LunchR

"Find good food, on the move"

Module: COMP41690 Practical Android Programming 2016-2017

Members: Aaron Kelly, Dylan Dowling, Kevin Mahon, Mohsen Qaysi, Nikita Pavlenko

Abstract and Introduction

There are a number of apps available on the Android Market that seek to provide mobile users with a way to search, discover and review businesses in the dining industry, or that have a relationship with food - such as pubs, cafes and restaurants. Providing users with a mobile application that helps them "crowd-source" the best option when reserving a table or finding good place to eat, have a high share of install bases of all apps the market.

Our application is aimed at users wishing to simplify process of finding nearby places to get some food on the go. It offers places located nearby, that are filtered by the amount of time users are willing to walk, which is specified by a fluid gesture on main screen. As well as that, the application is aimed to add a social aspect to selecting a place, by providing users with an ability to get in touch with their contacts and organize a lunch collectively.

We felt that such an application may suit people who spend a lot of time in campus or office enviroment, where users within walking distance from each other may organize their lunch together.

How our app is different

There are a number of different apps in this domain, however a distinction is made between how they work - which is mainly to do with the service and that audience the app is targeting. The target demographic, and the type service being provided, heavily influences the functionality of the app and the way in which many of these apps go about connecting people to businesses.

Review focussed apps such as *Yelp!* feature their extensive database of community submitted reviews as their core selling point. Although they do have a range of services that also include business search and even food delivery, reviews are central to their offerings. In fact, their review coverage globally is so large that the app is actually classified in the "Travel and Local" category.

Booking/reservation style apps such as Opentable and Zomato offer powerful features that can help a user book and manage multiple restaurant reservations. They do provide search and discovery features, table reservations is chiefly their core offering.

Lastly, there are also food delivery apps such as JustEat and Foodpanda, however they chiefly cater to different needs than our app (food delivery) and will not be considered.

The aims of food discovery apps which feature local businesses that are not restaurants tend to be simpler more informal than an app that targets these types of business. They generally are not so concerned with bookings and reservations, rather they just wish to inform a user of a selection
of places they can find that fits their criteria.

One of the closest apps available on the Android market that matches the core values of our own is an app called "Restaurant Finder". Although geared toward restaurants, its feature set is similar to ours and the major method of searching is by the user's current location. The choice of listings are also similar - in addition to cafes and restaurants, it includes any businesses that could potentially sell food, like convenience stores and gas stations.

Links to apps mentioned in this section

https://play.google.com/store/apps/details?id=com.akasoft.topplaces # Restaurant finder
https://play.google.com/store/apps/details?id=com.application.zomato # Zomato
https://play.google.com/store/apps/details?id=com.justeat.app.ie # JustEat
https://play.google.com/store/apps/details?id=com.opentable # Opentable
https://play.google.com/store/apps/details?id=com.yelp.android # Yelp!

Literature Review

Adoption of mobile applications to assist users in finding goods and services offered by local businesses has dramatically increased from previous years - particulary in the dining industry. Indeed, the mobile phenonemon is so large it is considered separate from "online search", which search is conducted at a fixed computer terminal.

A study of data usage by Chinese mobile phone users shows that 50% of that usage was spent on mobile shopping, social media and websites (Bai, Hui, 2015). This paper also provided results of an online survey that showed that amongst a group of 209 respondents, only 5% of respondents had never used a mobile phone app, with 51% describing themselves as "frequent" users of restaurant mobile search appsl.

With the rise of smartphone uptake around the world, it can be safely assumed that similar trends could be observed in other countries, with traditional models of customer search in relation to food are strongly tending towards mobile search as their chief method.

Taking this into account, an app targeting users' needs for food and social activity, combined with it's unique search function and a potentially unique userbase that had not been penetrated, was determined to be suitable enough for development.

Consideration was given to how the app would provide recommendations, and its impact on user choice. Given that Google already provides an API which
gives access to business reviews, this was considered ideal. A study conducted in mobile recommendation systems for restaurants (Marques, Gabriel, et. al, 2016) show that a global rating for a restaurant was considered to be the most influencing in regards to collaborative decision making. For this reason, the LunchR app has reviews and rating featured prominently on the busiess info page.

A benefit was realised in that our app required no affiliation with the recommended businesses. A research paper featuring a single-case study of a restaurant showed that the overhead associated with forming a unique partnership with the business actually resulted in reduced profitibility (Lee, Sonny, et al.). It also demonstrated that this partnership required time spent on a ramp-up period, integration of IT systems and staff training. With this taken into consideration, it was determined that there was an advantage in our app requiring no interaction from businesses in order to function, which means it is positioned for immediate entry into the app market with an immediate benefit to the user and at no cost to the business. Although there is a point to be made about unfavourably reviewed restaurants losing business from the additional exposure of their review ratings on our app, it was agreed that a reviews of an unfavourable business would be transmitted on many different channels, be it word of mouth, printed reviews or traditional online reviews - of which our app would constitute a very small part.

References

Bai, Hui. "An examination of customers' adoption of restaurant search mobile applications." (2015).
http://aut.researchgateway.ac.nz/bitstream/handle/10292/9254/BaiH.pdf?sequence=3

Marques, Gabriel, Ana Respício, and Ana Paula Afonso. "A Mobile Recommendation System Supporting Group Collaborative Decision Making." Procedia Computer Science 96 (2016): 560-567.
http://www.sciencedirect.com/science/article/pii/S1877050916320452

Lee, Sonny C., Jia Sun, and Ping Wang. "Innovating Mindfully with OpenTable: A Restaurant's Experience." (2012).
http://terpconnect.umd.edu/~pwang/Lee+2012-ICIS.pdf

Originality

Our app differs from the above examples in a few ways, the first being the way search works. Our selection button uses *time* as the search method - this was thought to be appropriate as the assumptions that were made about the userbase would be that they would be on foot (could not travel far) and that there was only a limited amount of time they could use for travelling to the destination. This already makes assumptions about the user, reinforcing that it's primary userbase is targeted at walkers in the middle of large urban or campus areas.

A consideration we had when designing the app is that it would be as agnostic as possible when it came to the users country of origin. The quality of the results provided would simply a function of how much Google Maps knew about their current area - as long as their was an index of local food establishments where they were standing, then they would receive results. There is no special setup required or agreements that need to be made for different countries.

Contrasting how this works with the other app offerings on the market, the quality of results given by the larger apps depends heavily on a users country of origin - they specifically state on their app page which countries they work best in - inded some may not work at all due to non-integration of reservation/booking systems.

The restrictions placed on the app mean that the choice of results is narrowed, meaning our app gives much less search results than other apps offered on the marketplace. This was considered ideal, as the intention was to take the users in a given area and conentrate them

into a much smaller pool of destinations, reinforcing the social aspects of the app as they were more likely to find each other.

The social aspect is our final difference - many of the apps offered only offer reviews as the only social interaction. For the most part, the app is single-user targeted. Our app attemps to make the experience much more social, providing users of LunchR a way to communicate with each other, via the "LunchR Chat" activity.

Design

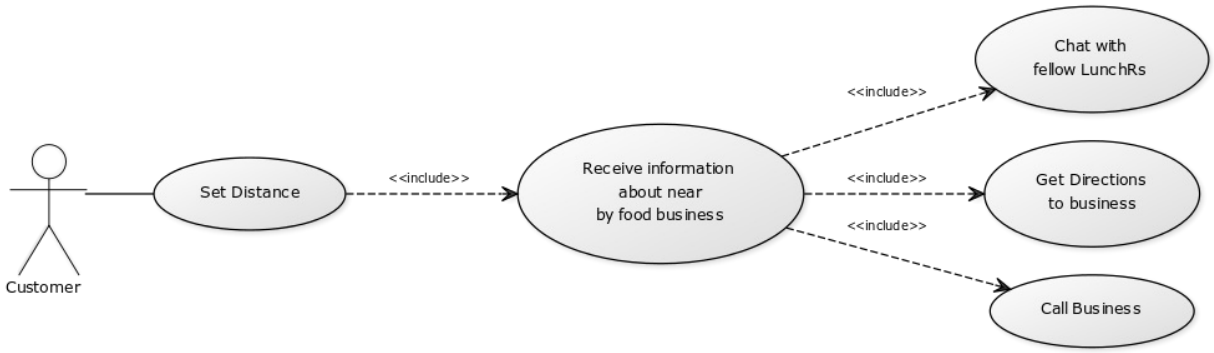
Design

Right from the beginning the idea was to have a very simplistic and highly intuitive interface. We felt that presenting the user with a clean interface with only two very simple functionalities (circle expansion & button click) would motivate the user to quickly take out their phone and get a result in seconds. Not having to wait, scroll, select, decide, plan or assess their potential journey. The app was designed to essentially remove the whole thought process and decision making involved with choosing where to eat in a spur of the moment. While this may seem ridiculous, it is in fact extremely helpful as countless times we've all been in situations where your friend or someone in a group says “where do you want to eat”. The idea of implementing a user filtered search was discussed, however the majority of the group felt that adding such a feature within the start up screen would conflict with the original plan to create a minimalistic interface. We decided to have no filter and if they don't like the result they can quickly search again.

Target Audience

Our app is mainly aimed at people age sixteen and upwards who are situated in urban areas. We say sixteen and upwards as typically those who are under that age do not have an income and therefore would not be interested in searching for a random food business near by. However, for outliers (those who are sixteen and under with money in search of somewhere to eat) the app is perfectly accessible. The reason for even specifying sixteen is really just to have a light definition of our target audience. Repeating what was stated above, it is for people situated in urban areas. This may be obvious but the reason for this is simply down to the business count in their area. That is, if you are using the app in a remote area and you specify to walk the max distance, it's very possible that you will not get any results.

Use Case



Implementation

The application was developed from a simple idea which was the elimination of all the thought process and decision making involved with choosing a place to eat in.
The user will be presented with circle expansion to choose the distance they are willing to walk. (5 , 10, and 15 minutes) which are fixed measurements. Those measurements represent the distanced used in the request which used with *Google Maps API* to search for nearby places.
The flow of the data from one activity to another is as follow:

- **Result activity:** pass the current location lat and lng as well as the Radius the user choose to walk in the case it is in minutes (5,10,and 15) to the Selection activity.
- **Selection activity:**
  It used the data passed to it from an intent to allow it to make the *Http Request using Google Maps API* to search for location based on the input. An example of the request:

```
        String url =
String.format("https://maps.googleapis.com/maps/api/place/nearbysearch/json?
location="
                + myCurrentLocation.latitude + ","
                + myCurrentLocation.longitude
                + "&radius=" + radius +
                "&types=food"
                + "&key=" + APIKEY);
```

- Results:

```
{
    html_attributions: [],
    results: [{
        geometry: {},
        icon: "https://maps.gstatic.com/mapfiles/place_api/icons/lodging-
71.png",
        id: "ff7867be0da413f728ef874868ffd8b38d806c9d",
        name: "Pembroke Townhouse Dublin",
        opening_hours: {},
        photos: [],
        place_id: "ChIJXx-vL5mEREgRI01b8tvih0E",
        rating: 4.1,
        reference:
"CmRRAAAAgBt49LXBlkOZatPWnL8Xoh01N1TU1S1N9lIzspCgtlRtK4K_eM_SBizAYD2Jq7ZNvhZ1rA
kjWAWsWRKY2cGzcwlICtfznjQdFjBHJC7lC2Ky8FBKOFwk-iZp-v1va3coEhC-EzT46XA0-
NclIY1GDhlYGhQKh_RWInflLD2_CV8sFUjYkPr-Jg",
        scope: "GOOGLE",
        types: [],
        vicinity: "90 Pembroke Road, Dublin"
    }, {}, {}, {}, {}],
    status: "OK"
}
```

The result of the request is JSON ObjectArray with locations.We are only interested on some key in result which are geometry, place_id, and name. Moreover, We store all of them and use them to popluate the *ListView* and *Map Markers* for each location. When clicking on a marker you will have the ability to open the location on *Google Maps App* or get *direction* to the place from your current place using *Google Maps* app installed in your phone.

-**Info activity:** The idea was when the user decides to which location he/she wants to go to or get more information about it. They can click on the desired place from the *ListView (an **Intenet** will be triggered to pass the ID of the place to the Info activity. This ID will be used to make an Http request to maps Google APIs to retrieve data about that specific location as well as getting an image of the place using its **"photo_reference"** see the request bellow)* which will take them to the *Info activity* to get more rich details about that place such as an Image of the place, the place rating, is it **Open NOW** or **Closed** as well as you can check their website for the food menu if they have one.

```
// Build a URL for the image request:
Uri builtUri = Uri.parse(REQUEST_LOCATION_PHOTO).buildUpon()
                .appendQueryParameter(MAX_WIDTH, Integer.toString(imageSize))
                .appendQueryParameter(QUERY_PARAM, photo_referenceVal)
                .appendQueryParameter(LOCATION_ID, APIKEY)
                .build();
// Using Picasso library, we can load the image really easily.

Picasso.with(this).load(url).error(R.drawable.placeholder).into(myImage); //
Display the image of the location
```

-**Social activity:** This part of the application is really cool. Compared to other apps in the market our app is different we do not require you to know anyone around you like other apps which require groups of people to know each other. You can go to the social section of the app to see and chat with people using the application. You can set up a nickname to use and start chatting without the need to sign up.

Lessons learned

- **Result activity:**
  The circle in the centre of the screen is meant to represent a radius on a map. Expanding the circle increases distance. Although in the activity we do not tell the user the distance exactly, rather we specify the minutes it will take to walk to the destination. This is much easier for the user to understand rather than calculating the distance from their position.

  In order to create the circle, a custom view class was needed as the circle needs to be re-drawn every time a finger is put on the screen, moved and lifted off.
  The first part of drawing the circle onscreen was to get the devices screen width and height. Then dividing these in two we can get the centre of the screen.
  The next step was to recognise when a users finger was on the screen, moving on the screen or has been lifted off the screen.

  When the user places their finger on the screen I immediately get a point on the screen. As they move their finger I calculate the distance from that point to the point where their finger is on the screen. Then once the finger has been lifted I set the new radius. The max radius is set to the height of the phones screen, so that the circle will not go beyond the height of the phone, although this was a major challenge I encountered. When working with custom views, sometimes getting the screen size proved to be quite a pain. Eventually is was pointed out that I could use DisplayMetrics in order to get the height in pixels, this appeared to work on the devices it was tested on.

The part I spent most of my time on was learning how to draw the circle continuously as the user moved their finger around the screen. While the code I had written seemed correct, I couldn’t understand why the circle was not being drawn repeatedly on screen. Eventually I discovered that I needed to use the invalidate method. The invalidate method basically means re-draw on the screen. When something in the view changes, in order to reflect that change on screen invalidate method needs to be called. This is particularly important when working with custom view classes. Unless using a a custom view you may not see this method as it is called in the background automatically when say for example you set the text of an object using an inbuilt setText method.

- **Selection activity:**
  Key idea of this activity was to integrate populated components into it. The activity consist of a Map fragment and a ListView fragment. These fragments represent information about the same places in two ways: a location(pin) on the map that would lead the user to directions, and an entry in a listView which would allow to find out more infromation about the business inside of the Info activity.

  Initial step was to retrieve information about nearby places via an API call, data from which is returned in JSON format. While doing this I've learned how to use Maps API to get and parse JSON output for necessary information.
  Populating the data onto the screen turned out to be challenging as it wasn't straightforward for neither of the fragments I've placed on the screen. In the end ListView required to be created as an external activity and then put into a fragment of a screen, and Map fragment required a lot of trial and error to get it to display all of the necessary location pins. During this part I've found that different sources of learning Andoroid than Google Developers also came useful, as a lot of time was spent trying to learn about issues that may have not been described there.
  I found that there was quite a bit to learn in terms of basic components of mobile app developement, and this may have had an effect on planning part of an application. However, having worked with some of the components now, I would certainly feel more comfortable to attempt creating another application in the future.

- **Info activity:**
  Mobile development is a little bit different than software development for desktops. However, mobile development is a bit harder because most of the time you are working with a smaller screen size and trying to fit a lot of things in it. One of the mistakes new mobile developer make is not giving the planning phase a lot of time and jump into development straight away. I faced some of that during the development of this project. I chose **Google Place API** as my initial API to use to get information about a specific place. At a later stage, I realised I made a mistake and I had to rewrite the code from scratch because, for some reasons, not all the places give me the right details I wanted using **Google Place APIs** . Moreover, using an Http request using **Google Maps API** was more reliable and works all the time with a lot of flexibilities. I had to parse the JSON Object from the request, but I got to have control over the data to take and remove.

- **Social activity:**
  A major desire was to have persistent data shared across all users of our app, that did not expire after the app was suspended or closed. Information was given in class on implementing local storage, but the idea was posed if we were able to create an activity that used both local and remote storage, in order for data persistence and communiction between our app userbase. The idea was to create a social chat room environment for users of the app.

Initially it was proposed that a very simple web server could be used to handle POST and GET data. Although this was attempted, it turned out that the POST and GET methods are not handled automatically by simple HTTP web servers - php functions need to be written to properly handle the requests, and local storage needs to be managed by the administrator also. Within Android Studio, there are a selection of services offered by *Firebase*, via `Tools -> Firebase`. The *Realtime Databes* option offered a number of features suited the use case for the activity, and because of the tight integration with Android Studio, using it worked pretty much out-of-the-box.

Final conclusions

In conclusion, it was both a challenge and a reward to work on a group software project together, particularly when it required elements such as design, user interaction and storyboarding - in addition to the programming. Building an application for a target demographic of phone users is notably easier when the devloper actually knows what the user wants. It threfore helped that each of the team members all sought to have a social-lunch based application on their phone for their own benefit. As the team are all students, it was relatively easy to construct the ideal kind of storyboard we all wanted for the application. Since we each experience difficulty on a frequent basis with having lunch as a group during term time, our goal and the process to get to it was far more clear than it would be if we were developing the application to address a niche market that none of us understood.