SwipeInvite

Sprint 1 Planning Document

Team 16

Andrew Davis

Kyle Krynski

Paul Ryan

Tejaswi Namuduru

Zening Chen

Overview

The main objective of this sprint is to build up all three major areas of the project into a usable, but limited form for the user. These three major areas are 1) the user interface, 2) the client side background operations, and 3) the server side background operations. By building these three areas up in tandem, all areas of the project will be able to keep up and react in real time to other systems, essentially evolving together. In order to keep a limited scope, this sprint will primarily focus on allowing users to create and manage groups, as well as their own personal profile.

The SCRUM master for this sprint is **Paul Ryan**. Meeting will be conducted on **Tuesdays and Thursdays at 4:30pm**.

The risks associated with sprint are primarily associated with building a robust enough base to handle many of the later complicated interactions that will occur. For instance, the server will only be managing a small network of group and user objects in this sprint, but in the future it will need to handle a huge amount of events, groups, users, and the links between them. Thus, it is important for us to design the backend for a server to handle these future interactions, rather than being short sighted and only optimizing the server for this sprint. On the client side, we will need to take care not to overload the user's view with excess information, as this early stage should not need to display that much. If we overdo the views, then we will have trouble making the product look nice in later sprints when we try to pack in more features to the interface. Basically, the main risks for this sprint revolve around managing the small chunk of implementation efficiently, while keeping in mind the bulk of features that have yet to come.

Details

User Story 1: As a user, I would like to be able to manage my personal account and information within the app.

- Dev Questions: What will the app look like to a new user? What login information will be required? How will they log out and log back in?
    - Task: Design the views for a first time user login. Design the views for a returning user login. Design a control panel to managed the overall views of the app, including login/logout choices. (Zening Chen, 3hrs)
- Dev Questions: How will the controller handle the login process? How will the controller handle the workflow of creating a new user profile in the data? How will the controller handle a logout process? How will the controller gain access to an existing user profile on the server?
    - Task: Design the controller methods to handle a first time user setup. Design the controller methods to handle the views that need to get pushed to the screen. Design the controller methods to create a new user profile in the data model. Design the controller methods to access and write user profile information to the server database. (Zening Chen, 2hrs)
- Dev Questions: How will the data model on the client represent a user profile? What methods and variables need to be part of this representation?
    - Task: Design the class associated with a user profile. Design the class internals to handle groups, events, and acquaintances for later use. (Zening Chen, 2hrs)

- Dev Questions: How will the server handle a new user request? How will the server store a new user's information in the database? How will the server handle an existing user pulldown request?
    - Task: Design the server to handle messages from a brand new user. Design the server to store new user information. Design the server to handle profile pulldown requests from the client. (Zening Cheng, 10hr)

User Story 2: As a user, I would like to be able to create a group.

- Dev Questions: What will the views look like for creating a new group? What options will be visible?
    - Task: Design the interface for creating a group from scratch. Design the input views associated with the data for a group class. (Zening Chen, 3hr)
- Dev Questions: How will the client handle the view pushes to the screen as the user enters information? How will the client create the appropriate data model from the user's inputs?
    - Task: Design the client methods to ask the user the necessary information to create a group. Design the client methods to manage a new instance of a group class in the data model. (Zening Chen, 2hr)
- Dev Questions: How will the model represent a group?
    - Task: Design the group class to handle the members who are a part of it, the events that are part of it, how public or private it is, and who has the power to modify it. (Zening Chen, 2hr)

- Dev Questions: How will the server represent a group in the database? How will the server notify users of a new group that they are invited two? How will the server link users to groups?
    - Task: Design the server database model for groups, and their association to users. Design the server message handlers for group notifications. (Tejaswi Namuduru, 5hr)

User Story 3: As a user, I would like to be able to add specific acquaintances or people to the group I am creating.

- Dev Questions: What will the views look like for searching person or group?
    - Task: Design the interface for inviting from scratch or listing people you have been in groups with before. (Tejaswi Namuduru, 3hr)
- Dev Questions: How will the controller manage sending the information to the server, and then receiving the result?
    - Task: Design the client side message handler for adding people to the server instance of the group. Design the controller method for searching the existing acquaintance database. (Kyle Krynski, 10hr)
- Dev Questions: How will group class be able to hold and manage the list of members of a group?
    - Task: Create a list of members of a group in the class. Create methods to access and modify this list. (Tejaswi Namuduru, 2hr)
- Dev Questions: How will the server handle searching users and adding them to groups?
    - Task: Design a search function on the server based on client messages for parameters. (Paul Ryan, 10hr)

User Story 4: As a user, I would like to be able to select and search which groups I would like to become a part of in the future.

- Dev Question: How will the server differentiate between groups that are public and groups that are private?
    - Task: Design the server to handle flags for public and private groups as search parameters. (Tejaswi Namuduru, 3hrs)
- Dev Question: What will the search page look like?
    - Task: Design the UI layout as it is shown in the scratch (Paul Ryan, 5hrs)
- Dev Question: How will the controller handle the search messages between itself and the server?
    - Task: Design the controller methods to handle search requests for groups. (Tejaswi Namuduru, 2hrs)

User Story 5: As a organization leader, I would like to be able to find all people interested in my group.

- Dev Question: How the view be displayed for an organization leader looking at interested people in the group?
    - Task: Design an invitation view to manage the incoming interest requests for a group leader? (Paul Ryan, 5hr)
- Dev Question: How will the controller know if the user is the administrator of a group? How will the controller edit the group interests locally and on the server?
    - Task: Design the controller methods to communicate with the server on group interests and update model and view objects correctly. (Paul Ryan, 5hr)

- Dev Question: How will the data model know if a user is the administrator? How will it store the list of interested people?
    - Task: Design the data class for a list of invitations. (Andrew Davis, 7hr)
- Dev Question: How will the server keep track of the interests and leaders?
    - Task: Design the group database objects to handle leader notifications and interested factions. (Tejaswi Namuduru, 10hr)

User Story 6: As a group creator, I would like to be able to control who can modify the events and the members of my group.

- Dev Question: Will the details about access and permission be saved for each user or for each group?
    - Task: Design the data model for a group in a way to minimize the ability to tamper with local permission data. (Kyle Krynski, 5hr)
- Dev Question: How will a group control screen look?
    - Task: Design the view for how a group administrator can manage the group. (Andrew Davis, 7hr)
- Dev Question: How will the controller push group edits to the server?
    - Task: Design the controller message handlers to make the necessary changes to the local and server instances of the group. (Andrew Davis, 10hr)
- Dev Question: How will the server handle group edits?
    - Task: Design the server to manage the group database dynamically, allowing edits to be made on objects. (Kyle Krynski, 9hr)

Backlog

- Functional Requirements

1. As a team leader, I would like to schedule time that every member in the team is available for meeting by checking and comparing their schedules.
2. As a professor, I would like to find the best office hours that work for most students and notify all students in a manner such that they will actually address it.
3. As a busy person, I would like to be notified / keep track of near-future events simply.
4. As a member of a friend group, I would like to schedule a group study session with my other friends and select the location.
5. As a person whose 21st birthday is in a week, I would like to invite everybody I know and know how many people are going to show up so I know how much cash to bring.
6. As an introvert, I would like to reject an invitation to a party or outing in a non-confrontational manner.
7. ~~As a new college student, I want to get updated organization call out information I am interested in.~~
8. As a college student living off campus, I would like to let all my friends in the area know that there is free food somewhere and for a how much time.
9. As a business owner, I would like to be able to send promotional event information to possible local customers.
10. As a working mother, I would like to send my calender to my paid babysitter to let her know when she has to take care of my child.
11. ~~As an organization member, I would like to be able to send organization activity information to students who are interested.~~

12. As a professor, I would like to set exam and homework reminders to students in a colloquial manner.
13. As a student, I would like to be able to sync all my courses directly into my calendar by simply connect it to my mypurdue account. (if time allows)
14. As a user, I would like to be able to see and resolve conflicts in events that I choose to accept.
15. As a non-profit Community manager, I would like to be able to see charity participants availability for simplified coordination.
16. As a group leader, I would like to be able to schedule recurring events.
17. As an event creator, I would like to be able to choose who sees how many people have accepted the event.
18. As a non-smartphone user, I would like to be able to make decisions on events by text and have my electronic calendars get updated. (if time allows)
19. As a group leader, I would like to save and load templates for events to send to a group. (if time allows)
20. ~~As a user, I would like to be able to change my account settings (password, email settings)~~
21. As a user I would like to be able to invite non-app users via email or text message. (if time allows)
22. ~~As a user, I would like to be able to add and remove myself from groups specific to university clubs, organizations, project groups, etc. (if time allows)~~
23. As a Google Calendar user, I would like to be able to add events to my existing calendar.
24. ~~As a user, I would like to be able to manage my personal account and information within the app.~~

~~25. As a user, I would like to be able to create a group.~~

~~26. As a organization leader, I would like to be able to find all people interested in my group.~~

27. ~~As a group creator, I would like to be able to control who can modify the events and the members of my group~~.

- Non-Functional Requirements

1. Storage space: many of our users will probably be considered power-users, who want our app to be as space-efficient as possible. We can achieve this by following the thin-client model for our application. We will also save space on the server by deleting the info of events that have passed.

2. Push notifications: We envision our users only using our application in two situations: creating an event and responding to an invitation, so we will have to notify users right away when they receive an invitation.

3. Swipe to accept: most users prefer convenience over detail and we hope that this feature will be of high utility. A user will be able to receive invites and swipe to accept or decline an invitation without unlocking their phones. They will be able to view the details of the invitation (host, time, location) in the notification and decide if they want to accept or decline the invitation.

4. Security: User person information, passwords, and schedules will need to be hidden from public view.

1. Scalability: The main part of the system that will need to be scaled is the server, which will essentially become an algorithmic database challenge.

2. Platform: This application should run on most modern Android smartphones, running Android 4.2 and above.

3. Reliability: It is important that a user receives every event that they are invited to. We don't want our users to miss out.

4. UI: The UI needs to be simple, beautiful, and easy to use. Users won't want to use an app that looks bad or is not intuitive.

5. Performance: The client's response time is more critical than server response time. We don't want any freezing or crashing.