Project Title: Silvous
Team Members: Travis McClure, Jimmy Tsao, Danny Vo, Sri Bandi
UT EIDs: tam2983, jt28593,dpv292, sb44884
Repository URL: git@bitbucket.org:ee461l_project/461lproject.git
https://bitbucket.org/ee461l_project/461lproject

1. Motivation

One of the most challenging things to do as a person is to meet new people. Some people have social anxiety, which makes it difficult to meet new people. Additionally, when someone moves to a new location, more than likely they won't know anyone. Whether this person has social anxiety or not, it will be hard to make new friends. This person might want to explore the city or area, but doesn't know what the best things to do are. On the other hand, if someone is with their friends and they don't know what to do, more than likely they sit at their house bored. It's tough thinking about something to do and hard to get your friends to attend the event. This android app addresses these problems.

There are many ways for people to find friends and see what events are happening in surrounding areas such as how Facebook has an events section and by looking for events on google. However, Facebook's events doesn't show all of the events happening in the area and many events don't show up unless you have a friend that invite you to the event or if you are invited to the event. Additionally, most of the events on Facebook are only big events, not smaller events. With this in mind, people don't have a way to know about the smaller events, which is something the proposed app will accomplish.

This app will allow people to meet new people and allow people to know and attend events that are occurring nearby, whether it's a big event, or a small event such as getting food with a stranger. Whether you would like to meet new friends or find something to do, this app is the perfect solution for you.

2. User Benefits

This app helps out more for people who are looking to make new friends, whether a person has lived in the area for their whole life, or if the person just wants to make new friends. Having social anxiety and wanting to meet new people is a tough combination to have. This app sets everything up for the people who have social anxiety, they just have to go to the event. The event is already set up and the people at the event want to meet new people too, so it's a perfect thing for someone who has social anxiety.

On the other side, this app is great for people who want to find something to do or they want to explore their area. The app will show all the events occurring within a certain distance of their current location. This will give people options to find something to do with their friends. For example, lets say its the summertime and it's hot outside and no one knows what to do. Someone looks at our android app and sees there's a pool party going on down the street. The friends go to the pool party and have the time of their lives. The friends found something to do and in the process they met new people.

If someone is new to the area and they don't know anyone, this app will quickly get them to meet new people. They go to any event they want and they will have met a lot of people. This will help the person get to know people in the area and will help them explore their new area and know what things there are to do.

Whether a person wants to meet new people or find something to do, this app benefits them both. Silvous allows people to come together and create lasting friendships. It lets people explore the area and find out how amazing their city really is.

3. Features and Requirements

Silvous in its current iteration provides the core functionality as described in the previous section. To meet these ends, Silvous features the ability to login via Google, create user events, and display nearby events on a map, then receive directions to the desired event. This required the usage of several Google developed API such as Firebase, Maps, and Places.

Silvous is intended to support individual settings for unique users. By requiring users to register/login to the app using Google sign-in, it attaches an email to each user and prevents the same user being registered multiple times. Once a user logins for the first time, a unique SilvousUser object is created for the user by using the unique login ID generated (and placed in the Firebase datastore). The SilvousUser object contains user preference data such as the distance at which they want events to be shown and so forth. Each following login will load the user preferences in the form of the SilvousUser object (from Firebase) by finding the unique login ID of the user. In addition, Silvous supports the ability for multiple users to login individually on each machine.

Silvous's main feature, the ability to create events is supported by several API as well as Android objects. When the user navigates to the "Create Event" page of the app, they are greeted by several text fields in addition to a Google Places location autocompleter. The autocompleter allows users to accurately find their desired event location, and on the backend side allows the application to determine the latitude/longitude of the event. The other text fields are there to allow the user to type in the name of the event, the description of the event, and the date/time of the event. After the user hits the submit button, all the user entered information is pulled using Android activities, passed to a SilvousEvent constructor, and then pushed to Firebase.

The complementary feature to the user event creation, is displaying user created events to a user of the app. which is done primarily with a combination of Firebase and Google Maps. When the user accesses the maps activity of the map, Google maps is pulled up and all nearby events are represented by markers. The user can click on the event markers to lookup details on the event, and then request directions to the event. If directions are requested, the app will automatically redirect the user to a browser version of Google maps with the route loaded and ready to be followed by the user to the destination. The events are loaded from the Firebase datastore to ensure all the event data is up to date.

With this suite of provided features, the application currently targets SDK 25 Android devices with support for devices running as low as SDK 14. However, at lower versions of Android software, devices may run into random issues with GUI elements such as the textfield or auto-completion. In addition Firebase may have hiccups. These issues were not encountered through the devices we used, and are potential problems inferred through the reading of Google/Android documentation. For debugging purposes, the usage of log statements was used. The ratings feature was scrapped, as it was determined to have little to no sense for events to be rated prior to their occurrence or during the event since events are wiped off the database as soon as they exceed their time period.

4. Design

Our app centers around the Users and Events. Hence, our primary defined classes are the SilvousEvent and SilvousUser classes. We use a subject-observer design pattern to communicate between these two classes, with the singular User being the subject and nearby Events being observers. This arrangement was chosen so that the User could track its location and then report this information to Events. These events then determine if they are nearby the User and display themselves onto our Google Maps screen. Additionally, we use Google Firebase to assist with storing our data, Google login services, and Google Maps as external submodules of our application. Our general class diagram is presented below:

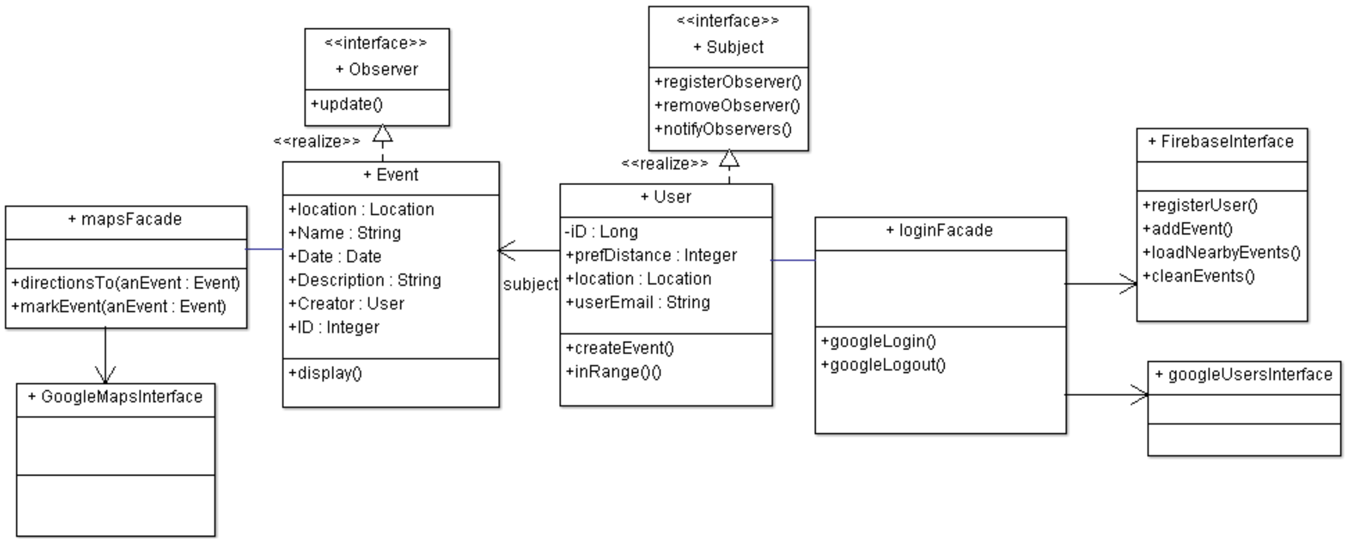


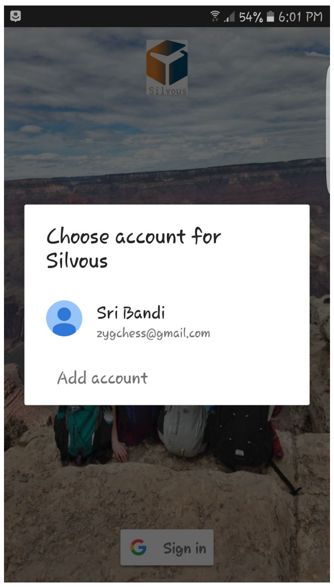
Figure 1: Class Diagram

All of our SilvousUser and SilvousEvent data is stored in a Firebase server, which is accessed in our loginFacade. Here, we call methods from the Google APIs such as firebase and the Google user services. This allows our app to interface with these rather complicated calls with minimal hassle. We also have a mapsFacade, which makes calls to the Google Maps API. Similarly to our loginFacade, this simply wraps calls to the Google Maps API in a format that is simpler to use.

The major external technologies that we used in this project were Google Maps,Google Places, Google Login, and Firebase. We also used a host of android utility features, such as Location, Log, and built-in gradle features provided with Android Studio. We used Google Places and Maps features to display our map, place markers at Event locations, and generate an auto-complete field for users to enter Event location in. Google Login was used to provide some user information and authentication, as well as to access the User's location for our targeted event display. We gained knowledge regarding the implementation of these features using the many Google and Android tutorials that can be found online.

Of the Android libraries that we used, we leaned most heavily on Intents, Activities, Location, and Date libraries, as well as a large number of smaller associated libraries. We used Activities to generate separate pages of our GUI, including our sign-in page, main page, map page, and Event creation page. Within these activities, we had Intents to take specific actions which are implemented through methods in our user-defined classes or in external APIs. Additionally, we used Location and Date to assist in our record-keeping and distance-calculating operations.

5. User Interface

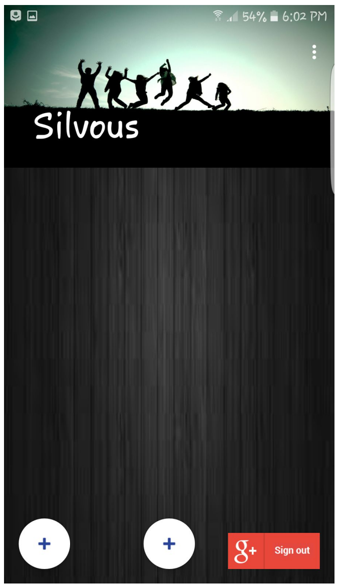


Choose account for Silvous

Sri Bandi
zygchess@gmail.com

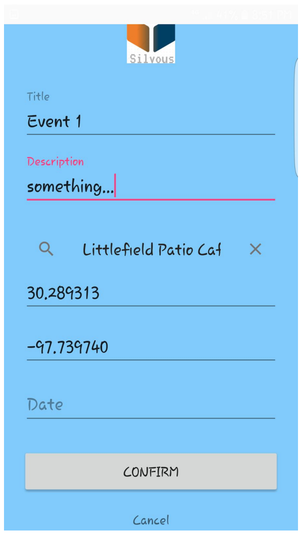
Add account

Screenshot 1



Screenshot 2

Our user interface was inspired from the ones similar to Apple's. We wanted to show that simplicity is elegance in the design and looks of Silvous. Screenshot one is the google sign in where you can select which account to log into. After successful authentication and login, screenshot two is the main screen. This main screen has four key buttons for user interaction. The first button on the bottom right is to successfully sign-out of this account and go back to the main screen. The design of this button was more tricky than normal because we needed to override the back button to not take us back to the main screen, and only let the sign-out button take care of that. The the menu option button on top shows up as setting which we have not implemented but added for future development and scalability. The first plus button on the bottom left takes the user to the third screenshot.



Title: Event 1

Description: something...

Littlefield Patio Caf

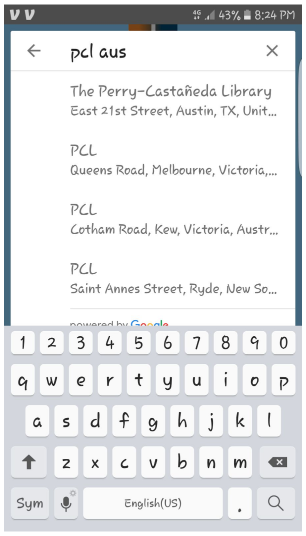
30.289313

-97.739740

Date

Screenshot 3

Screenshot three is the event addition option. The user can edit the details of the event with a title and a brief description. Then they can also add the address of the location of the event. Here our team added the option of google address auto fill. Once the “Search a location” is pressed, the app is taken to screenshot 4. As it shows, when the user types in any sort of address, it auto fills using the google auto fill. Then the “AddressLat” and AddressLong” in screenshot 3 are also filled because after selecting the address, the latitude and longitude are automatically fetched. The user can confirm this event or cancel it. After confirming, all the data is sent to firebase, and the screen automatically goes back to the home screen. The second plus button takes the user to the actual google maps where the events that the users add will be displayed. In screenshot five, the event that we added in screenshot three is displayed on the map. From here the user can click on the event and get connected to google maps for directions.



pcl aus

The Perry-Castañeda Library
East 21st Street, Austin, TX, Unit...

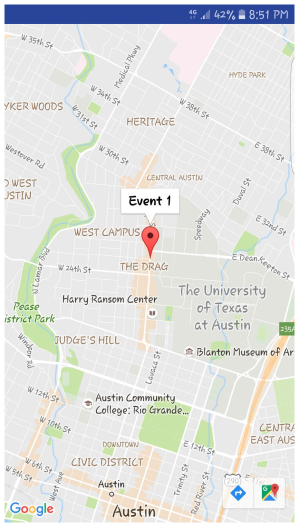
PCL
Queens Road, Melbourne, Victoria,...

PCL
Cotham Road, Kew, Victoria, Austr...

PCL
Saint Annes Street, Ryde, New So...

powered by Google

Screenshot 4



Screenshot 5

6. Testing

Unit Testing

This was primarily done through the creation of small JUnit test suites. We focused on functionality testing of individual class methods and whether they were properly constructed. Examples of this are the generation of random events/users, asserting their contents with their getter methods, and reasserting the objects after the usage of setter methods. Additionally, we started our test cases by using the following example inputs:

<u>Event</u>
EventName: TestEvent
EventDate: 12:06:2016 13:00
EventDescription: It's a test description!
EventCreator: jmtsao
Location: 2304 Whitis Ave #338, Austin, TX 78712

<u>User</u>
Name: Jimmy Tsao
Username: jmtsao
Email: jmTsao@utexas.edu

Component Testing

There was not an easy method to test on a component level using JUnit. so we utilized log messages as well as manually observing the contents of the Firebase datastore (Firebase doesn't initialize from JUnit testing). Component Testing was primarily focused on the accuracy of pushing and retrieving objects from our cloud storage system. To maximize the efficiency of this testing, we created a chunk of code to launch Firebase accesses on creation of the Android main_activity. The launching of several chunks of code was controlled by easily changed boolean values.

Integration Testing

This part of testing was primarily done through the same methods as Component Testing as we required the usage of Firebase for most features. We tested the interaction of GUI elements with the actual SilvousEvent that was being pushed to the cloud using a variety of event names, dates, and locations. In addition this was where pulling SilvousEvents from the cloud was tested. Several events were pushed to Firebase then were checked to see if they displayed or not when the map activity was accessed if they were in an appropriate distance range. The removal of events that had already "expired" was tested by pushing several events (both expired and yet to take place) then running the code for deletion.

Systems Testing

We tested the entire app first manually with two devices. One device logged in through google, then add an event while the other device is idle waiting for an update on the event. After updating, the second device was used to check to for the created event by accessing its map activity.