Activitize

Design Document

Team 2:

Michael Golden, Richard Gorbett,

Tyler Preston, Tori Shurman

Table of Contents

1. [Purpose](#_kcn5jfydg8e8) 2
   1. [Functional Requirements](#_88avmsufcntb) 2
   2. [Non-Functional Requirements](#_x4e4x97n1c) 4
2. [Design Outline](#_6wb2jzroho76) 5
   1. [Client](#_9gpf7q7v215y) 5
   2. [Server](#_psg6kg2m71i9) 5
   3. [Database](#_l29gvtv8rgar) 5
3. [Design Issues](#_i5zeq313ms5x) 6
   1. [Functional Issues](#_s1gyt1iw4qi2) 6
   2. [Non-Functional Issues](#_3wl5jba4sevg) 6
4. [Design Details](#_wpk01wql9y99) 8
   1. [Database Schema](#_9n3p4rz8oed9) 8
   2. [Class Design](#_swttupawv8si) 9
   3. [Description of Classes](#_o59z2ofiqzk6) 9
   4. [Account Creation Diagram](#_7mw2s53hmlo4) 10
   5. [Event Creation Diagram](#_og6kl33smqyh) 11
   6. [UI Mockups](#_agt6021f6okh) 12

# Purpose

Currently, there is no one single application to use for planning group events. You can use a combination of messaging, Facebook, and GroupMe, but nothing that combines the strengths of these three applications. An application that would allow you to easily plan group events, including features like commenting on the event, or polling for event options, is especially important now in a world that connects almost exclusively online. We intend to create such an application in order to simplify group event planning.

Activitize is a social media platform based on events and activities. The basis for our application is that users, mostly likely college students and young adults, will be able to easily find friends to participate in events together and create new events or activities.There are applications that have features we are interested in yet nothing that contains all of the features we will be working on. Facebook, GroupMe and doodle all deal with events and have RSVP features like our application will. Facebook and groupMe lack the poll aspect of events. Doodle and groupMe do not capture the social media aspect of events. Activitize will be a social media application that revolves around events that will integrate features from Facebook, groupMe and doodle.

## Functional Requirements

1. Users can create and edit their profiles
   1. As a user, I would like to create an account.
   2. As a user, I would like to have profile that can be viewed by others.
   3. As a user, I would like to modify my profile.
   4. As a user, I would like to know that all of my information is secure.
2. Users can network with friends in the app
   1. As a user, I would like to add friends.
   2. As a user, I would like to search for new friends.
   3. As a user, I would like to be able to connect my facebook account to the app, so I can easily find friends.
   4. As a user, I would like to be able to create groups of people that I can participate in events with.
   5. As a user, I would like to see the events my friends are involved in.
3. Users can create and modify events
   1. As a user, I would like to create sub-events, i.e. events that are created for people attending the main event.
   2. As a user, I would like to be able to remove people from events I have created.
   3. As a user, I would like to be able to create a new event.
   4. As a user, I would like to modify events I have created.
4. Users can respond to and participate in event planning
   1. As a user, I would like to rsvp to events I am invited to.
   2. As a user, I would like to extend invites to friends for existing events.
   3. As a user, I would like to be able to comment on events that I have created or am invited to.
   4. As a user, I would like to favorite events I have created or am invited to.
5. Users can modify their privacy and preferences
   1. As a user, I would like to be able to mute notifications from events I am not interested in.
   2. As a user, I would like not to be required to change my password often.
   3. As a user, I would like to not have ads.
6. Developers can monitor app activity
   1. As a developer, I would like the user to be able to review the app, providing questions, comments, and concerns.
   2. As a developer, I would like users to be able to change their password.
   3. As a developer, I would like to allow users only 3 failed login attempts.
   4. As a developer, I would like there to be minimal time to load information on events and events of friends for the user.

## Non-Functional Requirements

1. Should be able to use this application on an Android device
2. Should be able to integrate with Facebook
3. Should be able to have alternate login method aside from Facebook
4. Should be able to utilize Google services in some-form
5. Should use some sort of near-real-time or real-time solution for updates
6. Should be scalable to be user demand
7. Should meet standard security practices
8. Should have an interface that is simple enough so that the average user would be able to use the application without struggling
9. Should be able to have negligible downtime while allowing for continuous development
10. Should be able to be high performing while maintaining proper security practices
11. Should have the ability to receive feedback in some form
12. Should document and provide API for other services to be able to hook into
13. Should design database schema independent of application so that it follows proper database schema design and can be ported for other uses
14. Should incorporate a three-tier architecture
15. Should separate business logic from client logic

# Design Outline

The application will use a client-server model. The client operates through an android interface by connecting to the server. The server will handle requests by the client; examples include: creating an event, joining an event, and modifying an event. The server will also handle the queries to the database. The database will store the necessary information about events and about each user. Below is a diagram about the setup of the client-server model.



## Client

The client sends requests to the server, as well as receives responses from the server. The client then interprets the response and displays them using the Android interface.

## Server

The server receives requests from the client. The server then interprets the request from the client and completes the action needed for the request. The server also fetches the necessary information from the database to complete the request. The server then formats the response, and sends the response to the client to be interpreted.

## Database

The database will be implemented in SQL, which should allow for easy queries that reduce the work done by the server. The database should handle data relating to the users and the events. The fields of the database will be further explained in the Design Details section.

# Design Issues

Below are some of the issues we encountered when designing our system.

## Functional Issues

1. In what format should users provide feedback on the application?
   1. Email link
   2. **In-app feedback page**

We chose to provide an in-app feedback page that sends an email to an account that we will create for the application. In this way, the user is not directed out of the app in order to send feedback which tends to be disruptive and can inhibit the user experience.

1. How should users be able to navigate between events, their feed, and other views?
   1. Tabbed view
   2. **Hamburger menu**

While a tabbed view along the bottom of the screen would allow the user to be able to access everything very easily, it could become cluttered if too many tabs were added. For this reason, using a hamburger menu for navigation between views allows us the ability to add more views without adding clutter, and it also keeps unnecessary information out of sight.

## Non-Functional Issues

1. Which language should we use for building an Android application?
   1. Java
   2. **React Native**
   3. Xamarin

We have decided to build our application using React Native because it allows us to potentially create both iOS and Android versions of the app without having to build two completely distinct applications. While Xamarin also allows you to build iOS and Android versions of an app using C#, we chose React Native because the components that you write in Javascript interface directly with the equivalent components for iOS or Android. While developing the app using native Java and the Android SDK would also be an option, it makes it more difficult to reuse our work if we decide we ever want to build an iOS application.

1. What login options should users have for the app?
   1. **Facebook**
   2. Google
   3. **Create a username and password unique to our system**

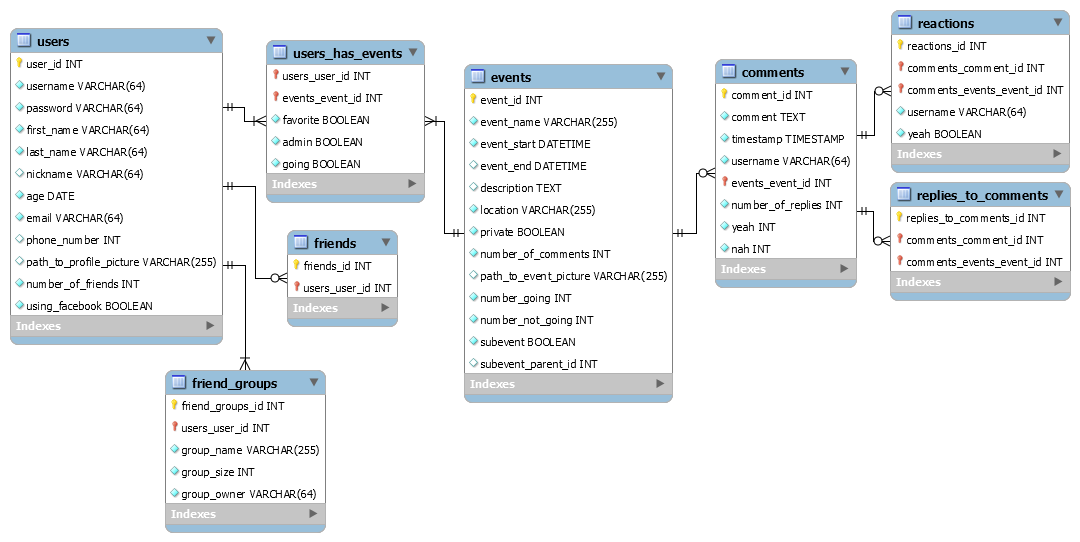
While providing a variety of login options would allow us to serve the widest variety of users, we have decided to allow users to login with Facebook or to create a username and password exclusively for our system. This would still allow us to serve everyone while simplifying the types of keys we would have to store. Providing Facebook login is still an important feature as most of our expected users, generally high school and college students and young adults, tend to have Facebook and use this to log into other applications. On the other hand, providing our own login system will allow us to provide for those users who do not have Facebook or who like to refrain from logging in with Facebook.

1. Which database implementation should we use?
   1. **MySQL**
   2. MariaDB
   3. Oracle SQL

When deciding which database implementation we should use, we ultimately chose MySQL. In comparison with Oracle SQL, MySQL stands out because it is open source, so is available to us to use at no cost, while still offering the features of Oracle SQL that we need. While MariaDB does still offer all of the features of MySQL plus some extra, we still chose MySQL in this instance because the additional features are not necessary for our purposes.

# Design Details

## Database Schema

The database is currently broken into 8 tables, all being related to other tables in one way or another. Some tables have one to many relationships, there is a case of a many to many relationship (which includes a join table), and several of the relationships even have a possibility of a zero or more aspect to them. Almost all of the data is normalized and is designed for efficient joins and queries. The data being contained includes user info, friend info, friend group info, event info, comment info, and replies and reactions to comments info. The schema was designed in a way that it was not biased by the actual programming requirements/restrictions, but rather it was designed in a way that reflects the relationships among the data and the best way that certain relationships can be realized when accessing the database. 

## Class DesignClass Design.png

## Description of Classes

User

* Class to describe all users, both those who login with Facebook and use our login system.
* Contains user information such as name, DOB, login information, and a url for their profile picture

Lobby

* Displays user information as well as friends, events, and favorite activities
* Can be viewed by other users depending on privacy settings

EventFeed

* Defines a display of multiple Events for users to choose between
* Structure is similar to a folder with multiple events inside

Event

* Contains all event information such as location, time, and type of event
* Contains a list of all admins for the event
* Structure is similar to a file inside of a folder

## Account Creation Diagram

Account Creation.png

## Event Creation DiagramEvent Creation.png

## UI Mockups

For a interactive walk through of our UI, follow this url: <https://goo.gl/qaFmKq>

|  |  |  |
| --- | --- | --- |
| Screen Shot 2016-09-19 at 1.46.29 PM.JPG | Screen Shot 2016-09-19 at 1.46.39 PM.JPG | Screen Shot 2016-09-19 at 1.46.49 PM.JPG |
| Landing Screen | Log-in Screen | Sign-up Screen |
| Screen Shot 2016-09-19 at 1.47.13 PM.JPG | Screen Shot 2016-09-19 at 1.47.02 PM.JPG | Screen Shot 2016-09-19 at 1.47.23 PM.JPG |
| Event Feed | Navigation Menu | Event Creation Screen |