COS 333 Spring 2016 Design Document

Project/Team Name: Holic
Designated Project Leader: Andrew Tao (at9@princeton.edu)
Group Members:
Belinda Ji (bji@princeton.edu)
Jenny Peng (jqpeng@princeton.edu)
Emily Zhang (emilyz@princeton.edu)
Brandon Zhou (brandonz@princeton.edu)

1 Overview

Holic is an app that allows users to track their own and their friends' drinking experiences to make them more aware of their limits and stay safe. In particular, users are able to enter data about their drinking on a night out by pressing a button for every drink they take. Users are also able to enter a chat room with their friends, where each member's drinking history for the night are displayed and members may chat with each other. The creator of the Room, the Admin, is the main point of contact for the other members of the Room: the Admin can update the other members' drinks, and is automatically one of the people notified when someone's BAC is reported to be high.

Tracking their own drinking helps users be safer, and Holic will notify a designated friend with a warning and the user's location if the user's calculated BAC rises above a safe level or if the user has returned to a designated Home location, determined by checking the user's GPS location. Anyone looking to keep track of their drinking or any group of people looking to drink safely together may use this app for that purpose.

Users may also enter empirical judgments of a past drinking night to help Holic update their happy place and help them get there safely in the future. Holic will recalibrate based on user feedback to send warnings earlier or later. Holic also sends monthly reports of user drinking and their reported feedback.

We will develop Holic using the Android for the front and middleware and a VPS to host a web server to handle the Chat Room.

2 Requirements and Target Audiences

The target audience is young adults who use alcohol, or anyone who wants to increase their awareness of their alcohol use, may want to keep track of their intake and let their friends know how much they have had, or may want to keep track of how much their friends have had. There is no easy way to do this, but we provide a platform designed for this task, which is a more accurate way of keeping track of drinks compared to people just having to mentally keep track of their drinks, as people commonly have to do when they go out.

Users must be running Android (4.1). Holic lets them easily keep track of their statistics and view past history. The system thus also provides a platform for people to track their drinking patterns and habits over time, ideally leading them towards a healthier drinking lifestyle.

3 Functionality

Usage Scenario 1

Alice is hosting a Lawn Parties pregame for some of her friends. She creates a Room and adds his friends to it, becoming the Admin of the Room. Her friends use the Chat Room to coordinate when and where the pregame is. As an Admin, Alice can update the drink count of each of the friends within the chat. Her friends also have the opportunity to update their drink counts for the situations in which Alice is not keeping a strict eye on them. Uh-oh, looks like Bob has had a little too much. Thankfully, he is within Alice's Room. As an Admin, Alice is notified when Bob's BAC becomes a little high.

Because Bob goes hard early on, he is tired by noon and wants to go home. Unfortunately, the main act has not yet performed, and Alice is a huge fan of Nate Ruess. She has to keep track of all the other members of the Room anyway. Bob has had a lot of water at this point, and seems to be sobering up, so Alice lets him walk home alone. Because Bob has preset a Home location, the app sends Alice a text notification when Bob reaches his dorm room.

Usage Scenario 2

Bob, the novice drinker, wants to go out on a Saturday night. He downloads Holic, designates Alice as his point of contact (Alice accepts the invitation via text), and doesn't set his home location because he is paranoid. Over the course of the night, every time Bob pulls out his phone it reminds him to log his drinks, and he does. Later that night Alice receives a text that Bob's BAC is starting to get high, and the text contains Bob's location. Alice finds Bob, checks on him, and walks him home. Thanks to Holic, Bob wasn't left by himself when he had a little too much.

Usage Scenario 3

Carol is a lightweight. She downloads Holic and enters her height, weight, and gender. The first time she uses the app, she has one drink. Despite Holic not displaying any warnings, she feels dizzy and disoriented. The next morning, Holic asks her how her night was. She reports her buzz, and the next time she drinks she has twice as much as last time. Now, Holic knows this is already a lot for Helena, and it texts her friend Alice to check on her. Holic responds to her feedback and recalibrates for her. At the end of the month, the app also creates a statistics page that summarizes the average number of drinks she's had per session, as well as her average feelings after each one.

4 Design

Holic uses the standard three tier system with the frontend, middleware, and most of the backend residing on the Android application stack (client-side app). The frontend serves as the screens/activities of the application (Android XML). The middleware (Java/Android API) is the application logic for handling persistent data, the algorithm to calculate BAC, calculations for the user's history/statistics. The backend is is responsible for saving/serving persistent data on the client (Java/Android APK) which will be the user's profile and saving/serving chatroom data on the remote server.

Frontend:
On first startup the user interface consists of a form to save the user's information/stats (more in the middleware). The startup activity/screen introduces the application with text explaining the application as well as possible tutorial images for use cases. Once the user has entered his/her information the data is passed to the middleware.

For all subsequent startups the application opens into the home screen.The home screen consists of a list of options available (design TBD). Options include: setting a new session, accessing Rooms and Chats, alcohol history/statistics, and settings.

When a user wants to start a new session he/she will click the option from the home menu. The user interface for active sessions is relatively simple, containing buttons for adding drinks (e.g. +1, +0.5, +0.25, etc) as well as number to indicate the user's BAC level. The rooms may also be accessed from the active session screen. When the application is loaded into the foreground or started again and a session is active, the application opens into the session screen.

The chat feature contains three additional screens/activities: the interface for all Chats you are in, the interface for a Chat Room, and the interface to create a new Room. The first interface lists all

Chats as well as a button to create a new Room. Each Chat in this interface will contain some identifying info or preview info for that Chat Room. When a user presses the button to create a new Chat Room, the next screen/activity is invoked. This new screen is the interface for creating a new Chat Room; here the user adds other users to the Chat Room and submits the information to create the Room. The third screen is the interface for a Chat Room which has two parts: normal Chat functions and a display for the status of all users in the room. The normal Chat functions include a scroll for all messages and a text box for new messages. The other display includes information on the status of everyone in the room laid out in a grid pattern.

Middleware (Business Logic):
The middleware connects the user interface with the backend, separating what is stored on the client's phone (history and statistics, chat logs) and what is sent to the server (chat logs, current BAC level to refresh the user's status on their friends' phones). It also contains the algorithm to calculate BAC and statistics for the user's history, which are aggregated monthly.

On the user's phone, the middleware takes user input (drink counts and times, height, weight, physical gender) and calculates their BAC (using public formulas as an estimate). This calculation also takes into account past user feedback on their personal tolerance. The progression of number of drinks and BAC over a night is saved in the user's history on the client side, and the current BAC is periodically sent to the server to refresh the user's BAC status on their friends' phones. The middleware is also responsible for passing data to the backend (both the client's persistent data and sending requests to the remote server).

Backend:
The backend of this project will involve the implementation of the Chat Room in our application and saving client-side data. For the remote server, we will use a virtual private server (VPS) to manage a Linux machine. The machine will host a web server where the people using the app will be connected to. Each chat room will be associated with a unique chat ID. There will be client side code that sends messages as well as additional info including: chat id, username, etc. to the web server. The server side code opens a port/binds to a socket and receives the message and stores the message in the server via HTTP requests; serving messages are done in a similar fashion. On the client side, the app can then check the server every couple of seconds to see if there are new messages. In addition, we can implement push notifications on Android using Android's cloud messaging API.

Reach Goals for the Project:
There are two reach goals for this project that we will work on if time permits.

- The first is to implement administrators in the "group chat" part of the project. The admin will be able to control the amount of drinks of any person in the chat. The admin will also

be able to get the location of any member of the chat if the member of the chat enables that feature.

- The second reach goal is implementing automatic texting with GPS location incorporated, which happens in two cases. When a person has arrived safely home after the night is over, we plan to implement this using the GPS coordinates and setting a threshold distance from the entered home location. Also, when the app decides to automatically notify the emergency contact that the user is too drunk, we plan to allow the user's GPS location to be sent as well, so that the emergency contact can safely locate their friend.

5 Timeline

March 19 - Set up server, project repository, create basic Android applications. Standardize elevator pitch and finalize application name.
March 26 - Implement saving/serving user information, adding drinks and BAC calculations (starting a session). (Prototype to show TAs)
April 2 - Implement automatic texts and notifications if BAC reached certain levels.Write server side code for chat room. Obtain Google Play app license in preparation for live code deployment.
April 9 - Implement screens that relate to the chatroom which includes all functionality except saving/serving data to and from the server
April 16 - Connect middleware and backend. Get the chat working. (Alpha)
April 23 - Fix all errors from alpha and implement one of the reach goals (admins in chatroom) (Beta)
April 30 - Fix all bugs and clean up code (Ready the Demo)
May 10 - Dean's Date (Tech Demo)

6 Risks and Outcomes

- None of our group members has much experience with app development, so we don't have as much familiarity with developing frontend, middleware, and backend components and having them function smoothly as a single unit. In particular, none of us have ever programmed for Android devices before, so even using figuring out the Android IDE will not necessarily be a quick process.
    - We will be taking steps to be as familiar as possible by the time spring break is over.

- The question of whether we need a developer license, and whether we can get it from Princeton. Obtaining a Google Play developer license seems to be a matter of requesting one, but perhaps there are other licenses or similar roadblocks we may run into.
- There are potentially too many functionalities that are difficult (having an Admin for the chat rooms and Home location etc.) and may be underutilized that we are hoping to implement. However, we are also prioritizing broad picture functions over the small details (e.g. chat rooms),
- The implementation of the chat room could be tricky depending on the design decisions we make. If you want to add your friends into a group, is it better to identify users by their phone numbers, or allow people to "friend request" each other's profiles? If some people in the group leave early, should they automatically leave the group or continue to receive notifications? Depending on how we choose to implement specific features, we could hit different obstacles: if people switch phones, want to turn off notifications, etc. Depending on what inconveniences arise, we may choose to switch to a different implementation entirely, which would be particularly risky if our current implementation is already very well incorporated into the other components.
- Ideally the goal of the app is to accurately track one's drinking patterns and determine when someone has had too much, which in turn means that our frontend and middleware need to facilitate accurate calculations. In particular, there are problems that might arise with how the user inputs their amount of drinks. Our idea is to make the interface as simple as possible, so that even people under the influence of alcohol can continue to record their drink input--however, this raises the issue that people who are too drunk may just repeatedly record drinks even though they aren't drinking any. Although this may not reflect a technical obstacle in our app development, it does impede the app's overall functionality, and is still a consideration we should be keeping in mind.