# Alarm$^2$

## Design Document

Team 20

Kalpan Jasani, Ashwin Chidanand, John Redmon, Scott Walters

February 02, 2018

## Problem Statement

The modern alarm clock is usually the built in alarm app of a person’s smartphone. However, there are limitations to this that can be improved. Our project tackles the issue by focusing on smarter ways to wake a person up. These include setting alarms based on specific locations for travelers, notifying people if your alarm is not manually turned off, and set alarms for others in groups. In short, it is a smart alarm!

We considered how limited many stock alarm apps on the market are. Many alarms are simply clones of others with some added sleep tracking statistics or require challenges to be completed to turn off an alarm, rather than just flipping a switch. While these are good, we felt that we could innovate in a different direction altogether, of this simple day to day application.

Alarm$^2$ has a focus not only on innovation, but well implemented innovation. We believe this alarm is the first of their kind, and can serve as future inspiration for others who wish to push the boundaries on seemingly mundane apps we take for granted.

## Design Outline

Our application has the following important features:

- Call/Text a friend if you fail to manually turn your alarm off.
- Be woken up at a specific location, and further have the ability to adjust the radius when you will be woken up.
- Be a part of groups where the group admin can create alarms for everybody.
- Learn about yourself and the different alarm features upon seeing important and relevant statistics over the past year. These include your success rate of waking up for time based alarms, of the feature of the location based alarm, etc.

## Architecture and Design

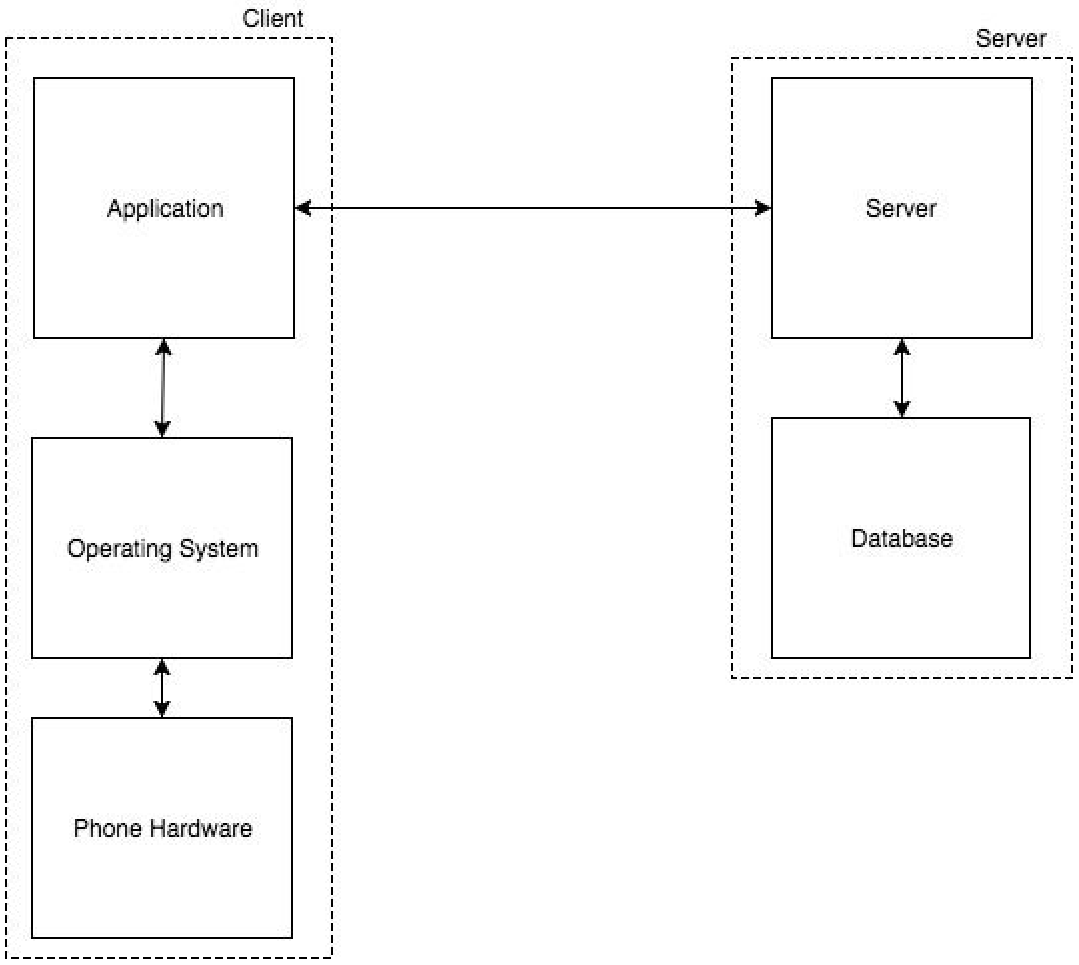


Diagram 1: Diagram explaining the architectural outline for our system

Our application needs to collaborate with the Android system on the phone regularly:

1. We plan to have the application place a call and send a text message, if the user doesn't wake up to stop the alarm.
2. Furthermore, we will require a background service that runs all the time. We, importantly, need to monitor the alarms' times and locations in this service, and also communicate with a server where our group features will be updated from.

3. We will also require to store data on the local storage: SD card, or phone’s internal storage. This data includes voice/text messages, and alarm settings.

We will also need to communicate with a server to:

1. Store long term statistics data
2. Store all user account information.
3. Store groups’ information

## Design Issues

Non Functional:

1. Which database to use in the back end server?
    a. SQL
    **b. Elastic Search**

    Decision: We will use Elasticsearch for our data storage on the server. We do not plan to use our server side database heavily, as most of data will be stored on the user's device. Elasticsearch is easy to integrate in our system and it does not take much experience in creating queries relevant to our project.

2. Which type of framework to use for the back end server?
    a. **Python Django**
    b. Node.JS

    Decision: Node JS would be great if we we were heavily dependent on the server for our application, as it can handle a large amount of simultaneous connections with clients and would be great for scalability. As the majority of our app's functionality takes place on the user's client and the dependency on the server is small, we decided to go with Python, as our back end developers have more experience with it and Elasticsearch is easy to connect with using Python

3. Where should the server be located?
    **a. AWS**
    b. Use our own machine

    Decision: We will use AWS to host our back-end server. The main reason for using our own machine would be that it is free to do that. However, having AWS service has advantages:

    1. We can obtain free usage in AWS for this initial phase.
    2. Secondly, if we have a large increase in users, AWS will allow us to scale up.

4. How to develop the client?
    a. **Android Studio IDE with provided framework for Android development**
    b. Other

    Decision: We use the standard, or Android Studio IDE. It is sufficient for most purposes. We don't need extra speed and other different features, and hence we will adopt the popular and fast way of using Java and Android Studio

5. How should the syncing of clients be handled?
    a. Have clients send refresh requests to the server and database
    b. **New data is pushed to the clients by the server.**

    Decision: Having client requests would lead to loss of battery and slightly more data usage for the individual client. Therefore,  we will use push notifications so a client will be notified of a change and updated as soon as the change is made to the database.

6. Which tool to use to call other people?
    a. Twilio Service
    b. **User's phone and phone number**

    Decision: We chose to use the user's phone over Twilio because it is free and although Twilio offers some useful features for larger scale products we do not think that we will require the promoted features enough to choose it over the built-in functionality.

7. How are we going to store the alarms that are not active?
    a. **Local storage**
    b. On Server

    Decision: We chose to store alarms locally since they will not take up a significant amount of memory and can be easily accessed.

8. Who chooses the time for an alarm to fail?
    a. Automatic time out
    b. **User sets a time period manually. If within this time, if alarm hasn't been stopped or snoozed, it has failed.**

Decision: We have thought of the option of having time of, for example, 2 minutes, or 5 minutes, after which it is considered that a user has failed to stop the alarm. But, users are different, and therefore, we keep it to the user to set the time, in minutes, as to when the alarm has failed. If the alarm has failed, a backup number would be notified, or the group admin would be notified.

9. How should the phone numbers and names attached to them be stored?
    a. Name:String, phone number:int
    b. Name:String, phone number:long
    **c. Name:String, phone number:String**

Decision: We will be storing both the Name and Phone Number as Strings. Many of the builtin API functions of both Google Maps and Android take the phone number parameter as a String variable. Additionally, if a number has leading zeros they would be omitted if stored in a numerical variable type. The Name and Phone Number will be stored as a pair where the user of the application sees the name and the backend sees the number.

Functional:

10. Should the user need to have a login?
    **a. One time verification code**
    b. Tie in with social media
    c. No login

Decision: We chose to have a one time verification code. We briefly considered using a social media login to track users but decided to use the built in phone number instead. By sending a verification code, we negate malicious users from coming up with any phone number to harass other users.

11. What should the maximum radius for the location feature be?
    a. x miles
    b. Percentage of travel time
    **c. No limit**

Decision: We will not put a limit on the maximum radius a user is allowed to select for location based alarms, as this would limit what the user may want to do. Instead we will allow them to select any radius, and if they are currently within the radius a pop up will notify them of this, and prompt them to change the distance.

12. How should notifications before assigning a contact for “notify a friend” be sent?
    a. E-mail
    **b. Text message**
    c. Push message to the app

    Decision: A text message will be sent to the receiver of the message. We decided an email would not ensure the recipient would be aware of the message when it happened. Due to the time based nature of this application, we valued the method that could send the message faster. We also considered sending push notifications to other users that had the app, but this would alienate non app users and require extra man hours spent coding and debugging where a text message can suffice.

13. As a user what is the first thing I should see when I open the app?
    **a. Alarms set**
    b. Make new alarm option

    Decision: We chose to allow users to see the current alarms they have set. We believe it is more important for the user to see what alarms they already have set rather than assuming they will create a new alarm every time they open the app

14. How should the user get to select options for the alarm?
    a. Drop down menu
    **b. Option page**
    c. Different for each alarm

    Decision: There will be an option page for each alarm. Upon touching an alarm, a menu of that alarm’s features set during its creation will appear and can be edited. We prefer the same procedure for each alarm to help streamline the user’s understanding of the app. The drop down menu was considered but we believe having a separate option page will be more elegant and not interfere with interactions of the alarms below it in the UI.

15. What should the options be for the location based alarm?
    a. **Radius**
    b. Street/Train stop
    c. **Coordinate**
    d. Combination

    Decision: There will be a combination of radius and coordinates used. The user will have the option to be woken up at precise coordinates of the location they input, or put an X mile radius around their location and have the alarm go off whenever the enter this radius. The street/train stop only accounts for a subset of users, so we decided to choose the options that would benefit the most people using this app.

16. What happens if the location features are not/stop working?
    a. **Pre assign ETA for alarm**
    b. Disallow location based alarms
    c. Wake up user if internet connection goes down

    Decision: Upon setting a location based alarm, the ETA will be shown. A standard alarm will be set in the background to go off five minutes before the ETA if internet connection to Google Maps is lost. The user will also have the ability to manually override the back up alarm time. We believe completely disabling  location based alarms in the case of lost internet is short sighted and waking up the user any time the connection goes out may prove frustrating if connection is frequently going in and out.

17. Does the user have to confirm to join a group?
    a. **Yes**
    b. No

    Decision: The recipient of a group alarm must verify that they want to partake in the group alarm. There are two reasons. The first and less important is that the user may want to wake up earlier than the group alarm time and set their own alarm. The second and more severe is to prevent someone from spamming a number with group requests, pulling pranks by remotely setting alarms for a number at 3 am., etc. The sender of the group alarm request will be notified if the recipient accepts the alarm or not, but cannot for them to accept. Finally, as an extension of this, there will be the option for a user to block group requests from a number if pestering grows to the level of harassment.

18. How does the user differentiate between what alarm is what type?
    - **a. List style**
    - b. Icon identification
    - c. Folder system

    Decision: We will use a list style that is organized chronologically. Additionally, users will have the option to view only the alarms that are in the category Standard, Location, or Group instead of viewing all alarms they have set. The icon identification is somewhat merged with this idea as the icons will be used to organize the items in the list. We agreed the folder system would be unnecessarily clunky

## Design Details

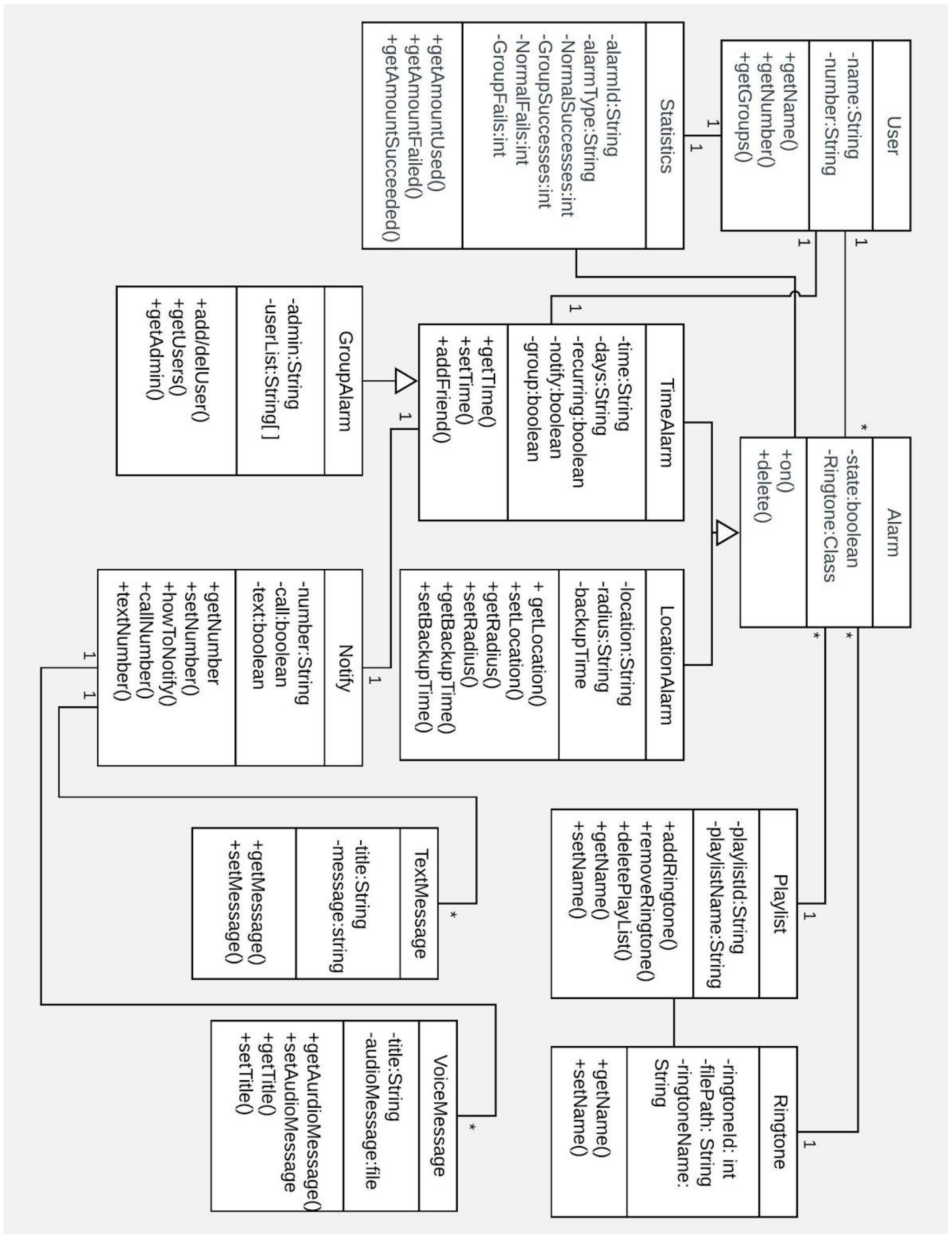


Diagram 2: Diagram showing the class diagram

## Description of Classes

- User
    - All users are an instance of this class
    - Will store name and phone number (unique ID)
    - Created upon account initialization and when there is a user set as a “notify a friend” feature
- Alarm - Superclass for all types of alarms
    - Stores boolean for its current state (On/Off)
    - Stores custom playlist or ringtone class
- Time Alarm - Subclass of Alarm
    - Stores time and days for the alarm
    - Allows for “notify a friend” and gives the option for group alarms
    - Created when user adds a new standard alarm
- Group Alarm - Subclass of TimeAlarm
    - Stores ID and name of admin and a list of all users part of the group
    - Created when user adds a new group alarm
- Location Alarm - Subclass of Alarm
    - Stores destination for alarm, radius for alarm to be triggered, and backup time incase location of the device becomes unavailable
- Statistics
    - Handles the various statistics
    - Accessed from options menu
    - Created when app is first opened
    - Updated every time an alarm goes off, recording its success and of the “notify a friend” feature’s success
- Notify
    - Handles the sending of text and voice messages to other phone
    - Does not handle editing messages
    - Created when a user sets the feature to “notify a friend”
- Text Message

  - An instance of a custom text message
  - Sends message to Notify class when requested
  - Created when user chooses to add text message in the option menu
- Voice Message
  - An instance of a custom voice message
  - Sends message to Notify class when requested
  - Created when user chooses to add voice message in the option menu
- Ringtone
  - An instance of a custom ringtone
  - Accessed by Alarm class if user wants custom ringtone
  - Created when user chooses to add a new ringtone
- Playlist
  - A collection of song ringtones
  - Accessed by Alarm class if user wants custom playlist as a ringtone
  - Created when user chooses to add a playlist

## Flowchart

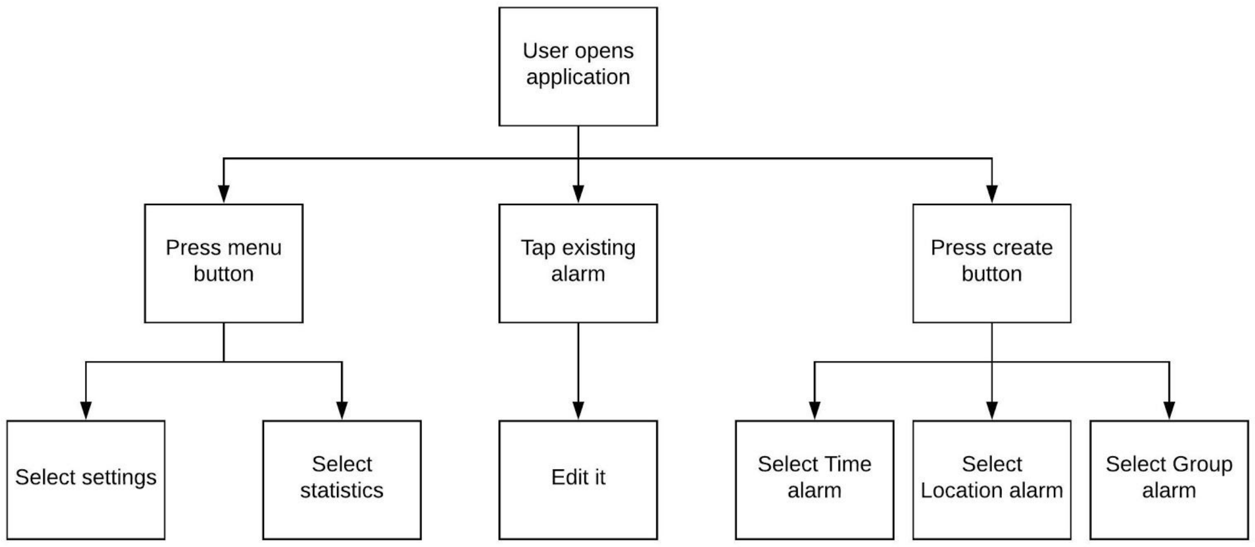


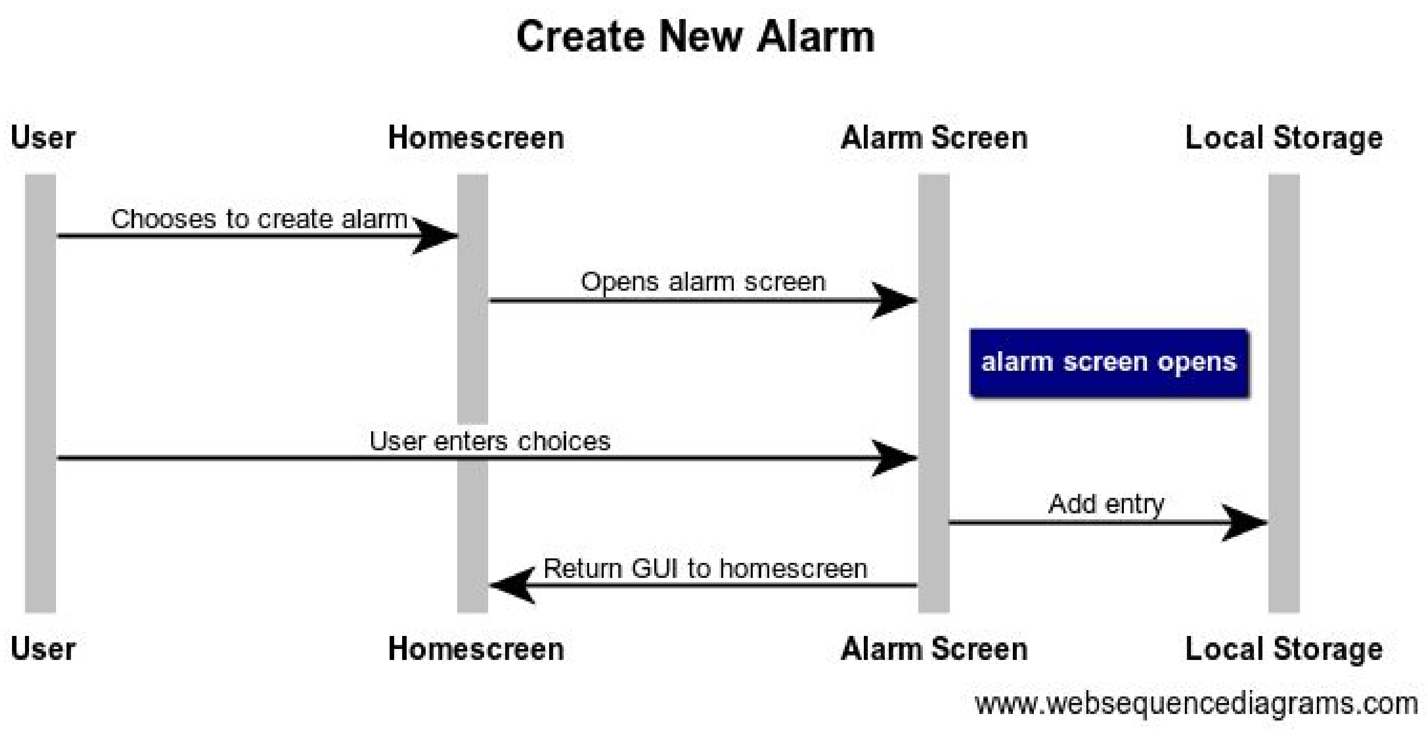
Diagram 3: Diagram explaining the user flow

When the user opens the application apart from the first time, he faces three choices, as shown. The menu button is on the top left side of the screen (as seen from the UI mockup).

- As part of settings, the user can:
  - Change his name
  - Set custom text/voice messages.
  - Change the default time when an alarm is considered not stopped manually
  - Set away mode for all groups at once
  - Cancel all alarms
  - Logout
- The statistics will give meaningful numbers in lists.
- When the user opens one alarm, he can change its settings, and also delete it
- When the user taps the create button, they will then be given a dialog box wherein they select the type of the alarm. For all alarms, the user would be able to set the ringtone.
  - For time alarm, a user can pick the call and/or text message option to notify a friend, and some more features.

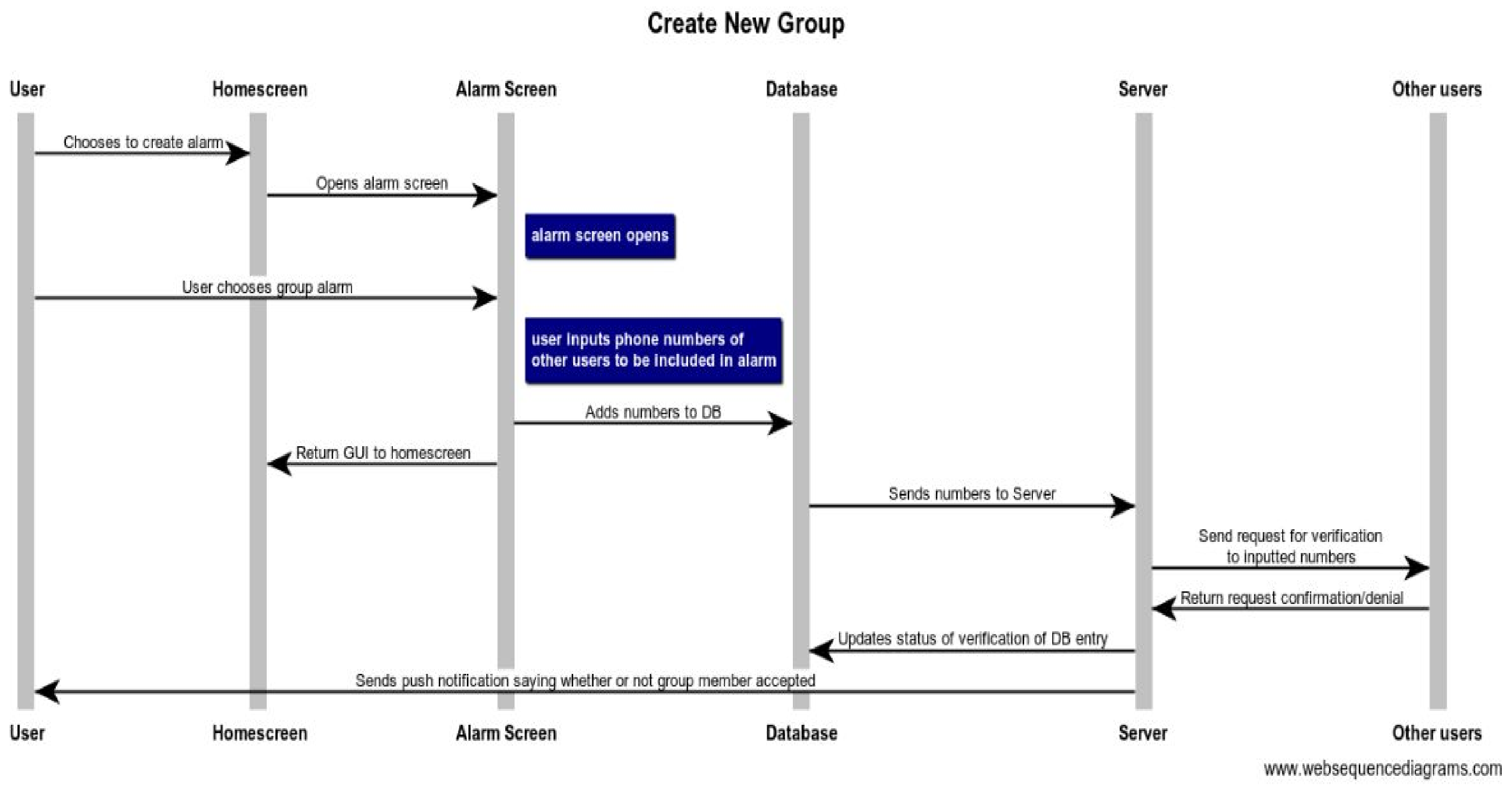
    - For the group alarm, the user will then proceed to name the group, and add other users.
        - The user can then create an alarm within the group. It would automatically notify the others on affirmation by the user
    - For the location alarm, the user can change:
        - The location
        - The radius
        - The backup ETA time
- After creation of any alarm, which would take an unknown number of steps, finally, the alarms would show up in the alarm lists, and the user would reach back the home page.

## Sequence Diagrams



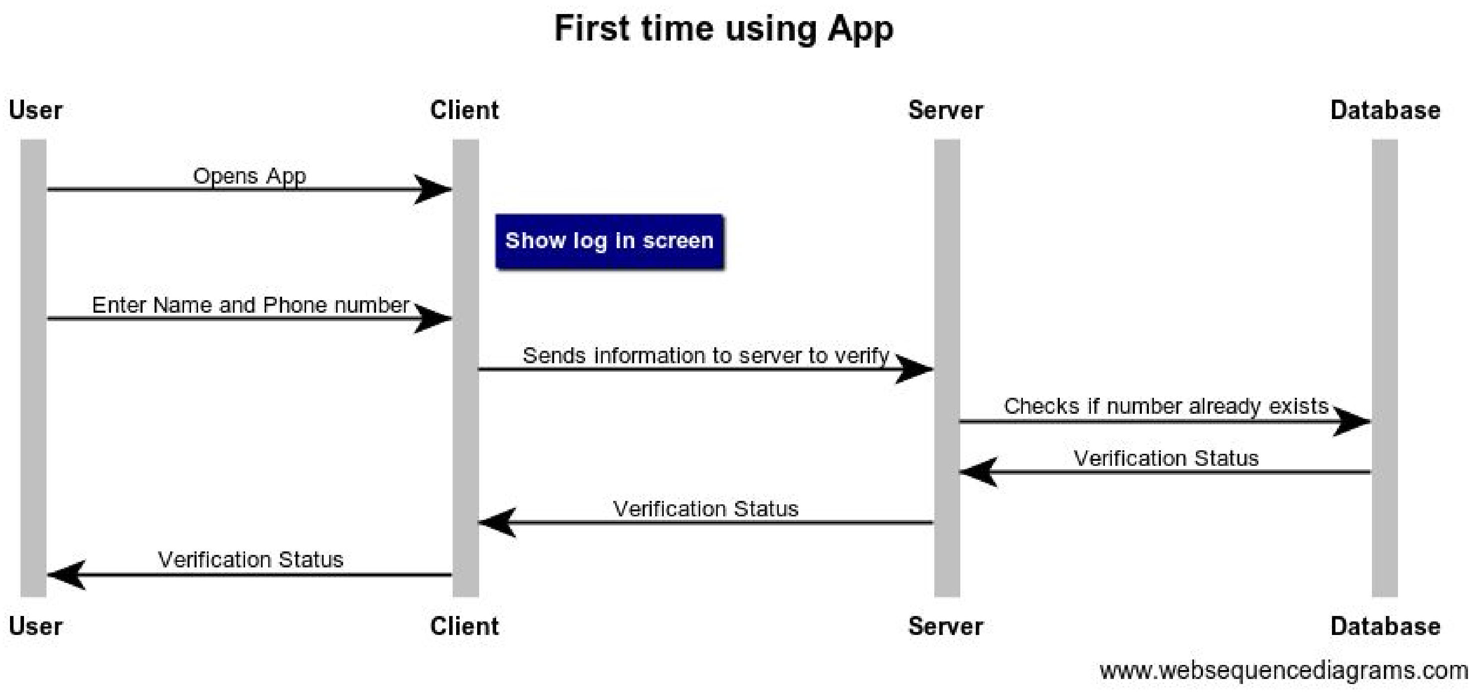
Creating a New Alarm

1. User will open the app and choose to create an alarm from the homescreen.
2. A ‘Create Alarm’ page will be opened and the user will select the options he wants, such as type of alarm, the time or location, which ringtone or playlist to use, and who to notify.
3. Once created, the alarm settings will be stored on local storage.
4. The GUI will be added to the user’s home screen.


Create New Group
User
Homescreen
Alarm Screen
Database
Server
Other users
Chooses to create alarm
Opens alarm screen
alarm screen opens
User chooses group alarm
user inputs phone numbers of other users to be included in alarm
Adds numbers to DB
Return GUI to homescreen
Sends numbers to Server
Send request for verification to inputted numbers
Return request confirmation/denial
Updates status of verification of DB entry
Sends push notification saying whether or not group member accepted
www.websequencediagrams.com

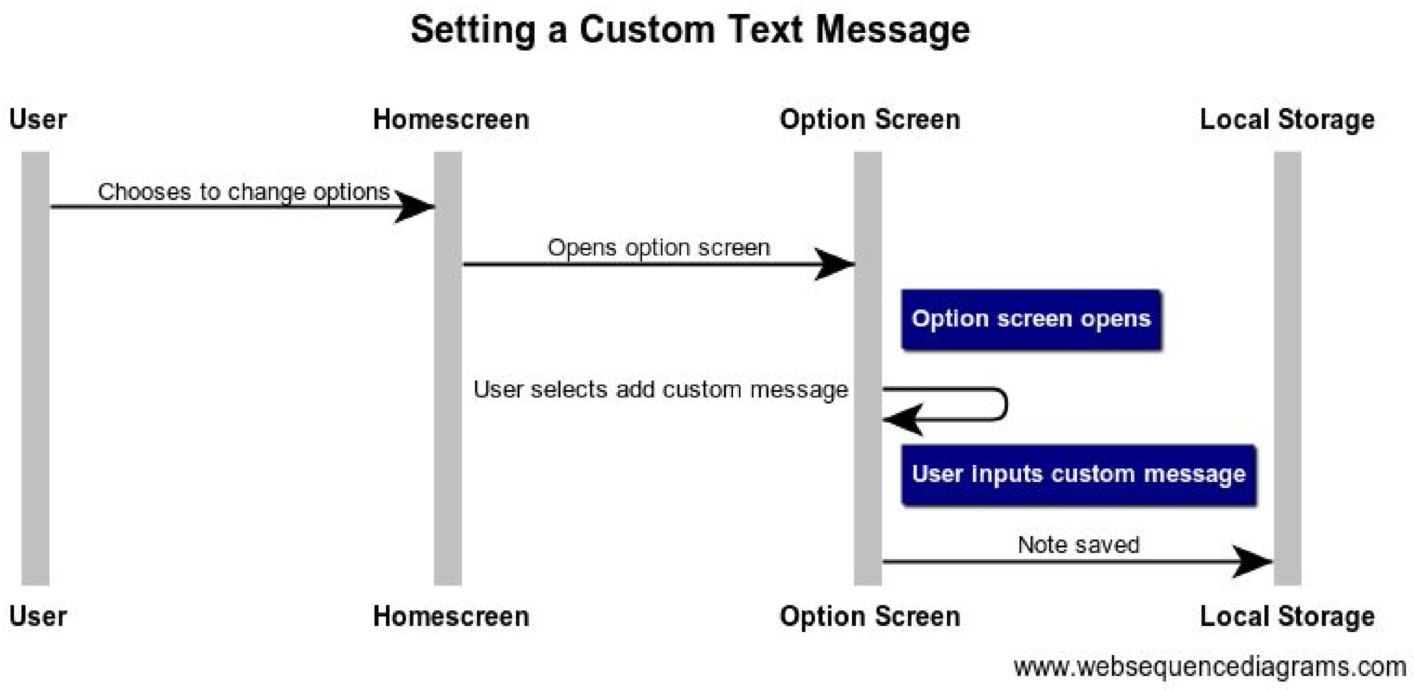
Create New Group

1. User will open ‘Create Alarm’ page from homescreen.
2. User selects group alarm and adds the phone numbers of other user’s they would like in the group.
3. The new group alarm and number will be sent to the server where it is added to the database
4. The server will notify the user’s via push notifications.
5. Other group members will have to choose whether to accept or deny the invitation.
6. The user’s confirmation or denial will be sent back to the server, where the database will be updated with the active users.
7. Once user’s have accepted or denied the alarm the group admin (creator) will be able to view the active users of the alarm.



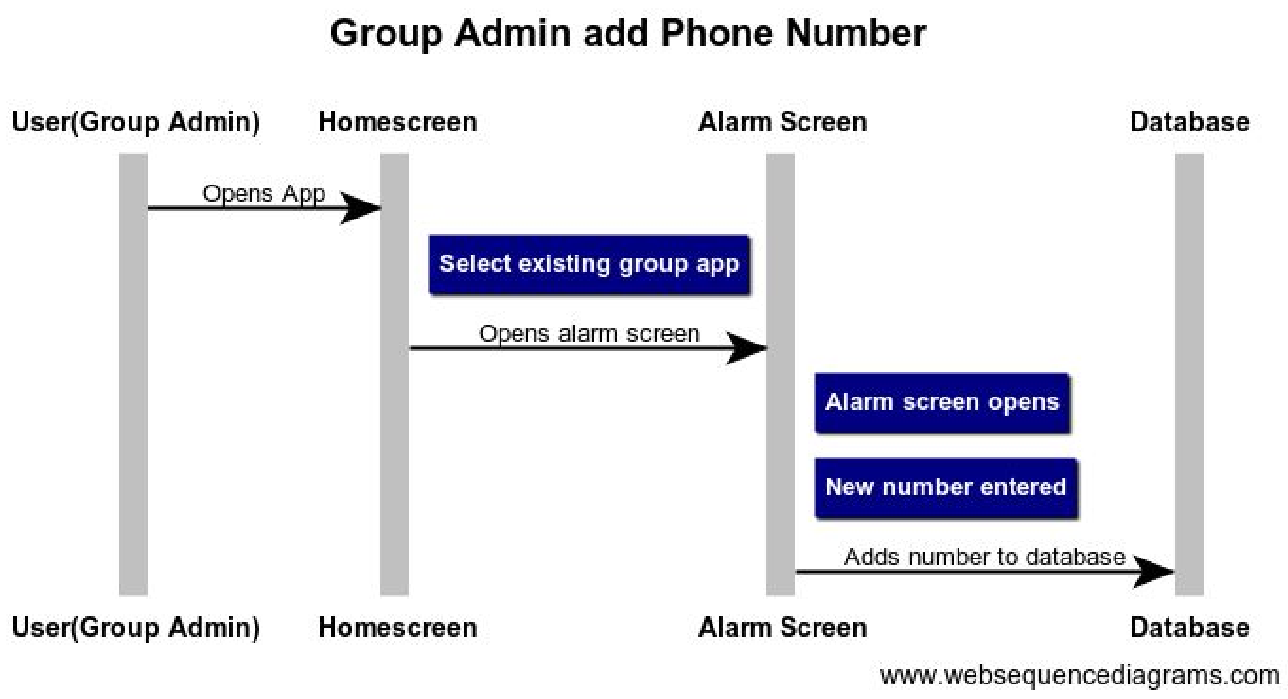
First time using App

1. The user will open the app and a one time login screen will be displayed
2. The user will enter his phone number and Name.
3. A verification page will be showed and a text message with a verification code will be sent.
4. Once user verifies the number, the back end service will check if the phone number already exists in the database.
5. If the number does not exist, the user will be added to the database and can start using the app
6. If the number does exist, the server will update the user's client and the user can begin using the app


Setting a Custom Text Message
User
Homescreen
Option Screen
Local Storage
Chooses to change options
Opens option screen
Option screen opens
User selects add custom message
User inputs custom message
Note saved
User
Homescreen
Option Screen
Local Storage
www.websequencediagrams.com

Setting a Custom Test Message

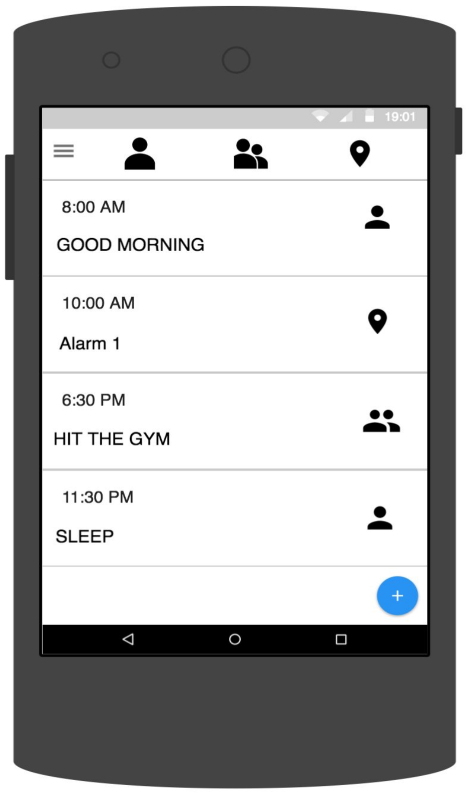
1. User opens the app and selects options
2. Option screen is displayed where user can select to add a custom voice or text message
3. User will insert the custom message
4. Custom message is saved in phone’s local storage and custom message now appears for the user to select when creating an alarm

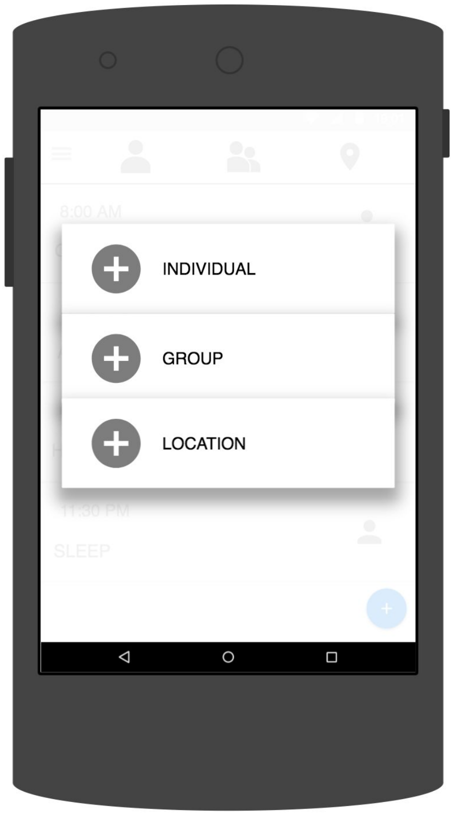


Adding A Member

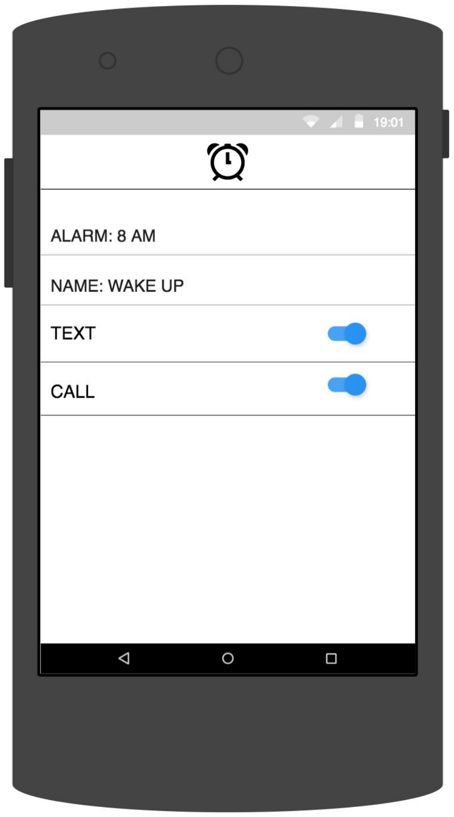
1. A user must be a group admin to add a user
2. The admin will select the group, opening the alarm page for that group
3. Admin will add number to the group
4. The number will be added to the group in the database, and the server will notify the user via push notification

## UI Mockups





Mockup 1 and 2: Home screen display Display for selecting type of alarm to create



ALARM: 8 AM

NAME: WAKE UP

TEXT

CALL

Mockup 3: Display for creating an alarm :