People Movement - Final Design Document

Introduction

People Movement is an Android Application on the front end that uses Firebase in the backend for its computations and queries. The fast-paced world we live in today requires the need to better mobilize and organize events like protests and marches. Our project offers planning for the mobilization of people in events such as protests and large gatherings. The app will result in a decrease in the likeability of violence breaking between two opposite groups of a protest and allow for a safer environment both for those involved in the protest as well as the general public. The front end work will be done by Shaked. She will be creating the interface for the Android app. The back end work will be done by Kate. She will be writing the cloud functions to deploy backend code. Together, both Kate and Shaked will be working on maintaining the database.

Users Interaction

User	Device	Interaction Level
Event Organizer	Android App	Use all functionalities of the app including creating events and providing event information
General User	Android App	All functionalities of the app including searching the groups database and joining groups

Description of Use Cases

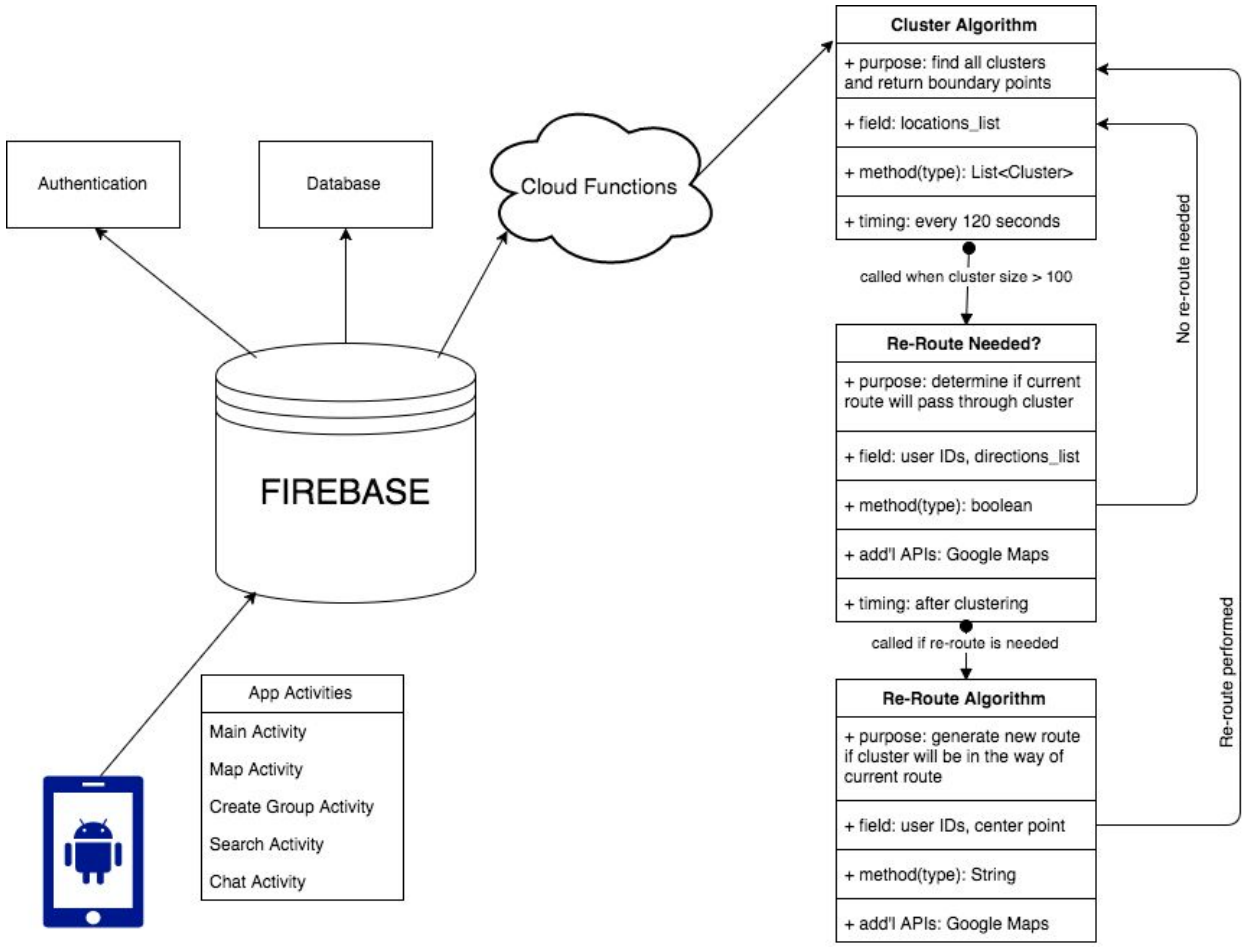
Use Case	Description
Create a group event	A user wanting to organize a group event will take advantage of our app functionalities of creating a group event and uploading it to the event database to allow other users to search for the event and join it
Join a group event	A user looking to join an existing group event will use our search features to search group events and join them
Receive updated walking routes	A user not taking a part of a group that wants to receive an updated walking route that takes into account events around the city

Components:

- Firebase - Cloud Functions
    - Functional Requirements:
        - **Cluster Algorithm**: scan map and determine the clusters of people interacting with the app
        - **Re-route needed?**: determine if any current directions are set to interact with cluster and return true if there are any such directions
        - **Re-route Algorithm**: use Google Maps API to generate a landmark to avoid and return landmark back to Android Studio to generate new set of directions.
    - Non-functional Requirements:
        - **Performance**: cluster algorithm must run every 120 seconds
        - **Responsiveness**: re-route algorithm must deliver new directions in real-time
        - **Scalability**: for delivery of the product in Spring 2018, functions must be able to scale for at least 100 people
        - **Security**: outside sources must not be able to change re-route algorithm return coordinates
- Firebase - Database
    - NoSQL Database
    - Functional Requirements: maintain structure throughout program execution
    - Non-functional Requirements:
        - **Performance**: maintain hierarchical structure
        - **Responsiveness**: updates in real-time
        - **Scalability**: database must be able to scale for thousands of users
        - **Security**: outside sources must not be able to change database queries and entries
- Android App
    - Functional Requirements:
        - **Group Creation**- create a group event and provide: event name, event start time, start location, end location, and event description
        - **Group search**- search for existing group events using the event name
        - **Map View-** provide map view for users to view routes and their group members' locations
        - **Directions**- calculate the best walking directions for the users
        - **Group Chat-** chat for users to communicate with their group members
    - Non-functional Requirements:
        - **Performance:** Come up with walking directions in a timely manner both for groups and individuals
        - **Responsiveness**: Calculate routes in real-time to provide updated walking direction
        - **Scalability**: be able to accomodate large groups thousands of people

- **Reliability**: Group members need to be provided reliable information to safely separate from a group knowing they could rejoin again later
- **Security:** detect what information given by users is correct and what is flawed

Firebase and App Interaction:
The app will interact with Firebase using the Firebase database reference in the app to send and receive data from Firebase, and implement listeners on the data to be updated to firebase automatically.



Database - NoSQL

User	Group
Longitude	Group ID
Latitude	Group Organizer ID
User ID	Start time
Group ID	Origin
Landmark	Destination

Classes

Class	Description
Map Activity	Map view to allow users to receive a walking route and view their group members locations
Main Activity	Home page of the app
Create Group Activity	Page for users to create new groups
Search Activity	Page for users to search for existing groups
Chat Activity	Chat for users to chat with other group members

Additional APIs

API	Description
Google Maps for Android API	Allow for map view on app and direction generation

Algorithms

- Clustering Algorithm
    - Hierarchical Clustering
      Algorithm is used to identify clusters from our users location data and returns an array of cluster objects where each cluster contains 5 coordinates
    - Algorithm:

1. Assign each location into a cluster of its own, so if we have N locations we have N clusters each containing one item. Let the distance between the clusters be the distance between the items they contain
2. Find the closest pair of clusters and if distance is less than 50 meters (1/32 of a mile) merge them into a single cluster so now we have one less cluster
3. Compute distances between the new new cluster and each of the older clusters. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

- Re-routing Algorithm

Description: This algorithm is deployed after the clustering algorithm returns a Cluster List<Cluster> holding the clusters generated by the algorithm. If a cluster is in the way of a current user's path, then re-route function will be deployed to give them a new set of directions avoiding the cluster.

Process (Cloud Function):

1. Obtain List<Cluster> from Clustering Algorithm
2. Create waypoint objects from List
3. Return objects to user's entry in database

Process (Android Studio):

1. Query Firebase for user's waypoints
2. If empty, stop
3. Else, pass waypoints into Google Maps Directions API
4. Generate new directions and send to user for update

Data Structures

- List<Cluster>: an array used by the clustering algorithms to store all clusters
- List<Users>: list to store all users. User object includes: user id, longitude, latitude, group membership
- List<Group>: list of all groups existing in the group database
- Map<LatLon>: maps the location id key with latitude and longitude value