Final Project Design

Team Name

Team5

Team Members and Email Addresses

Dylan Klohr | dylan.cs.ku@gmail.com
Gabe Magnuson | g400m001@ku.edu
Philip Wood | p3wood719@gmail.com
John Ward | johnw.mkv@gmail.com
Joe Werle | werlej@hotmail.com
Alan Wang | ultimate801@gmail.com

Meeting Times

2/15 @ 12:00pm
3/8 @ 12:00pm
4/5 @ 12:00pm

Lab Meeting Time

1:00pm Monday

Group Contact

Dylan Klohr

Project Sponsor

None

Project Description:

For our capstone project, we have decided to develop a location-based mobile platform. It's our hope, that said platform can be used to automate existing proximity-based actions and workflows (e.g., class attendance, information distribution, etc.). For example, a professor teaching a class in a large lecture hall such as Budig 120 who wishes to conduct daily-attendance currently faces several challenges without a modern technological platform to assist them. Currently, methods commonly leveraged for attendance in such a setting include all students signing in on a communal sign-in sheet, requiring either the professor or a TA to actually record said sign-ins manually. In addition to the time required to both pass the sign-in sheet and record its contents later, the additional challenge of authentication comes into play. In the current schema for classroom-attendance, there is very-little that prevents one student from signing both himself or herself in as well as other students not actually present in the class. This project is motivated by the desire to optimize common actions that have existed for decades and bring them into the 21st century by taking advantage of modern

advancements and insights in the realm of technology. Our platform would combat the aforementioned problems in that it would allow a customizable system for individuals to connect with others near-by and collect desired information from them, as well as provide natural authentication of who is actually "checking in." Upon the completion of our capstone project, we aim to provide a mobile utility platform aimed at providing new functionalities to a given population in a close-proximity to one another.

Project Milestones[1]

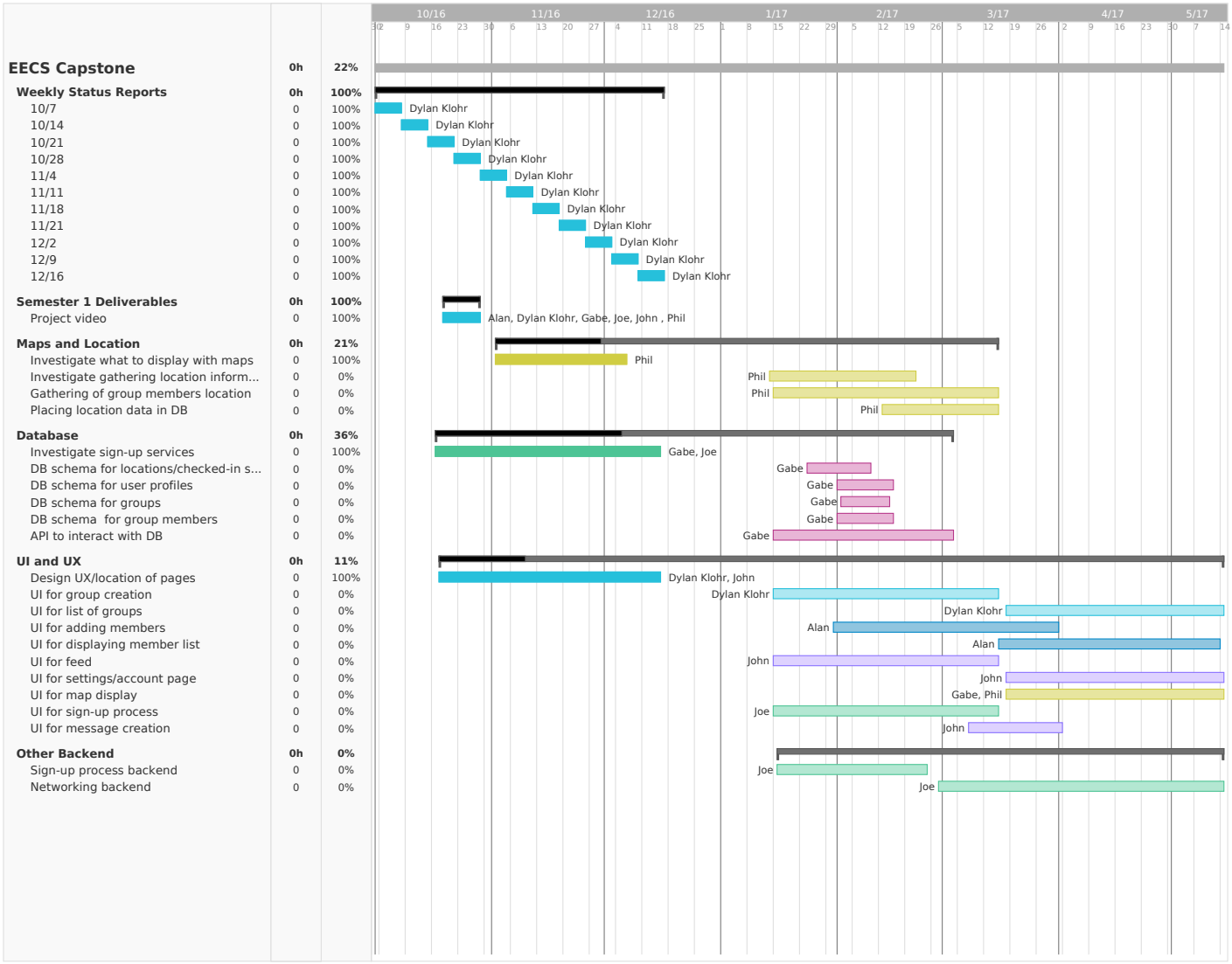
First Semester:

- Finalize project scope and desired features (Wednesday, October 12)
- Project Proposal Video Completion (Monday, October 24)
- Project Structural Design Completed (End of semester)
- Research:
    - Helpful third-party utilities identified
- Documentation
    - Class Diagrams
    - Domain Model
    - Use-Case Diagram

Second Semester:

- Database Implementation / Database API (Friday, February 15)
- Ability to create an account as a user (Saturday, March 4th)
- UI Functionality for sign in / sign out (Tuesday, March 14th)
- Ability to create a group and register additional individual accounts (Friday March, 31)
- Admin has the ability to send a message to the group (Saturday, April 1st)
- Admin has the ability prompt a check in with group members (Monday, April 10th)
- Usability and Unit testing completed (Friday, May 5)

Gantt Chart[2]



Project Budget

No Budget for this project

Work plan (current)

Dylan Klohr:

- UI / UX design and implementation
- Front-end

Gabe Magnuson:

- Database Implementation

Philip Wood:

- Map Overlays/Displays and location pinging

John Ward:

- User Authentication
- UI Implementation
- Front-end

Joe Werle:

- Admin Capabilities

Alan Wang:

- Group Creation/Settings
- Front-end

Github link

- https://github.com/dylanCSKU/eecs_capstone.git
    - Contact Dylan Klohr for access to private repo

Final Project Design

To avoid the archaic time consuming check-in and group management methods commonly used today, we've developed an application which streamlines the process, whilst providing additional organization and security. To provide our users with a custom and safe experience, all individuals who utilize our app must register directly with our application or using a common social media account. Rather than managing separate social media authentication responses ourselves, we are leveraging Firebase Cloud Message, which will authenticate users using any federated identity providers, and return a single user authentication token, known as a JSON Web Token (JWT), which we send to our database using an HTTPS request from the client. Once a user has attempted a sign-in we verify the user authentication token in our backend, to ensure it is up-to-date and corresponds to an existing user in our system. Our backend is currently hosted on the provided KU MySQL database.

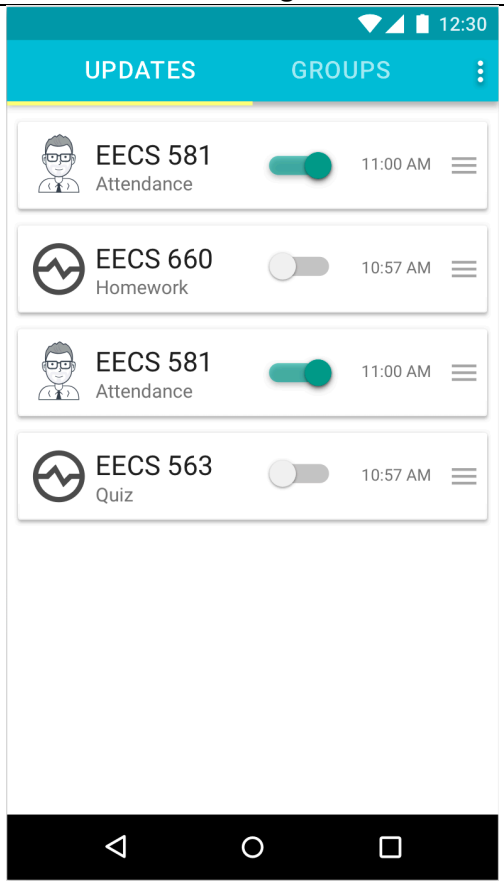
Following backend authentication, users will be directed to their dashboard which allows for immediate access to recent post and check-ins for groups they are associated with. Users can instantly check-in for active role calls that are visible on the dashboard, or may acquire additional information by selecting a desired notification. Selecting a check-in notification will open an overlay which will provide a description set by a group administrator, a pinned location on a map, and a check-in button – if the user is within the specified range. We feel that providing a map location allows for clear proximity visualization, allowing users to more easily recognize how close they are to any given check-in location. Mapping is done using the Google Maps API which is a part of Google Play Services. This comes with added pictorial flexibility and innate Google Maps features, such as pinning locations, adding overlays to maps, and providing navigation. Additionally, we seek to provide group administrators with an added sense security allowing them to check their members' location, given that they are within the specified radius of the check-in point, and that the member has provided permission for their location to be visible. We feel this addresses use cases such as class fieldtrips, where groups may be on the move or in large spaces, where line of sight may be obscured to group members.

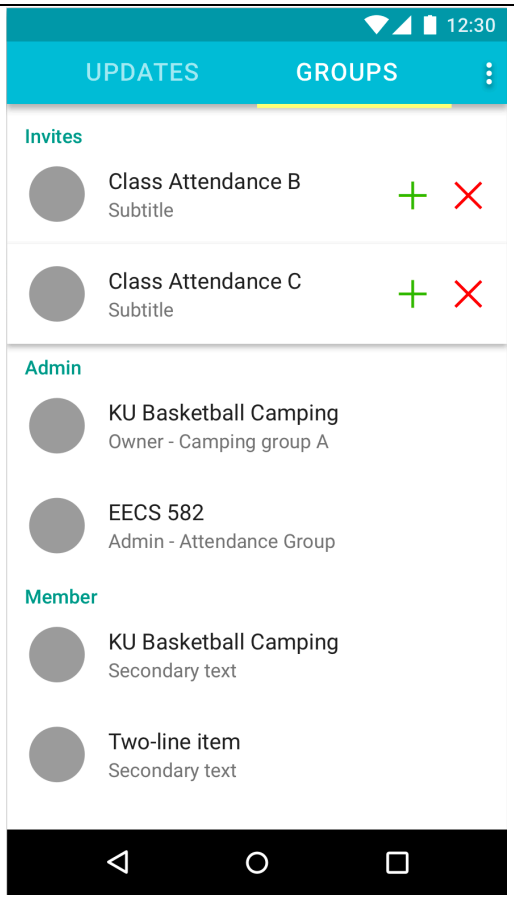
A key feature that our app introduces to the market is the idea of a mobile check-in. Many existing location based pinning or check-in services fail to address cases like the one above. By utilizing the Google Location Services API we provide administrators the option to bind the check-in location directly to their device, in addition to providing static location based check-in. While creating a check-in event admins will also be given the ability to specify a check-in radius as well as an event description, which are both visible to the member, as mentioned previously. Upon finalizing the event the client app will send the event information upstream, via an HTTPS request, to a PHP script running on our backend server. All appropriate changes will be made to database, at which point a push notification will be initiated to notify all group members of the check-in event. Push notifications will be completed using a call to Firebase's REST API from the server, providing the event description, radius, check-in type, check-in location, and all Instance IDs for group members. An Instance ID is simply an identifier used by Firebase to differentiate each individual instance of our app. The Instance ID is sent and verified when the user signs into the app, at the same time that we send the JSON Web Token. To clarify, the JWT corresponds to an individual user and is used to verify that specific user whenever and wherever they login, regardless of device. The Instance ID on the other hand specifically correlates to an instance of our app running on a single device. Once a push notification has been received by the client, the appropriate aforementioned check-in activity will be opened. Administrators are then able to view who has and has not checked in, as well as a running total of check ins for the group.

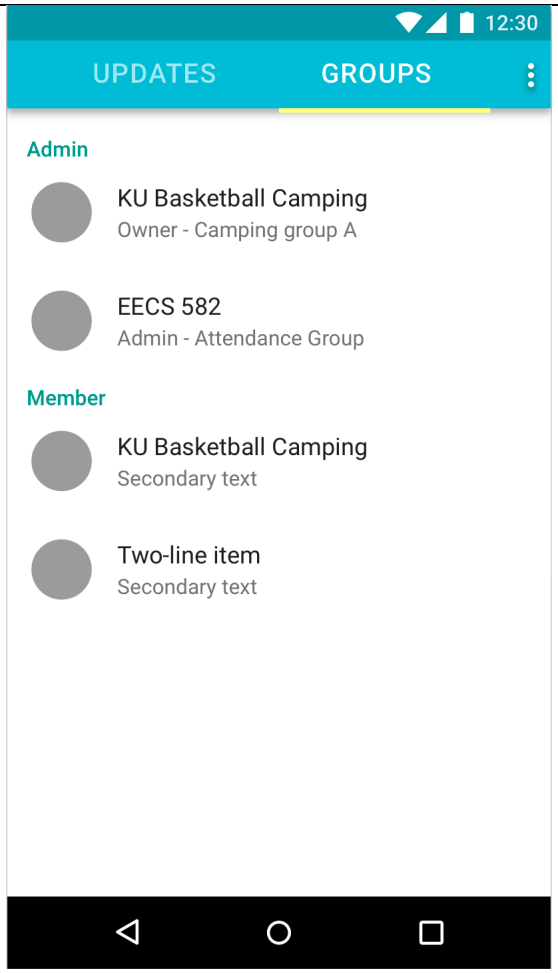
For easy group management we have provided a 'Groups' page which is accessible form the main dashboard. The groups page contains all groups that a particular user is involved in, differentiating them by admin/member status. Selecting a group will provide the user with a list of events, similar to the main dashboard, however only containing events from the selected group. Additionally, a settings menu will be provided allowing a user to exit the group, modify

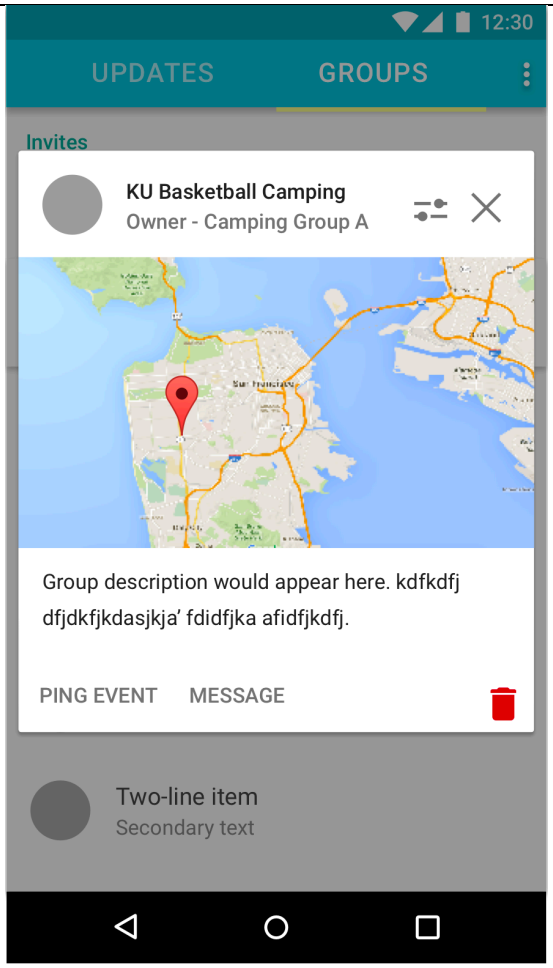
their visibility settings, and initiate an event, if they are an admin. We hope to provide users with an extra level of customization and security if they wish for their location to never be shared to the group administrators. Furthermore, location information is only shared to group administrators given that the member has checked in to the check-in event in question and is within the specified radius. These circumstances ensure that location information is only sent under conditions in which the user would want their location information visible. In reference to the app as a whole, when it comes to accessing private information, mainly user location, permission checks will be completed at runtime, as is standard in Android Application Development. Requests cannot be completed unless the user is currently providing the necessary permissions, which they can modify at any point within Android Application Settings.

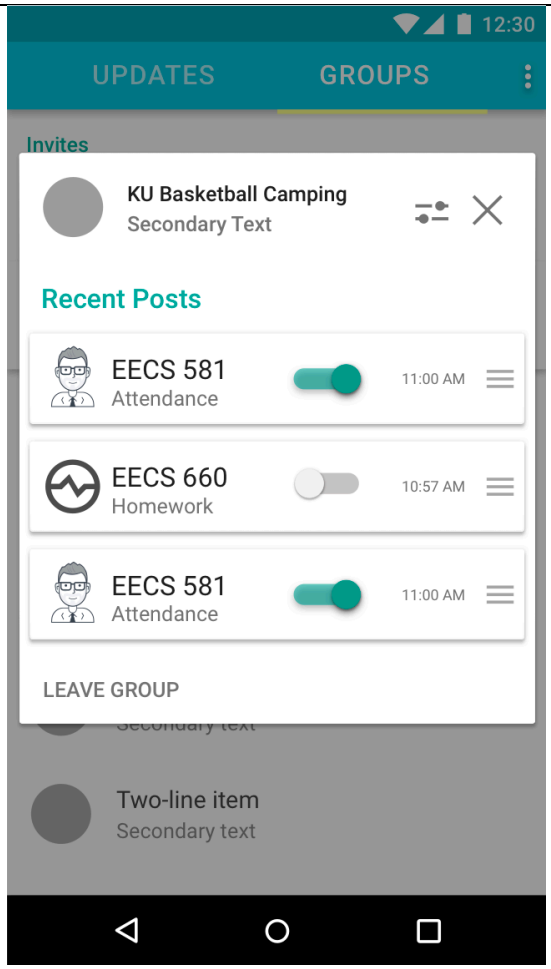
User Interface Design[3]

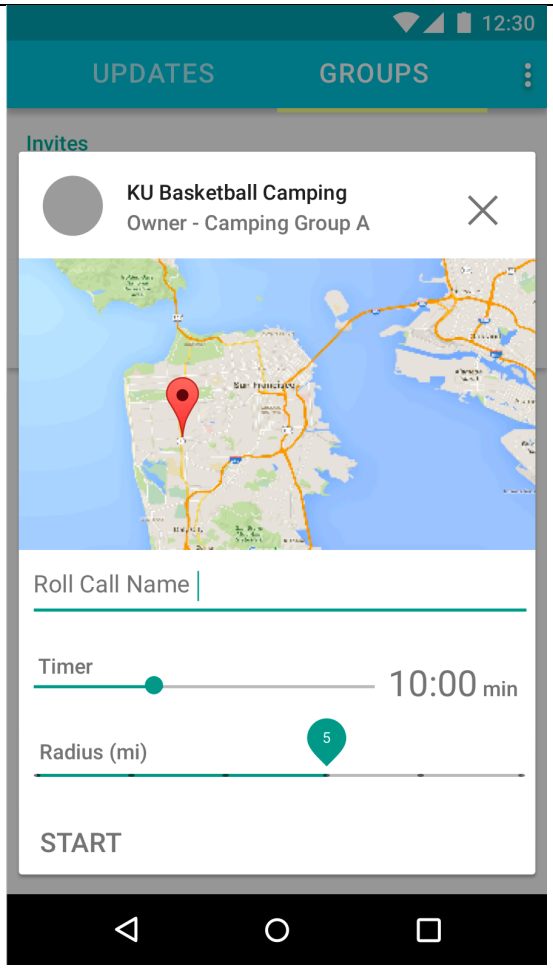
Title	Design
Updates Tab	

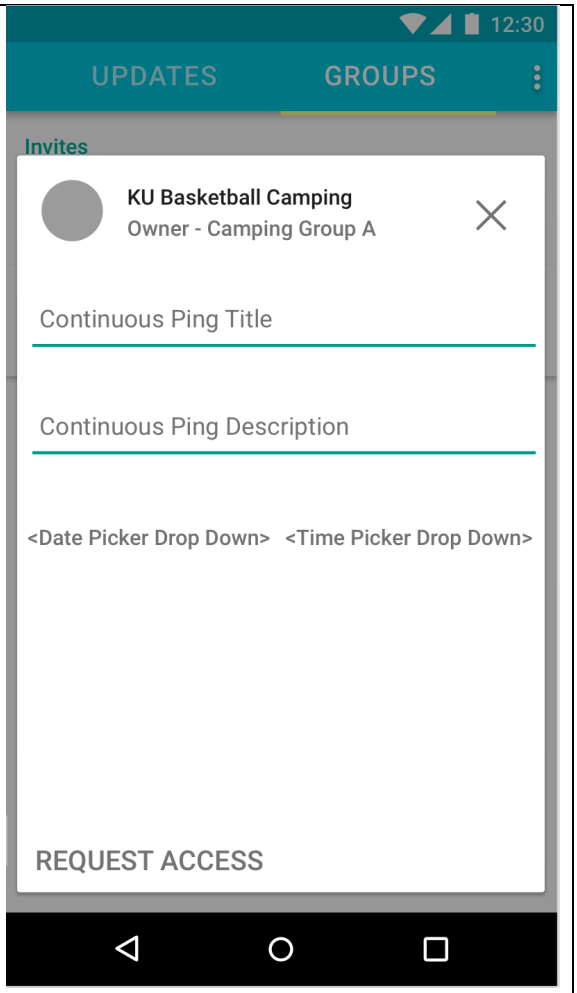
Groups Tab w/ Invites	

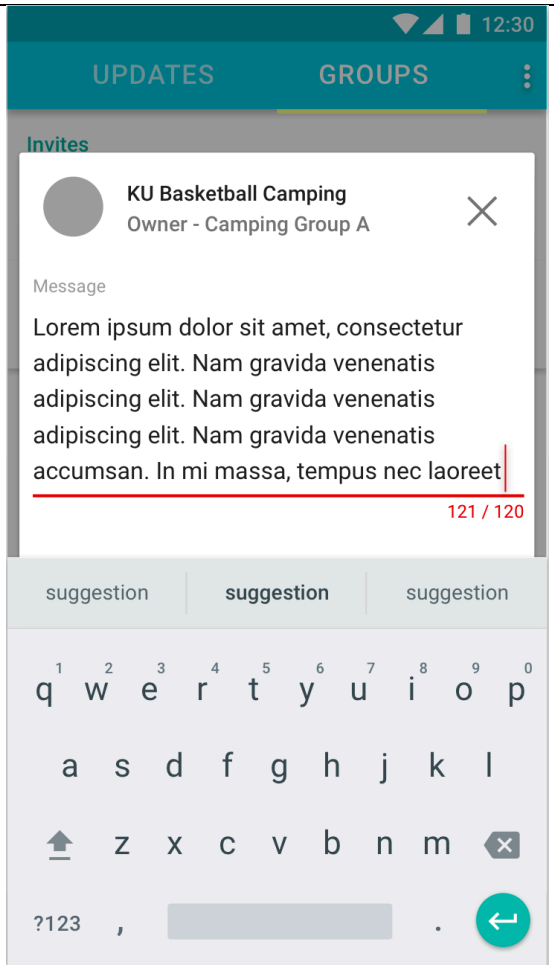
Groups Tab w/out Invites	

Admin Group View	

Member Group View	

Single Ping Event	

Continuous Ping Event	

Admin Group Message	

Intellectual Property and Ethical Issues

Intellectual property issues began to crop up more and more as we delved deeper into our project. The nature of our project being a location based Android application has us borrowing a lot of work from others in the form of API's and open source projects. It will be important as more third-party-leveraging functionality is added that we follow any requirements that come along with their licensing. Additionally, though not strictly a legal issue, it is important to us that our application does not come off as a copy of other applications that use Google location and map services such as Google Latitude. This will include paying particular attention to similar or competing applications to make sure this project is offering something distinct.

Because a lot of the work done by our app will be done on the back end, the question of boundaries that we may overstep ethically while making this app comes into play. A large ethical issue that we face is privacy as it relates to location based services.

Google already provides a location service that periodically calculates a devices location, and Android (6+) requires any app that tries to use this service to get permissions from the user before hand (at installation), however, our application will be saving and sharing locations between phones and the ability to let others see your location is a central moral concern. Throughout development we plan to answer this by implementing a design that balances simplicity while ensuring the user always has control over what is happening with his or her location. We will likely have to implement settings that the user can control exactly how freely we will be using their location with regards to how visible they appear to other users of the app. Due to our back end including both location and account information, it is additionally important that we not only make sure the user has control of their privacy but also that we securely store this sensitive information in a manner that is less-susceptible to malicious attack.

Change Log

Note: All changes to this contents of this document after we submitted our initial project proposal last semester are denoted with a superscript at the end of the header with content that has been changed.

[1] *Milestones*: We have made further refinements to specific project-deliverables as a result of work completed on the project over break, as well as additional project-scope refinement.

[2] *Gantt Chart:* Deadline and specific deliverables have been changed as a result of further project-scope refinement.

[3] *UI Design:* Detailed outline of idealized User Interface completed.