given us about the user (name, email/phone number) so that it will be very quick to sign up.

Another challenge is deciding how the supply list is created, and how that factors into cost sharing. Originally, we decided that the host could either create and buy all of the items on the supply list, and each person could reimburse him/her equally, or each guest could bring one item to the party that was either suggested by the host or by a guest. Ideally, we would like to allow both the host to create an initial supply list, and allow guests to contribute new items to the list if they see a need. However, if the host brings multiple items from their initial list, we would also have to incorporate cost-sharing into that party, even if each guest is bringing an item as well (so the host doesn't find buying a majority of the supplies to be a burden). Therefore, we can't have black and white "cost-sharing" or "supply-sharing" party types. A party can become a mixture of the two, which might be a challenge in our design.

Design Risks

Anticipated Misfits:

1. Making sure that the invite accounts for everyone you want at the party.

Response: We could allow the host to invite additional guests later, not just in the initialization of the party. If we have enough time to implement, we could allow guests to recommend other people they want to see at the party to the host. The host could approve additional guests at their discretion.

2. Determining if supplies people are bringing are evenly distributed. Determining whether contributions are defined solely by the host, or whether a guest can choose his or her contribution.

Response: We would allow both parties to define contributions, so guests feel like they have the same say in what items are at the party as the host. For most parties, we anticipate that supplies being bought would most likely be evenly distributed among users. We would also allow a scenario where the host

suggests and buys a majority or all of the items, but would allow the host to choose whether to use cost reimbursement for this scenario.

3. Items still being forgotten when creating/adding items to the supply list

Response: We could in the future include a "suggested item" concept where when the supply list is initially created, items would be suggested to be added to the list based on other items added already, number of people being invited, and the type of party.