MSIN2009 Group Project

# WHAT'S MAPPIN'


Yanting Li
Edouard (Teddy) Favre – Gilly
Onur Ozturk
John Oikonomou
Teresa Ricciardi
Davin Souza

Group X

# Executive Summary

This report provides a summary and evaluation of the development process of the creation of an events-listing platform/app as well as a business solution using the Scrum software engineering method.

The goal of this project was to deliver an innovative platform that entices millennial party-goers to attend and discover events tailored to their profile, and develop a network of business relationships with London venues and hosts.

Our initial idea was to create an events-mapping application, listing events in London on any given day. Upon group review we decided to refine our target market by concentrating on University of London students. This decision was made on the basis that the target demographic were early adopters of technology, were susceptible to monetary offers and enticements and were a demographic our group felt we could identify with.

Consequently, we decided to change the scope of the events we were listing, focusing entirely on nightlife; concerts, live music nights at pubs, late night bar happy hours and so on.

The project‘s output is the prototype of our events-listing app, “What’s Mappin?”, demonstrating its initial essential functionalities:

· Mapping of events to enable our user to visualise the availability of events in their vicinity.

· Create user profiles, drawing data from social media that can be further enhanced by behavioural data observation.

· Create a marketing platform through which venues can promote their events.

Despite having undergone two prototype iterations, given the nature of our project and its timescale, many of our application’s features have not been implemented yet. However, our product roadmap clearly outlines the features “What’s Mappin?” will include in future iterations.

Using the Scrum methodology of software development, our agile and dynamic team produced a minimal viable product ready for launch. This prototype is accompanied by a detailed report outlining our business case, requirements, analysis and design allowing any potential stakeholder to envisage and endorse our product.

# Introduction, Problem Outline & Solution

What's Mappin' (WM) is an events-listing platform that intuitively displays events geographically to allow users to visualise event availability.

Our innovative platform addresses a very specific problem. Many groups or individuals plan to meet up and socialise with a view to going out to *events* after. What we consider *events* are attending club nights or concerts that are in line with the group's interests.

That said especially where groups are concerned, choosing and planning an event beforehand can be a difficult process. Not only does everyone need to confirm their availability for a given date and time, but all group members must book their tickets for the event before it sells out. What's more there is always the likelihood that when meeting up beforehand, the group feels less enthusiastic about the previously decided plan and wish to go elsewhere.

Our app allows for spontaneity. The platform displays events geographically on a map, so as to see what events are on in the user's vicinity. A Nesta report showed a trend of nightclubs and venues closing in the city centre and an increase in city periferral venues opening (https://www.nesta.org.uk/blog/clubbing-map-what-has-happened-london-nightlife) (such as the new 2015 Printworks and the rise for Shoreditch as a 'clubbing area'). Our platform capitalises on this trend bring the nightlife towards you. It is increasingly likely there are venues opening outside the long-established Mayfair and Soho areas, and nearer to our user's location. The £26.5bn nightlife industry (https://www.nesta.org.uk/blog/clubbing-map-what-has-happened-london-nightlife), relies heavily on drink and merchandise sales. The average London partier spends £19.67 (https://www.eventbrite.com/blog/bar-management-for-music-venues-ds00/) on drinks on a night out and consequently venues would benefit from increased attendance and filling up capacity.

By displaying nearby events, we are encouraging users to attend events nearby so as to make sure that venues are operating at maximum capacity. A group looking to go out could be offered last minute incentives to attend events that have not sold out of filled up.

Another issue our app addresses is event exposure. Many young venues struggle to gain the exposure to attract the necessary clientele (https://www.nesta.org.uk/blog/clubbing-map-what-has-happened-london-nightlife). With the clubbing industry suffering a loss in revenue of 8% since 2013 (https://www.ibisworld.co.uk/industry-trends/market-research-reports/accommodation-food-service-activities/nightclubs.html), venues are having to market themselves in new and innovative ways. Despite this, the number of venues opening each year is on the rise. Our platform would allow venues to offer targeted discounts and incentives for events to users that suit their demographic.

A major element of our platform's functionality would be gaining information on our user. By encouraging users to create a profile or sign in through social media (described in data subsection) we would instantly access information about our users age, sexe and interests or 'likes' amongst many other forms of data. To optimise our knowledge of our user, our application would gather data on our user's behaviour, so as to learn how to optimise our targeted events marketing strategy (detailed in optimisation/data/user feedback section).

# Business Plan & Monetisation

We consider user experience and user interface to be key to expanding our user platform. Hence WM does will not be using external advertising banners or pop-ups of any from on the application. All our form of advertising will come incorporated in the app's functionality, listing events to targeted users.

For this reason, we have devised a monetisation plan that is based on venue referral commission. In this section, we will define level 1 referral, level 2 referral, and ticket sale commission.

Level 1 referral:

As can be seen from our app demo and mockup screenshots, when a user browses the map listings, they can click on an event *marker* to view information on the event from within our platform. We consider this to be a step 1 referral. This enables the user to find out the name of the venue, exact time and location, event details and line-up. This gives the venue a level of exposure. Our application will have informed a targeted user about what the venue has organised for the evening. For this level of referral, WM will incur a fee of £0.02 on behalf of the venue.

We consider this fee to be worth well below its future value, as if a venue were to get level 1 referrals from 50 users, they would have spent £1 on customer acquisition. Were just one of these 50 referrals to result in a ticket sale – assuming the venue is hoping to make more than a pound's profit on each attendee – the acquisition cost would be outweighed by the venues revenue-making capacity.

Level 2 referral:

If the user is particularly interested by the event information we have provided through our platform, they may choose to continue to the venue's website. From there the user would see the venue's entire marketing platform, increasing the venue's website traffic and giving the user a fully immersive description of the venue's event that night. Given the event hosts' confidence in their website design and marketing strategy content, website traffic is a welcome catalyst to ticket sales.

For this level of referral, WM will incur a further fee of £0.04. If 25 such users were to navigate onto the venue's website this would incur a fee of £1. It should be noted that these users, tracked by their device's IP address, now on the venue's website, are users our platform have deemed statistically likely to attend the event.

Ticket sale commission:

Finally when one of these users purchases a ticket for the event, WM will incur a commission fee based on the ticket sale. Initially there will be no commission fee so as to prove to venues that we are generating ticket sales. Once we have sufficient sales figures we will approach venues with our sales figures on their behalf and announce our monetisation method. As such, WM will provide a marketing platform free until we have gained sufficient exposure to leverage commissions costs. Venues who do not agree to our business model can negotiate or

choose to be excluded from this London-wide venue listing platform. For tickets over £5, a commission of 15% would be incurred.

Venues offering cheap tickets hoping to boost revenue through drink and merchandise sale will be able to grant us the permission to offer incentives such as offering our users a free first drink in their venue. Listings below £5 would incur a flat fee of £1. This is because ticket prices under £5 would not yield enough commission fee alone to justify the increased number attendees our platform will have provided. Incentivising is a major part of our platform's user-base expansion program (as detailed in marketing section) and we would welcome the opportunity to offer WM exclusive deals.

Scenario 1

If Club X were hosting an event which had not sold out, selling tickets for £3 and offering £2.50 drinks hoping to attract students on a week-night. An individual could encourage their group of three student friends to download our app and could use our platform to be directed toward Club X's WM event marker, then through to their website, and purchase 4 tickets, each receiving a free drink inside.

The WM fees incurred would be £0.02 (level 1) + £0.04 (level 2) + £1 = £1.06 for each user.

The free drink (beer or standard mixer) would likely cost the venue 50p. Despite an opportunity cost of £2, the free drink is shown to increase spending in venues (https://www.washingtonpost.com/lifestyle/travel/at-some-vegas-casino-bars-non-gamblers-lose-their-free-drinks/2017/05/11/d77a35dc-31cc-11e7-9534-00e4656c22aa_story.html) , increasing total drink sales, and hence would likely yield multiple sales of £2.50 drinks, each generating £2 profit.

So for scenario 1, the club has paid a customer acquisition fee to WM of just £3.18 and offered a free drink (£0.5) per ticket sale. Consequently it will have received £5.82 of ticket sales otherwise forgone and have incentivised their attendees to spend more during their evening at the event generating £2 further profits per drink bought.

Level 1 referral fees are a necessary component of WM's monetisation strategy, as our platform makes event announcements look exciting and enticing so as to encourage users to engage with the application and be more informed of the event landscape on a certain evening. The more the user engages with the events listed on our app the more level 1 referral fees WM will receive and a user could generate a number of such fees just by browsing through our platform, without even having to secure a payment through the application.

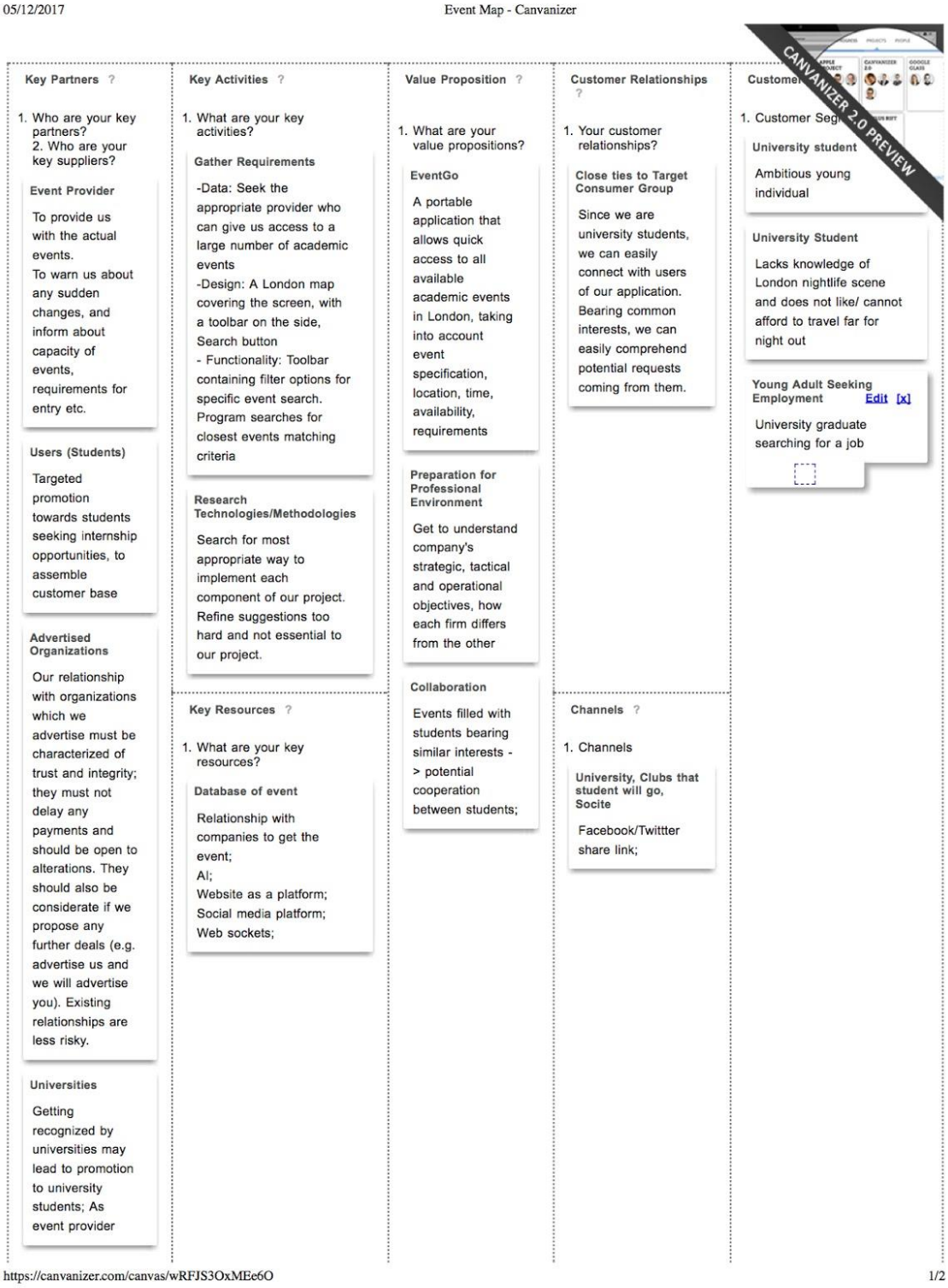
Another method of monetisation would be encouraging venues to promote their event further on our platform by purchasing certain advertising packages. Not only by offering our users targeted offers and incentives but also by purchasing listing promotion packs from the WM platform a venue could significantly increase its exposure to keen party-goers.

The reason we have decided to target students of the University of London as our target demographics in four-fold. The first is that we associate with that demographic and understand the users need. The second is that this demographic has a high price elasticity of demand and can easily be enticed by offers and discounts which are the core customer acquisition strategy of our platform. The third is that the WM platform needs to create a loyal customer base and to do so we are choosing to focus solely on London students. This will

enable our platform and logo to be associated with going out for a whole generation and demographic of London society. This leads on to the fourth point that this customer base will graduate and enter the job-market, becoming even more monetizable, and a trend in London Universities would quickly spread to other universities through such media and social media pipelines as ‘The Tab’ and consequently could spread to universities in other countries and culture.

The reason we are keeping our referral and commission fees low for our app launch is to encourage venues to use the WM platform so as to get a large enough pool of venues listing that our users feel that our platform represents a good proportion of each day’s nightlife canvas.

# Business Canvas Model



# Roadmap

As our platform is merely in its infancy, our main objective during our project's timeline was to produce a minimal viable product ready for market so as to begin grabbing market share and growing a user-base. Although our features include the ability to refine events, search etc, there are many features that WM would like to incorporate into its platform in the future.

First of all, once the user base is established, WM could expand into other events, gaining users from different demographics and with different interests and diversifying its listing platform. This would enable higher revenue, greater market stability, in case the nightlife industry began to suffer, for instance, and could increase our prominence and leverage over event holders.

Another consideration for the future could be that WM could host its own events. Much like certain ticket retailers such as New York Resolute have now integrated, hosting events in venues, WM could capitalise on its image of cool young spontaneity and start hosting live music events in London, promoting heavily to our user base.

We would also like to optimise our app to our user's preferences, and to do so would require time to collate user response and click data to gauge where and how we could improve. WM envisages including more 'filters' in its app side-bar so as to allow the user to refine their event searches to a greater extent.

As far as the monetisation of the platform, once WM becomes an established events listing platform with a loyal user base of soon-to-be working graduates, we would have a greater level of leverage over event hosts and would be able to charge greater commission and referral fees for our services, with a view to making the company profitable within 5 years.

A final feature WM considers for its roadmap would be a ticket swap feature, whereby people could swap tickets last minute with other users to gain access to otherwise sold out events. Once WM amasses sufficient users, there could be a big market for last minute swapping.

# Scrum project management

During this group project, our group worked in a cohesive, agile manner. Although we somewhat adopted the titles of software developer, UI/EX leader, testing specialist etc, every team member was able to contribute all sides of the project. As such we were following the basic principles of Scrum.

YanTing Li took the role of Scrum Master, organising and logging all sprint planning and reviewing sessions. YantTing also took close control of the requirements and formed the MOSCOW analysis contained below.

Teddy Favre-Gilly took the role of Product Manager, acting as the link between the technical and business persons within the team. In terms of output, most of Teddy's contribution came

in the business and market analysis section, where he devised the platform's monetisation strategy and collaborated with Teresa Ricciardi on the marketing and go-to-market strategies.

Teresa Ricciardi lead the marketing side of WM, producing an in depth report of marketing strategies used previously by apps targeting our same demographic and consequently devising a marketing strategy specifically for What's Mappin'. Teresa also collaborated with John on the competitor analysis below collating information on all of our platform's potential competitors and listed in bold below each case, why WM is different and why our user might choose our platform over our competitors.

John was head of the UX/UI team responsible for prototyping the app working with the Java code Onur had already implemented. As well as this John contributed to the competitor analysis, use case diagrams and was essential in thinking up of filtering features the product displayed.

Onur Özturk acted as our main developer, initially writing Java code to get our idea up and running to create a first prototype. In later iterations, Onur was instrumental in creating UML diagrams, sequence diagrams and statement sheets.

Davin was our team's main tester, running tests on each of our prototype's iterations, conceiving our data gathering and use strategy and was further central to creating all of our UML diagrams.

| | | Priority Level | Task | Team Member(s) | Description | Started | Complet | Notes | Links---> |
|---|---|---|---|---|---|---|---|---|---|
| Teresa | John | 5 | Competitive Analysis | Teresa & John | Have a look at the apps already 'out there' and other tachnologies th | X | - | Our USP | https://itur |
| Onur | Davin | 5 | Location Tagging | Onur & Davin | Learn to pull a map natively onto our app and for now just geotag ev | X | - | Keep a note of techn | |
| Teddy | Teddy | 5 | Backlog & Prioirities List | Teddy | Create clearer structured shared docs so people know their tasks ar | X | continuou | Let me know in here | |
| Yanting | Yanting | 5 | Mind Map | Yanting | Make sure all 'requirements' are in the chart so we can start filling in | X | continuous... | | |
| John | John | 4 | Business Model Canvas | John | input | X | continuous? | | |
| Teresa | Teresa | 4 | Stakeholder Analysis | Teresa | Stake holder analysis looking from the 1. User's perspective (UX, UI | X | - | | |
| Teddy | Teddy | 3 | Stat Attack | Teddy | Lookup stats on clubbing figure, revenue, demographics, startups, V | - | | | |
| Teresa | Teresa | 3 | Chat to Frank | Teresa | Find out what event planners seek from promoters & users. What have we not considere | | | note down anything | |
| Yanting | Yanting | 5 | Project Schedule | Yanting (ScrumMaster) | Keep a log & outlook on our timeline & deadline | X | continuous... | | |

All the team were aware of what work they had to prioritise thanks to the Backlog initially created by Teddy (version 1 above) and then improved upon and optimised by Yanting (version 2 below)

**What's Mappin Executive Backlog**

| Spri | Stat | Backlog Item | Section | Descripción | Owne | Support | Execute time(ho |
|---|---|---|---|---|---|---|---|
| 2 | √ | Who is the competitor | Business Case | Have a look at the apps already 'out there' and other technologies that could rival our app. Then look out what we could do differently to them. Look at their rating - are they good or bad? Look at their 1 star ratings - what do the users who are rating 1* say about the app. Can we do something to attract those 1* customers? e.g. - I have noticed in the "events near me" app, there is no map function in Android version - we are doing an Android app! | Teresa | John | |
| 2 | √ | Location Tagging | Design | Learn to pull a map natively onto our app and for now just geotag events onto the map. No need to think about date tagging yet or time, lets just keep it simple and build bottom up :) | Onur | | |
| 1 | √ | Backlog & Priorities List 1.0 | Scrum Management | Create clearer structured shared docs so people know their tasks and what they're supposed to be doing. | Teddy | | |
| 1 | √ | Mind Map | Scrum Management | Make sure all 'requirements' are in the chart so we can start filling in as we go along - maybe have a checklist of "started" and "completed" on the side? | Yanting | | |
| 2 | √ | Business Model Canvas | Business Case | input | John | | |
| 2 | √ | Stakeholder Analysis | Business Case | Stake holder analysis looking from the 1. User's perspective (UX, UI etc), 2. App's perspective (monetization - I will help you with this), 3. Events companies we feature (promotion etc) | Teresa | | 3 |
| 2 | √ | Stat Attack | Business Case | Lookup stats on clubbing figure, revenue, demographics, start-ups, VC investment etc | Teddy | | 2 |
| 1 | √ | Chat to Frank | Business Case | Find out what event planners seek from promoters & users. What have we not considered? (Frank Yuen) | Teresa | | 1 |
| 3 | √ | Project Schedule | Scrum Management | Keep a log & outlook on our timeline & deadline | Yanting | | 3 |
| 3 | | Competitors List | Analysing | Give Teresa all the competitors you find by excel sheet | John | | 1 |
| 3 | | Client Roadmap | Design | What will a student and client see when using the app? What pages will they go through? Registration page(Facebook and Instagram integration), Filter page, Map page, Setting page. Etc | John | | 3 |
| 3 | | Textual use-case | Design | Check the example in Moodle | Onur | | 3 |
| 3 | √ | Ux Interface for each pages 1.( | Design | example in Dribbble:<br>https://dribbble.com/shots/2747314-Event-Discovery-App<br>https://dribbble.com/shots/3258253-Map<br>https://dribbble.com/shots/3749277-Receipt-App<br>Or others you have found | John | | 4 |
| 3 | √ | Pricing Mechanism of Compet | Analysing | The three task is the sections in the analysing part. Check the marking scheme for reference | Teresa | Teddy | 3 |
| 3 | √ | Spending of Average student | Analysing | | Teresa | Teddy | 3 |
| 3 | √ | How to attract student custon | Analysing | Customer Acquisition cost | Teresa | Teddy | 3 |
| 3 | √ | Route map functionalities | Design | Write down what comes in your way and tell the group asap, the group will try remove the barriers, if not, we will record it into sprint meeting. The difficulties you face will goes into the design part. | Onur | Davin | 4 |
| 3 | √ | Time tag functionalities | Design | Write down what comes in your way and tell the group asap, the group will try remove the barriers, if not, we will record it into sprint meeting. The difficulties you face will goes into the design part. | Onur | Davin | 3 |

This shape represents a table slicer. Table slicers a supported in Excel or late

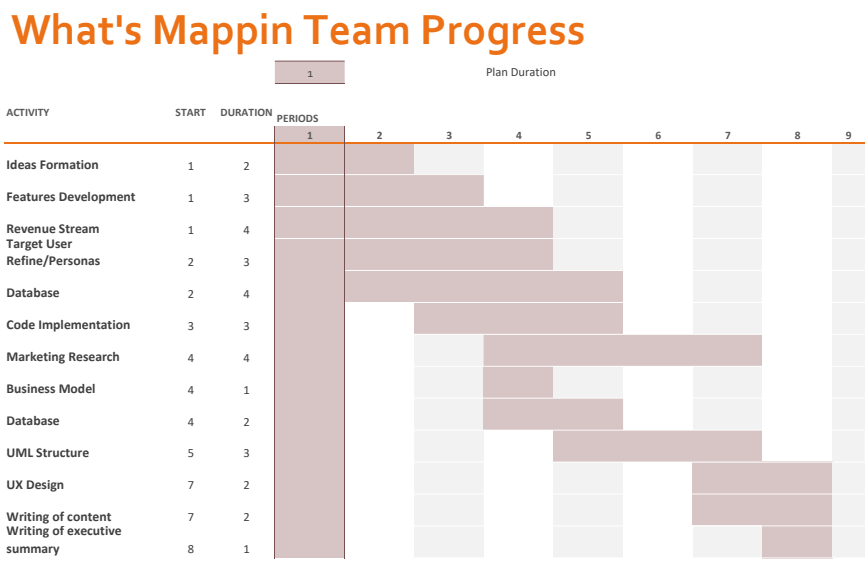
If the shape was modifiec an earlier version of Excel or if the workbook was saved in Excel 2007 or earlier, the slicer can't be used.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 3 | √ | Factbook and Instagram API | Design | Write down what comes in your way and tell the group asap, the group will try remove the barriers, if not, we will record it into sprint meeting. The difficulties you face will goes into the design part. | Onur | Davin | 3 |
| 3 | √ | Definition of app | Business Case | Write with an aim for submission ready paragraph: Problem, Solution, Customer, Priority and competitive advantage | Teddy | | 2 |
| 3 | √ | Business Case | Business Case | Industry and market feasibility analysis, revenue strategy and go-to-market Strategy<br>Use the business model canvas, analysation framework such as Porter five forges. I think the justification you write on last week is very good, let's also put down the professional's idea, why it will not work for us and why this app will be better. | Teddy | Teresa | 5 |
| 3 | | Logo Making | Design | You can draw one. If not, specify what would it look like, write in meeting backlog | Teddy | John | 3 |
| 3 | | Burndown Chart | Scrum Management | Visually track the productivity | Yanting | | |
| 3 | √ | Product Backlog 2.0 | Scrum Management | Updated a Backlog that is usable across the group and ready to show to the Lecturer | Yanting | | 3 |
| 3 | √ | Requirement | Design | Serve as the ultimate guide for Design and Business team | Yanting | | 3 |
| 3 | √ | Report Structure | Scrum Management | What are the sections and sub sections in the report | Yanting | | 4 |
| 3 | √ | Database | Design | | Davin | Onur | 4 |
| 4 | √ | User Personas | Analysing | Why would the user use our app, which scenario they would think to use it, why there /'s therefore a demand/market for our app/pictures of the people | Teresa | | 3 |
| 4 | √ | Textual use-case | Design | | Onur | | 4 |
| 3 | | UML Diagram | Design | Use IDE to generate a Class Diagram | Onur | | |
| 3 | | Competitor paragraph | Business Case | | John | | |
| 3 | | Design class diagram | Design | | Davin | Onur | 4 |
| 3 | | Analysis class Diagram | Analysis | | Davin | Onur | 4 |
| 4 | √ | User Interface 2.0 | Design | | Yanting | | 3 |

The group have attached pictures of our group sprint plans and reviews which we conducted all together at a café in Green Park 'Joe and the Juice'. The coffee shop with a silent room and chalk board at the back became an upholding of our group's culture and great working dynamic!

During our project timeline, one of our team members became an aunt and the baby (see above) became the group mascot.

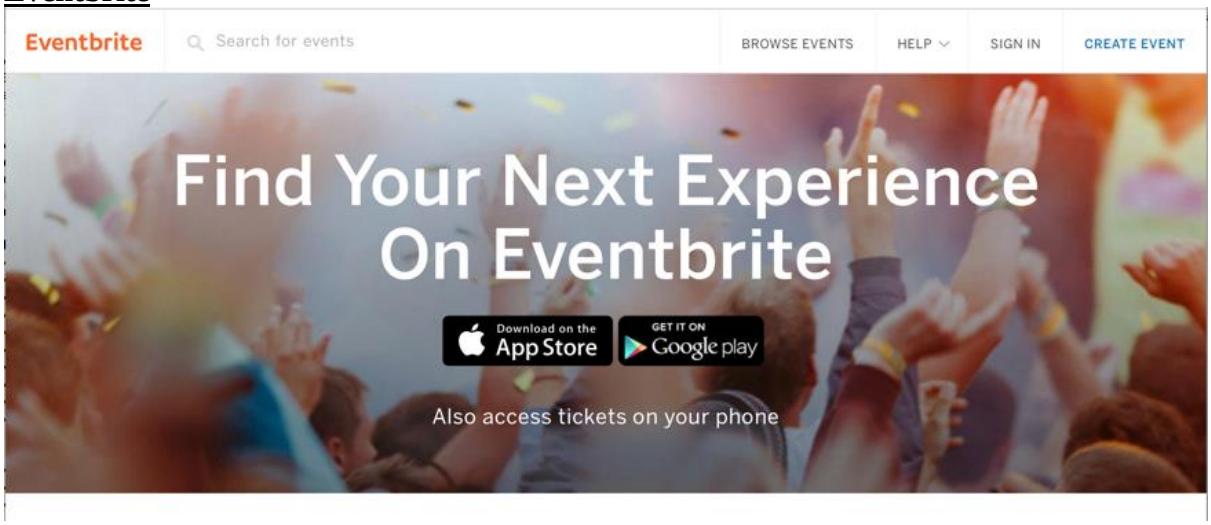
In terms of our broader timeline, we had set ourselves a waterfall timeline to show where our minimum progress should be at any given point (below) but since we were working in Scrum iterations we never fell behind on these deadlines and comfortably met our project deadline.

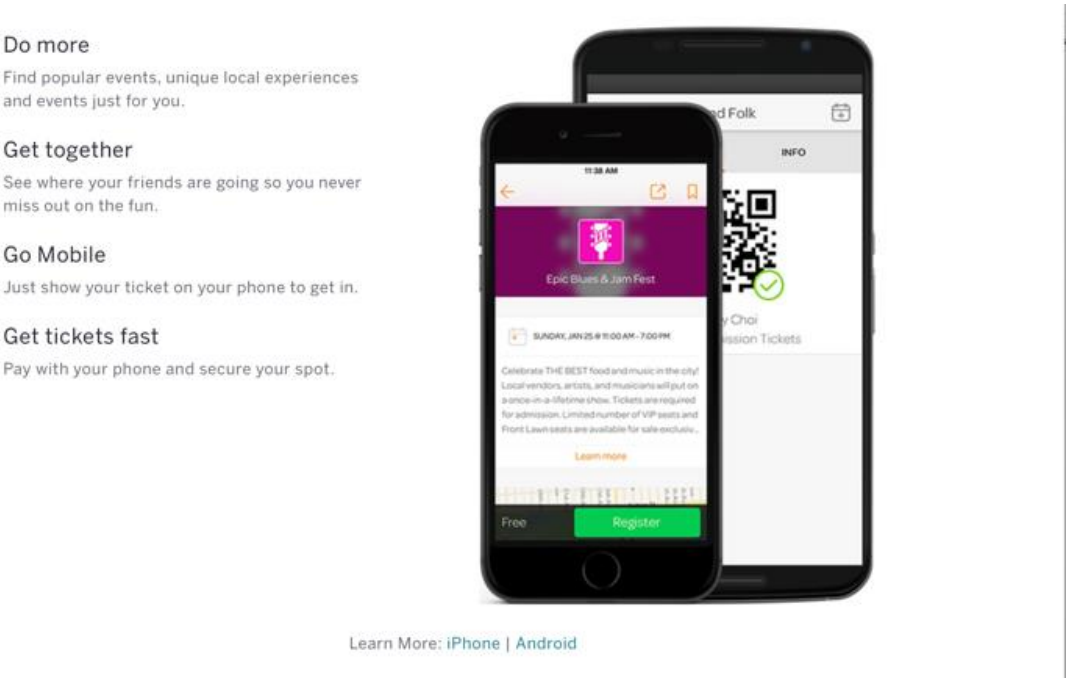



# Competitor Analysis

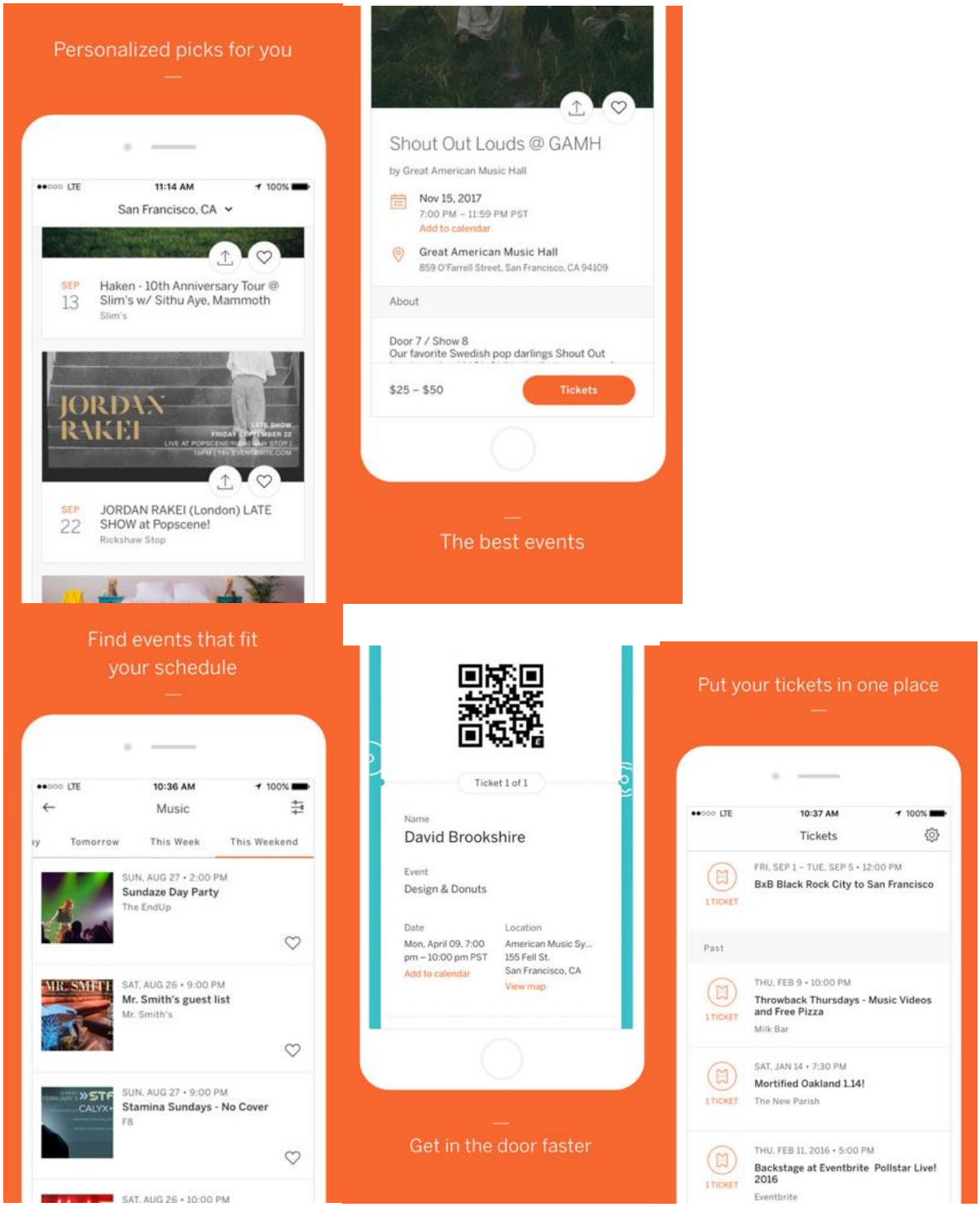
By completing a competitor analysis on some of the following events platforms/apps it is possible to identify what makes our product, an event mapping app, unique and can demonstrate the attributes we wish to apply in order to target our target market; students.

**Eventbrite**






Eventbrite is an event management platform for event seekers and event organizers. The Eventbrite app is an app enabling anyone to create, promote and sell tickets to any event whilst also helping people discover and share events they like. It allows a user to discover events near them, by selecting their location, though a “discover” tab where recommended and popular events in the area appear. Also, a user can connect to Facebook to see what events their friends are attending. Users can browse by category, keywords and location as well as being able to filter their results by relevance, date, distance and price.
To purchase tickets users can simply register or “get tickets”, where they can add tickets to a “cart”. After, a continue button is provided where they must confirm their personal information and enter the payment information which is then finished by a “complete order” button. The user will then receive an email with their order confirmation.
Users can find tickets in the Eventbrite app through a “Me” tab where they can view details about all events they’ve registered for, get in touch with their event organizer or show their ticket to be scanned at event entry. Also, there is an option to add the ticket to Apple wallet/passbook.
Another feature that Eventbrite offers users is the saving and sharing of events. To save events, users must tap a bookmark icon and can find them in the “Me” tab by tapping on the “Saved” icon. To share events, users need to tap the arrow icon on the “Discover” tab and these can be shared with a new text message, email, a link to the event, twitter, Facebook or WhatsApp. Also, if the users connects to Facebook they will receive notifications notifying them when friends of them are going to the same events (Antwonne D, 2017).



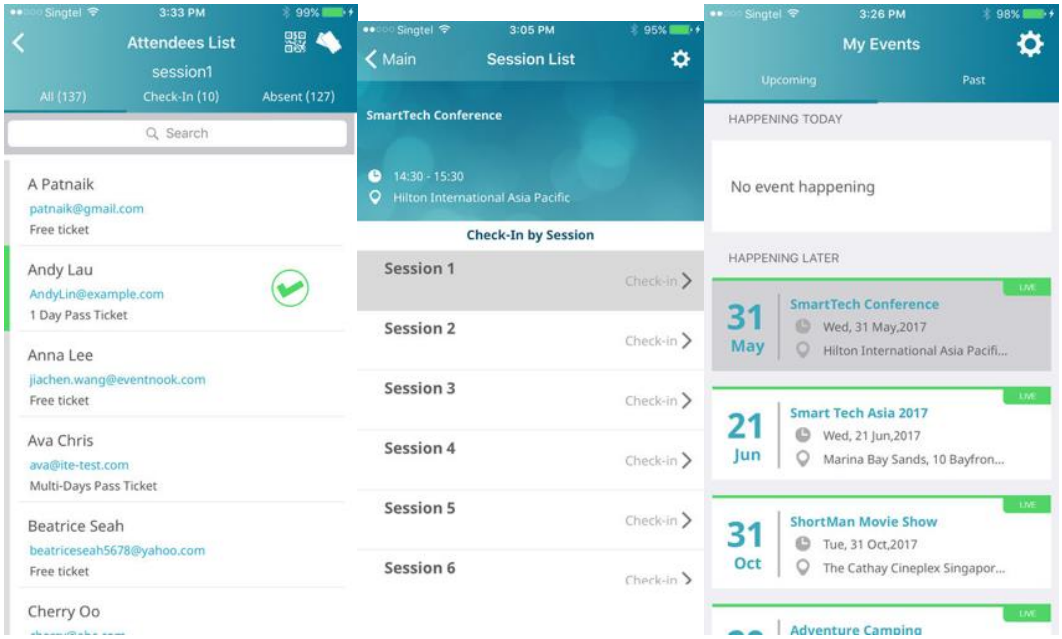

Currently, on the app store, Eventbrite is number 33 in Entertainment and rated 4+/5 with a total of 178 ratings. It is available in English, German, Spanish, French, Italian, Dutch, Portuguese and Swedish. In terms of event organizers, Eventbrite has a separate app called Eventbrite Organiser allowing these to sell tickets and manage guests and entry. (Eventbrite,2016)

Despite its excellent user interface, it has been found to have a couple of poor ratings on the app store. For example, a user rated EventBrite 1/5 stars and below the title "Worst ordering app/website" left the following comment "I ordered tickets through this website today and

it's terrible. First of all it timed out after asking for unnecessary information on every person attending the event. I never received all the tickets. Their "how can we help section" was not helpful at all. I called their customer service whose automated service told me to reply to the order email. I replied to the order email and received an automated email to say the mailbox is not in use. Tried contacting through Facebook, no response yet. Enough to test anyone's patience, I've been trying to sort this for 2 hours!!".

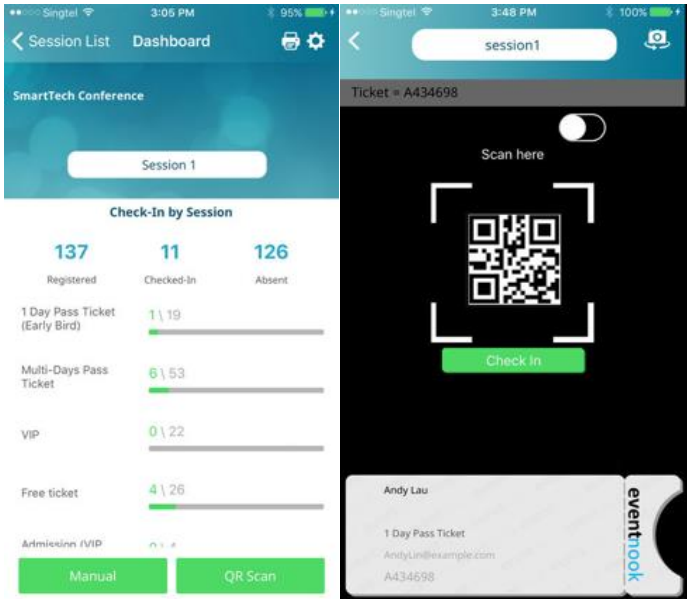
**WhatsMappin offers mapping features of events, clearly showing events on a map to guide users, promotions and free drinks, which EventBrite does not.**

**<u>EventNook</u>**

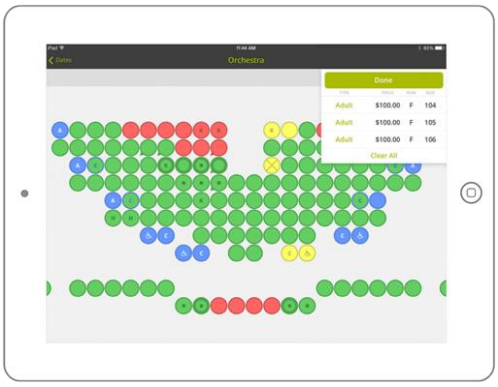


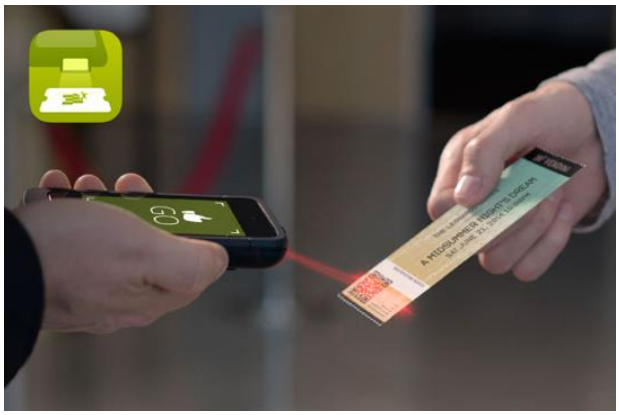

EventNook is an Asian leading event management software company offering complete end-to-end online event registration and ticketing solutions through an online source as well as through the app EventNook CheckIn Pro. EventNook is a platform/app only for event organizers so does not apply to event attendees like WhatsMappin. Features provided by EventNook include mobile friendly event registration, customisation of registration form and detail collection, easy online payment, discount & promotions, email customization, QR code ticketing, auto email confirmation, management of registration and sales in real time, analytics and advanced marketing tracking and auto-generated invoicing (EventNook,2017).

In terms of the app, it is free and contains all these features but specified on the app store EventNook Check-in Pro allows quick attendee check, QR-code scanning e-tickets and instant ticket verification, break-out session attendance tracking, custom designed badge printing, searching registered attendees by name and email and viewing what's happening with the attendance on the go (EventNook,2017).



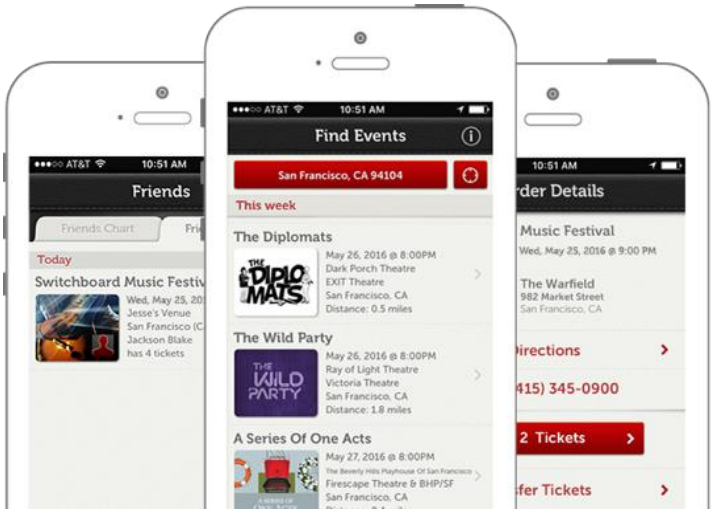

**WhatsMappin offers features for event seekers/ateendees which EventNook does not, as it focuses simply on event organizers.**





## Vendini

Vendini is a company that "provides dependable and easy-to-use ticketing, marketing, fundraising and patron management solutions to any-sized organization." It also offers an app called Vendini TicketScan with 5 different features including sales spotlight which allows access and sharing of live sales and inventory data; patron connection where users can meet their personal day-of-show assistant; TicketAgent providing a convenient box office solution; TicketScan allowing ticket scan and control of access to events and finally Walletini which allows to find, purchase, store and transfer events and tickets. Only the last 3 main features are relevant to our app as the others are not focused on events (Vendini,2001).
On the app store, there is limited information about the app with only one image of its user interface and not enough reviews to display a summary. However, it is rated 4+. overall (Vendini, 2013).

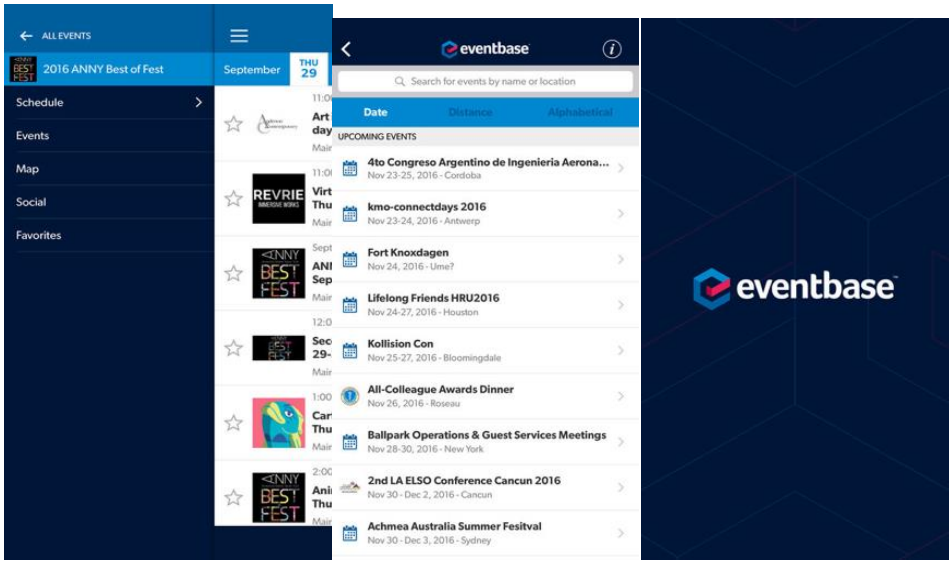


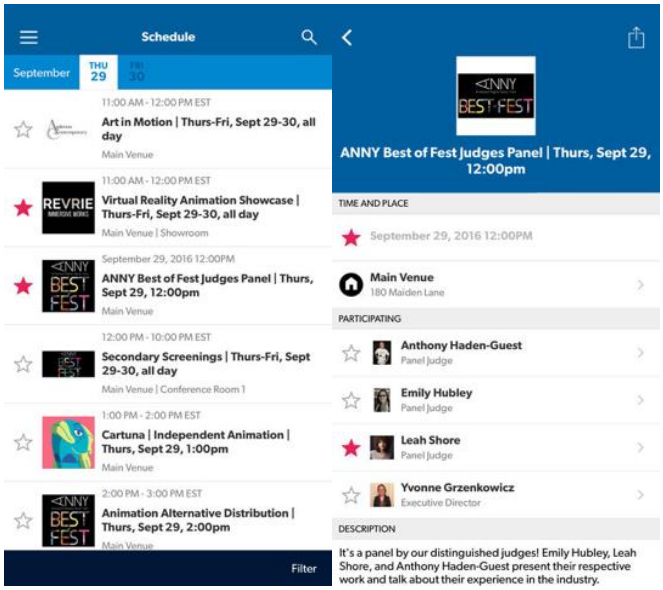

**WhatsMappin offers last minute ticket proposals and mapping features which Vendini does not, increasing last-minute sales for event organizers.**


## EventBase

EventBase is a mobile event technology platform targeting global enterprises with current customers such as IBM, Cisco, Microsoft, SAP etc. They have 7 different app products: Enterprise Event apps, Corporate Meeting apps, Conference app, trade show app, festival app, sports events app and Eventbase Free (Eventbase, 2017).

Eventbase is a free app which is in most relevant direct competition with WhatsMappin as it's claimed to be "the world's most powerful event app platform". It provides information on conferences, music and film festivals, trade shows and fairs, sporting and community events and many other events. This app is for event seekers only and provides the following features; searching for any event by category, downloading event schedule to be available offline, receiving updates to the schedule from event organizers, building own personalized schedule, find out who is at the event, viewing of maps with venue locations, sharing of experience on Facebook, Twitter etc and searching for sessions, participants and venues. There is not enough customer ratings to display summary on the app store (Eventbase,2016).



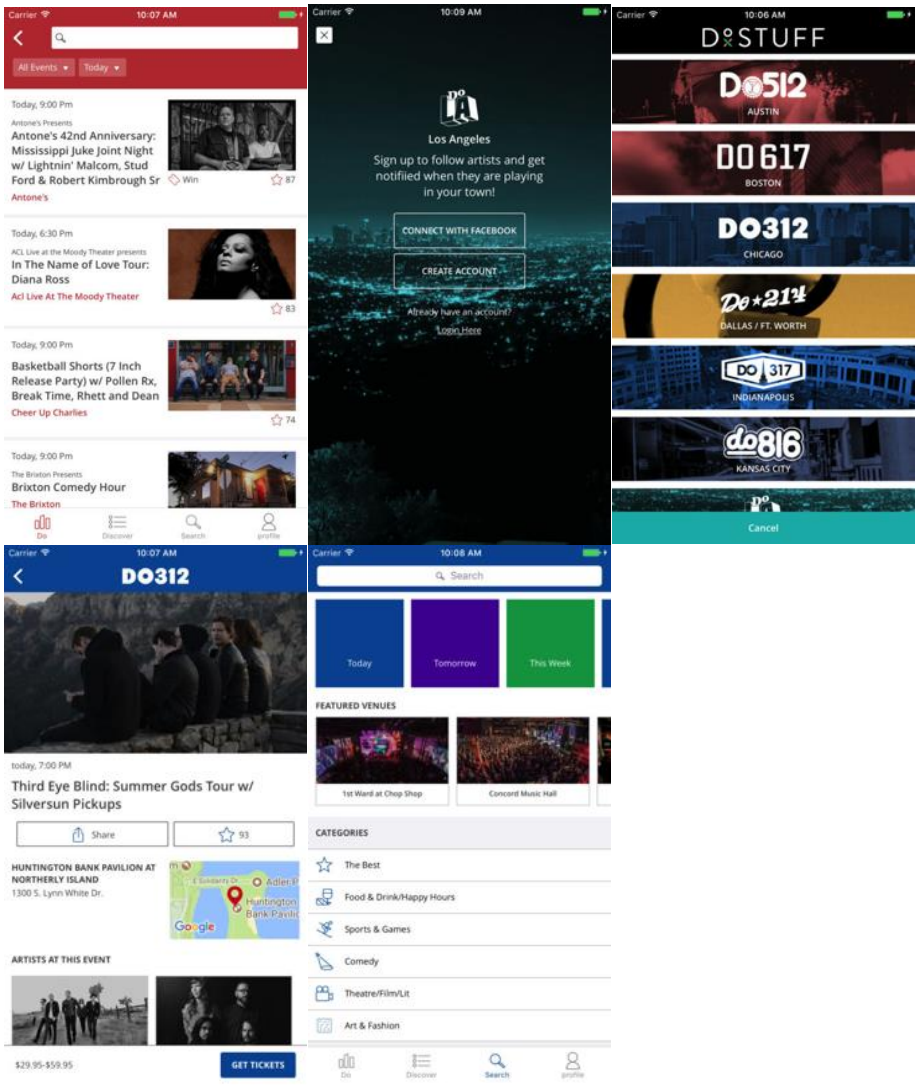




**WhatsMappin offer promotion for event organizers which EventBase does not, as it solely focuses on event seekers**.

## DoStuff

DoStuff is a local media properties network run by local teams in 20 cities, mainly in the United States, answering the question “What am I going to do tonight?”. The app provided on the App Store, and also available for Android devices, is called “DoStuff – What To Do Tonight”, is free and allows users to find out what is happening in the current city they are in (DoStuff,2017). Location-based search and redesigned profile pages are two features that this app provides. Also, users can browse events by category and locations in the “Do” tab whilst in the “Discover” tab users can find curated event listings and articles created by the network’s team. Users can share everything they wish through a share button and can add any event in the app to their calendar through clicking a plus sign on the event card. If users wish to be notified on events of artists they like they can click the plus sign next to their name and

will be automatically notified when these exist in their current city. To change city, users have to click on the city logo to go back to the selector. Currently, the DoStuff app contains 564 ratings on the app store but no overall rating (DoStuff,2017)

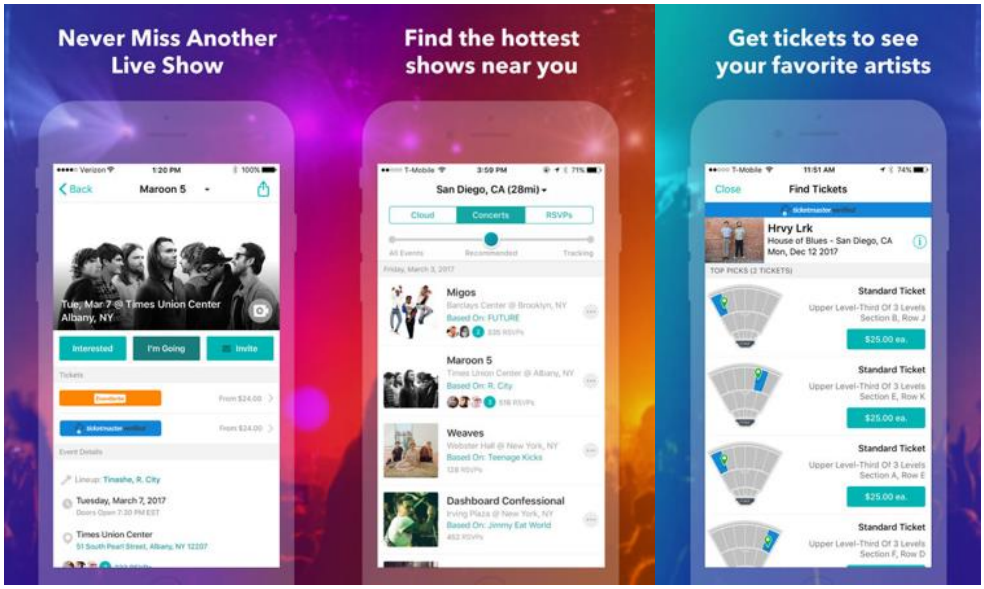


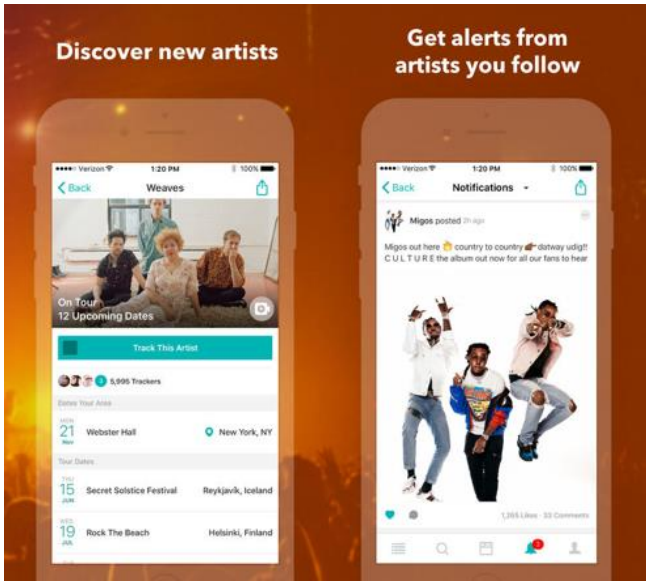

**WhatsMappin offers mapping features of events which DoStuff does not, clearly showing events on a map to guide users.**

## Bandsintown

Bandsintown is a concert discovery platform where fans can connect with their favourite artists. The app exists both on the Apple Store and for Android devices and allows users to search for concerts, shows, festivals etc by browsing through location or artist. After doing so, it displays the event details as well as the links to where to find tickets. Users can also track artists by pressing “Track This `artist” which will then notify them about upcoming events which they will perform at.
Currently, on the app store, it contains 3320 ratings, is rated overall 4.6 and is number 60 in

music. It has many positive reviews such as “London has many venues with many great bands performing every day, not perfect but it’s ability to capture and tell you what’s going on is what this AP is about!” (Bandsintown, 2017).



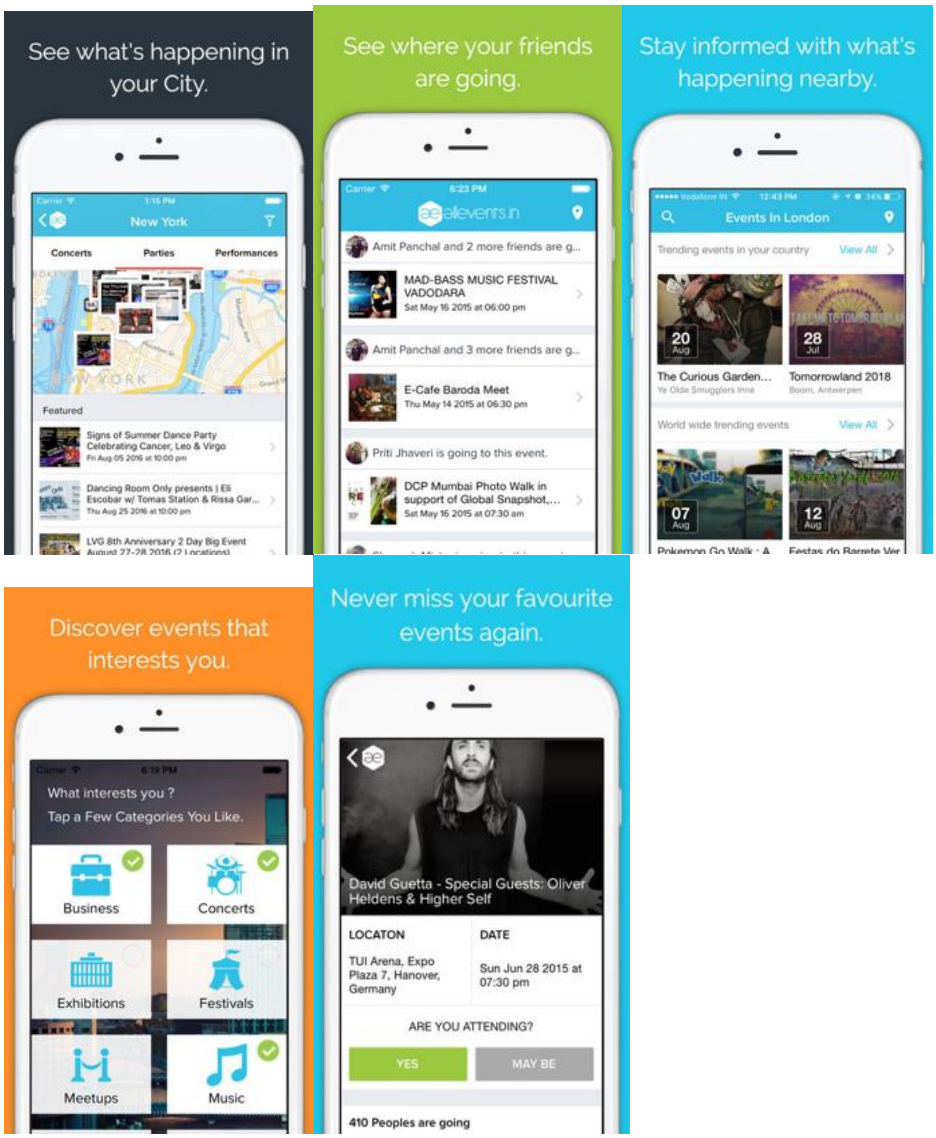




**WhatsMappin offers mapping features of events which Bandsintown does not, clearly showing events on a map to guide users.**

## All Events in City

All Events in City is an event aggregation platform to help people discover events happening around them, providing both an Android and Apple app (All Events in the City, 2017).The app does not require the user to sign up and allows them to browse events through categories e.g. parties, concerts, seminars etc, search events by location and mapping them on google, adding events to calendar and filtering events by date or categories. However, the user can

sign up and if he/she does so then he/she will be provided with additional features such as being able to collaborate with friends, finding events recommended for them, keeping track of events they have attended, know which events their friends are going to and follow their favourite event organizers to get notifications on these events. Moreover, this app is not only for event seekers but also for event organizers as these can use the app to promote the event. Currently, on the app store, it contains 59 ratings but has no overall rating (Amitech, 2017).




**WhatsMappin offers last minute tickets, money off and free drink proposals which All Events in City does not, captivating**

**<u>What differentiates WhatsMappin from its competitors?</u>**

By analysing WhatsMappin's competitors we can establish what features we can offer that our competitors have not implemented. None of these apps seems to have a feature of last minute tickets which our app will have an emphasize greatly as our target market, students, tend to purchase tickets "last minute" and would greatly value a platform that allows this process to be stress-free and easy. Also, WhatsMappin will constantly emphasize money off and free drinks on the relevant events it promotes to capture a greater share of the target

market. Furthermore, an important feature which is not apparent of any of these competing apps is a disability function, which our app will implement.
However, we still haven't implemented a tracking feature as many of our competitors do so that users can track artists, events or venues they specifically like. Also, we do not have an option to add the events to a calendar which would be helpful for students who keep a busy life where organisation is key.

# Marketing Strategy

In order to devise the most adequate marketing strategy for WhatsMappin we must perform an analysis of the marketing mix's four P's: Product, Price, Place, Promotion

**Product**

WhatsMappin is an event-listing platform/app that maps events geographically, allowing users to view events by location and availability. Our app provides a convenient solution for event seekers, looking for last minute ticket opportunities, and event owners, looking to fill up their event's. We target a student consumer group, which is very price sensitive and influenced by the behavior of others hence the word of mouth community is a key factor in generating success for our app.

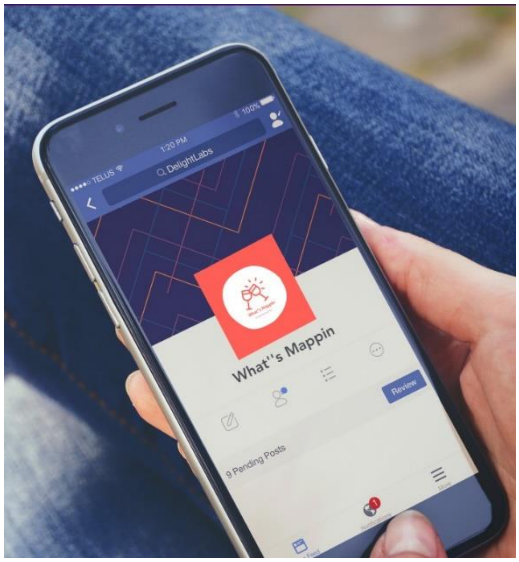

Our value proposition consists of offering the benefits of convenience, discovery and entertainment all in one app. This is possible due to the implemented features of filtering of events based on location and price as well as being search events which relate to users' favourite clubs or artists.

Moreover, our app maps the events so a user can more clearly see how to reach it depending on where he/she will be going from. Last minute tickets, money off and free drinks are features users can use in the app which attract more users to events.

**Price**

Currently, our app is free thus we do not have a specific pricing strategy set. However, in terms of making profit we do set a pricing strategy based on a referral scheme and ticket sales commissions, as emphasized in the business plan and monetization.

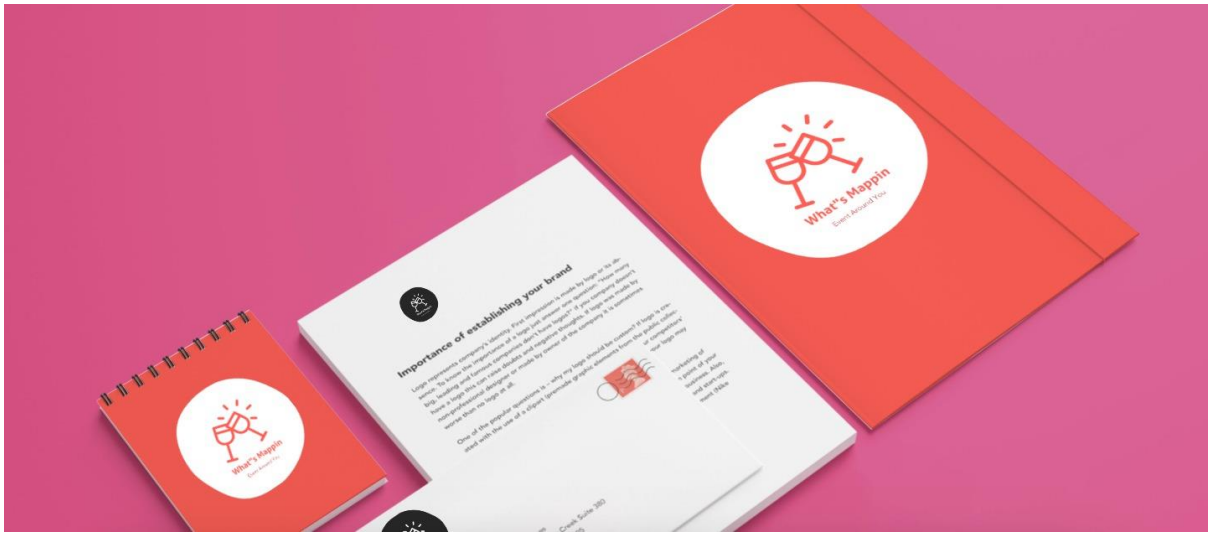
The profit made from these two strategies will allow further investment for the growth of the app and its exposure. For example, we will invest in producing leaflets to hand out to students at universities as well as advertisement on social media platforms such as Facebook and Twitter. Customer acquisition cost is key as it can be noted that a large sum of investment is used for it to increase the number of users using our app.




**Place**

In terms of place, our app will be available on the android store.

**Promotion**




The promotional strategy is specifically important in the early stages of the app, since we need to establish a positive word-of-mouth. Deliveroo, UberEats and Tinder are all great examples to follow in terms of their promotional strategy due to their established world-wide exposure, made possible through investing hugely into the promotion section of their marketing strategy. Advertising on social media as well as inside universities will cover a great proportion of our investment into the promotional strategy, as a positive word of mouth will be needed straight from the beginning.

Enticing first-time customers with generous discounts off their first orders, like Deliveroo, is a strategy WhatsMappin will adopt in order to increase its user base. Also, users will be able to insert promotion codes enticing them to use the app. Deliveroo used stunt marketing in the early stages of its lifecycle through, for example, staff in Deliveroo Singapore every Friday putting on kangaroo onesies and heading to the streets of the main business district to hand out flyers, notebooks and other merchandise, since they knew a high proportion of their target users (people who work at offices and order lunch) would be there. This is a strategy we have in mind by aiming at our target audience, students, when going to Universities and

performing a similar activity whilst giving out flyers and promo codes. Also, as seen by Deliveroo, influencers and bloggers promoting the app is a huge step in establishing brand presence. Therefore, we will contact these respectively to maximize exposure on social media (Lee, 2016).


Ubereats initially established a test market for the app in Toronto before rolling out into other states followed by other countries. This strategy allowed them to monitor feedback effectively and react appropriately, fixing any evident problems and offering quick solutions to customers' problems. In our promotional strategy, we will implement a similar concept by firstly only concentrating on the central London area, targeting university campuses, to gain initial insights and feedback, and after expanding into other locations whilst we optimize our app. Another important lesson learnt by UbereEats is the importance of listening to customers and adapting appropriately. UberEats interviewed and handed out surveys to customers which provided them with feedback on customers wanting more restaurant options and better food, which they used to optimize the app. Interviews and surveys are essential to optimize our app as we will confront customers on what they wish to see on the app, that isn't available, as well as any disappointing features they wish to be improved (McClelland, 2016).

WhatsMappin will host launch events at universities around London where entrance and free drinks will only be permitted if attendees show proof of having the app on their mobile devices. This strategy was implemented by the successful dating app Tinder and greatly impacted on the number of users downloading the app. Also, we will carry out personal outreaches through individually approaching students at universities and campus events to get at least 10 people to download the app, by handing them a free drink (Hackett, 2015).

# Data Selection and optimisation

A crucial element of What's Mappin lies in custom tailoring the in-app experience according to the personality and needs of the app user in such a way to provide them with a streamlined and non-intrusive experience. Keeping information relevant to the user and not overloading individuals with suggestions and recommendations that have nothing to do with them will be vital in increasing user retention rates, attracting new users, and increasing the overall satisfaction of the app.

In this section, we'll explore what data we'll be collecting from users, how and why, and additionally the procedures we'll be using the collect and analyse that data.

Data collection procedure:

There are two forms of data our app will collect. Registration data (data gained when the user creates an account) and clickstream data (data obtained based on user interaction with the app).

To quickly collect as much useful data as possible from users, we plan to use social logins within our app using the Auth0 API. The Auth0 API will allow users to quickly login from their favourite social media websites like Facebook, Google or Twitter etc. Not only would this grant us access to up-to-date profiles immediately, it'll also allow users to login quickly and this would also increase registration rates.

On the interface side, all the user would have to do is enter the app and click their preferred social media platform. A login request is then sent to the platform where the user's identity is confirmed, upon which they are registered and logged in. This method also means that we don't need to implement specific APIs for each social media platform or use their methods to request certain bits of data as the data we need to collect can be chosen beforehand on Auth0's website. However, the final finished product will still need code for call-back URLs, set credentials, hosted login pages with instances and methods for obtaining and holding user credentials, session handling, etc.

A. Basic user data – attributes/demographics (Explicit data gained in registration process)

(Number in brackets is priority for that piece of data. High numbers mean more relevancy)

1. No names (due to privacy concerns, we can't take their actual names, but we can take user names, and we can store their profiles in the database with ids' like Id1674363).

2. Email – Needed for registration confirmation, updating the user on new events, password resets, etc (4)

3. Age/Birthday – Can act as a minimum age requirement barrier to displaying certain events, can be used as part of a decision-making algorithm for event recommendations that are dominated by a certain age group or for recommending friends who are at the same age group. (3)

4. Education history – Same as age but with education history (1)

5. Friends – Can take list of friends out of Facebook and apply it to the users existing account on our app. (2)

6. Hometown – Useful for marketing and researching demographics, as well as friend recommendations. (1)

7. Current household location – Can be used to suggest events and friends from nearby vicinity. (3)

8. Relationship status – If single, can suggest events where opposite gender tends to be slightly more frequent. Extremely useful for the pickup community (3)9. Work history – Same as education history (1)

10. Gender – linked with relationship status. Some events are also gender sensitive, like gay night clubs. (3)

11. Ethnicity/nationality – Same as hometown. (2)

12. Average Income – Can be used for matching users to events they can afford (1)

13. Hobbies – Same as age/birthday.

Clickstream data is data gained from the process of the user interacting with the app. It keeps track of the behaviour and preferences, what type of events they like, how often they go to events, the time between usage, etc. This information can then be used to create consumer models for predicting which type of events (or just a certain event) are going to be sold out, what types of user usually dominate those events, and so. This data can not only be used to give better recommendations that adapt to the user's tastes, but also used for marketing and developing new business strategies around targeting a specific demographic, selling the data to research companies, or offering brand new services that provide consulting to event makers.

To collect clickstream data, we simply add counters variables to a user ID's column in the database and increment them whenever the behaviour behind that action is reinforced.

B. Advanced user data – Activity/Preferences (Clickstream data – gained in the process of interacting with the app)

1. Most viewed events – to understand user preferences and update their event recommendations (3)

2. Track user location – same as household location (3)

3. Logins per day – used to gauge whether individual is a light or heavy user and identify peak times for app usage (2)

4. Time spent on app – same as logins per day (2)

5. Time between events participated – Used to identify peaks, slumps, and overall trend of user activity. Can be used to discover users who are falling away from the app. (2)

6. Number of events participated by type – useful for event recommendations (2)

7. Opinion of events – judged by likes, or by similar mechanics (2)

8. Last sign in date – same as logins per day (2)

In addition to all the above data, we also plan to collect data on the user's device to help with troubleshooting any future bugs that might have not been caught during the testing phases of the app.

C. Debugging data

1. Device info – OS version, phone model

2. Network conditions – IP address, Mobile network, unique device identifiers.

How we plan to store the data in the long term and short term:

A short-term affordable solution which we can use now is to the put the data in a simple spreadsheet (or access which is an actual database software). We decided that the best way to go about this would be to use Google spreadsheets (for now) as a platform for storing user data. Since the tool is free and available online, there are no costs involved in establishing this as a database and the database will be available 24/7 online, so no need to host one in our own databases.

Somewhat detailed description of getting the data from the app over to google spreadsheet:

1. Create the google spreadsheet as a form.

2. Build app and store user data in application data folder in a file as a JSON object.

3. Get input hook from spreadsheet.

4. Integrate input hook into existing code using Retrofit API to transfer user data into app.

Once we begin acquiring many users (like 100,000), we plan to invest into an actual database and use a typical 3-tier architecture for storing data:

The client/presentation tier represents our android app (At the moment, a mobile client will be our interface with the user). This is the area where users interact with the app and data is first collected. The logic/application tier lies in the app and is responsible for transferring collected user data over to the database and receiving results from database queries on lists of events, etc. As for our data tier, we plan to use Oracle DB as our database technology for storing and retrieving user data. Oracle DB is currently one of the best and most dependable database platforms on the market supports android app development and testing (regarding the database). Oracle DB has a strong and robust synchronization engine capable of connecting many physically remote DBs synchronized to main backend Oracle DB system. Encryption for data transport and storage is also supported.

How we could use the data collected (I don't really have much info here):

In-app usage:

1. Event suggestions

2. Friend suggestions

Program on server shifts through data, makes a list of recommendations based on criteria/algorithms. These lists are send to the Users phone in app data folder, where the app reads the list and displays it as recommendations.

Marketing/Business usage:

1. Tracking user trends – popular events, etc

2. Tracking usage

3. Segmenting users and targeting specific segments and sub segments

# Requirements

## MOSCOW Analysis

### Requirement

| ID | Details | Priority |
|---|---|---|
| | **ACCOUNT** | |
| AC1 | The application shall allow users to login using Email | Must |
| AC2 | The application shall allow as guest | Must |
| AC3 | The application shall allow user to logout | Must |
| AC4 | The application shall allow user to view their wishlist | Should |
| AC5 | The application shall allow user to change name through setting | Should |
| AC6 | The application shall allow user to type in their promotion | Should |
| AC7 | The application shall allow user to login using Facebook | Could |
| AC8 | The application shall allow user to login using Instagram | Could |
| AC9 | The application shall allow user to save the event to wishlist | Could |
| AC10 | The application shall allow user to check their past event | Could |
| AC11 | The application shall allow to check existing promotion | Could |
| AC12 | The application shall allow user to share the events to friends | Would |
| | **PERFORMANCE** | |
| PF1 | The application shall allow user to login within 5 seconds | Must |
| PF2 | The application shall be available 24/7 365 a year | Must |
| PF3 | The user shall be able to click on the event and see rout to get to the event | Must |
| PF4 | The application are able to connect to internet | Must |
| PF5 | The application shall allow user to disability to use accessibility function | Should |
| | **Database and Data Analytics** | |
| DA1 | The application shall gather user name and status from Facebook Profile | Must |
| DA2 | The application shall gather real-time user location data | Must |
| DA3 | The application shall be able to store 5000 events | Must |
| DA4 | The application shall gather user behaviour of using the app including browsing, clic | Should |
| DA5 | The application shall optimise targeted event marketing stratgy based on data | Should |
| DA6 | The application shall automately generate a referral bill base on user click | Should |
| DA7 | The application shall gather user's listed interest | Should |
| DA8 | The application shall gather user followers number to determine level of promotior | Could |
| DA9 | The application shall gather user's followers and likes from instagram and Facebook | Would |
| DA10 | The user with higher likes will reveive more promotion | Would |
| DA11 | The database is log onto CRM platform | Would |
| | **Event Recommendation and Filter** | |
| ER1 | The application shall display the cost of the event | Must |
| ER2 | The application shall display event type | Must |
| ER3 | The application shall display event today on default | Must |
| ER4 | The application shall display the events in location mark | Must |
| ER5 | The application shall allow user to filter base on price | Must |
| ER6 | The application shall allow user to filter base on time | Must |
| ER7 | The application shall allow user to type in location | Must |
| ER8 | The application shall display events < 1km reletave users location on default | Should |
| ER9 | The application shall display extra freebies and discount for each event | Should |
| ER10 | The application shall display events as user move the map | Could |

# User profiles

| |  |  |  |
|---|---|---|---|
| | Katy Smith | Marta Jhonson | Alberto Marino |
| Age | 18 | 20 | 25 |
| Nationality | British | African-American | Italian |
| Location | London | London | Rome, Italy |
| Level of Education | High school<br>Currently on her first year of Bachelors degree | High School<br><br>Year abroad | Bachelors degree<br>Is going to start his Masters degree in the UK next year |
| Why do they demand the app? | Katy is a student in London looking for the best student nights to attend at the best price she can possibly get.<br><br>Looks for convenience; Wants to be able to search based on location and price.<br><br>Always makes last minute decisions | John organizes events for "insert London Club name" and is looking for a better platform to complement their Facebook marketing strategy so as to increase advertising last minute. Looking for event exposure | As a foreign student, Alberto knows little about exciting events happening in the UK and wishes to learn more before he moves to the UK.<br><br>Wants to know information about what is expected at each event to be better prepared. |

# Host Profiles

| |  |  |  |
|---|---|---|---|
| | John Williams | Joanna Miller | Cristopher Fleming |
| Age | Male | Female | Male |
| Nationality | 30 | 40 | 55 |
| Location | British<br>London | British<br>London - Camden | British<br>London |
| Level of Education | Bachelor degree and Masters degree | High School | Bachelor degree |
| | Now working as an event organizer for a London club | Owns a bar in Camden | Pub owner – live music evenings |
| Why do they demand the app? | John organizes events for XOYO and is looking for a better platform to complement their Facebook marketing strategy so as to increase advertising last minute.<br><br>Looking for event exposure | There is difficult competition in Camden due to the great number of bars and clubs in the area. Joanna wants to find a platform where she can bring exposure to her bar through advertising and giving out promotions to attract | Cristopher wants to attract more customers to the weekly live music events he hosts at his pub. |

## Textual Use Case Tables

| | |
|---|---|
| **USE CASE** | Establishing internet connection |
| **ID** | UC 1 |
| **BRIEF DESCRIPTION** | User must establish internet connection before opening the application. |
| **PRIMARY ACTORS** | User |
| **SECONDARY ACTORS** | - |
| **PRECONDITIONS** | - |
| **MAIN FLOW** | 1- User must be connected to internet via Wi-Fi or cellular. (Using Wi-Fi will give more accurate results in terms of plotting the location.) |
| **POST CONDITIONS** | User established internet connection |
| **ALTERNATIVE FLOWS** | - |
| **USE CASE** | Login |
| **ID** | UC 2 |
| **BRIEF DESCRIPTION** | User attempts to log into the system using one of the given alternatives (Facebook, Google+ Instagram). |
| **PRIMARY ACTORS** | User, Facebook, Google, Instagram |
| **SECONDARY ACTORS** | None |
| **PRECONDITIONS** | User must establish internet connection |

| | |
|---|---|
| **MAIN FLOW** | 1 User sees the login screen.<br>2- User gets to choose from three different login options.<br>3- Connection with the relevant API is made.<br>4- API checks for account information. |
| **POST CONDITIONS** | User is logged in. |
| **ALTERNATIVE FLOWS** | Invalid Login |

| | |
|---|---|
| **USE CASE** | Invalid Login |
| **ID** | UC 2.1 |
| **BRIEF DESCRIPTION** | Login attempt is failed due to missing account information. |
| **PRIMARY ACTORS** | Facebook, Google, Instagram |
| **SECONDARY ACTORS** | None |
| **PRECONDITIONS** | User tried to log in with a selection which does not hold the account for the user. |
| **MAIN FLOW** | 1- User is informed about the failed attempt and is asked to try again.<br>2- The welcome screen is displayed again, where the user gets another choice.<br>3- Following steps are identical with UC1. |
| **POST CONDITIONS** | Failed login |
| **ALTERNATIVE FLOWS** | - |
| **USE CASE** | Plotting Map Data |
| **ID** | UC 3 |
| **BRIEF DESCRIPTION** | Google Maps API plots the current location of the user on the map. |
| **PRIMARY ACTORS** | User, Google Maps |
| **SECONDARY ACTORS** | Event providers |
| **PRECONDITIONS** | User must be logged in with an account. |
| **MAIN FLOW** | 1- Current location of the user is plotted on the data along with the regular Google Maps data. |
| **POST CONDITIONS** | Current location is plotted on map. |
| **ALTERNATIVE FLOWS** | User's last known location is plotted. |

| | |
|---|---|
| **USE CASE** | Plotting Last Known Location |
| **ID** | UC 3.1 |
| **BRIEF DESCRIPTION** | If there is no internet connection at the given moment, the last known location of the user is saved as an instance to be used for later. |
| **PRIMARY ACTORS** | Google Maps API |
| **SECONDARY ACTORS** | - |
| **PRECONDITIONS** | User must have a known past location. |
| **MAIN FLOW** | 1- The last saved instance of the user is called by the code to be plotted on the map. |
| **POST CONDITIONS** | Last known location is plotted on the map. |
| **ALTERNATIVE FLOWS** | - |
| **USE CASE** | Plotting Events Data |
| **ID** | UC 4 |
| **BRIEF DESCRIPTION** | A request is made to the events API which responds in JSON format. |
| **PRIMARY ACTORS** | User, Events API, Google Maps API |
| **SECONDARY ACTORS** | Event holders, event location |
| **PRECONDITIONS** | There must be events present in the vicinity of the user. |
| **MAIN FLOW** | 1- A HTTPS connection is made with the events API, using correct parameters for the URL.<br>2- A JSON response is captured by the application.<br>3- The JSON response is parsed into Java objects.<br>4- Java code implementation, working with Google Maps, plots the events around the user given their longitudes and latitudes. |

| | |
|---|---|
| **POST CONDITIONS** | Events around the user are plotted. |
| **ALTERNATIVE FLOWS** | No events are present. |

| | |
|---|---|
| **USE CASE** | Receiving event description |
| **ID** | UC 4.1 |
| **BRIEF DESCRIPTION** | All the available description regarding the event is requested from the event API. |
| **PRIMARY ACTORS** | Events API, Google Maps API |
| **SECONDARY ACTORS** | Event providers |
| **PRECONDITIONS** | Event description must be present in the events API. |
| **MAIN FLOW** | 1- A request is made again to the events API regarding the description.<br>2- The response is received by Java code to be parsed into object format from JSON.<br>3- The useful information is held in variables to be displayed.<br>4- The Java objects are displayed to the user according to their selection of descriptions. |
| **POST CONDITIONS** | Event description is displayed |
| **ALTERNATIVE FLOWS** | - |

| | |
|---|---|
| **USE CASE** | Selecting Filters |
| **ID** | UC 5 |
| **BRIEF DESCRIPTION** | User may choose to apply filters to eliminate events of uninterest. |
| **PRIMARY ACTORS** | Google Maps, User |

| | |
|---|---|
| **SECONDARY ACTORS** | - |
| **PRECONDITIONS** | Events must be tagged before displaying. |
| **MAIN FLOW** | 1- The filter screen is displayed with all the filters on.. |
| **POST CONDITIONS** | Unchecked events are removed from the map. |
| **ALTERNATIVE FLOWS** | - |
| **USE CASE** | Displaying account information |
| **ID** | UC 6 |
| **BRIEF DESCRIPTION** | The information about the user account will be displayed on this selection. |
| **PRIMARY ACTORS** | User |
| **SECONDARY ACTORS** | - |
| **PRECONDITIONS** | User must not have logged in as a guest. |
| **MAIN FLOW** | 1- The information is pulled from the servers in an encrypted way.<br>2- The screen shows the following:<br>a. Change options for both username and password.<br>b. Delete my account option.<br>c. Privacy<br>d. My past events<br>e. Sign out |
| **POST CONDITIONS** | Account information is displayed |
| **ALTERNATIVE FLOWS** | - |

| | |
|---|---|
| **USE CASE** | Promotion |
| **ID** | UC 7 |
| **BRIEF DESCRIPTION** | A page which allows users to enter their promo codes to have free drinks at events. |
| **PRIMARY ACTORS** | User, Admins |
| **SECONDARY ACTORS** | - |
| **PRECONDITIONS** | User must have acquired a promo code. |
| **MAIN FLOW** | 1- The screen shows up with a textbox allowing users to input their promo code.<br>2- After the code in inputted, the servers check the legitimacy of the code.<br>3- If the code is valid, the user is granted free drinks at events. |
| **POST CONDITIONS** | - |
| **ALTERNATIVE FLOWS** | - |

| | |
|---|---|
| **USE CASE** | Accessing the Main Menu |
| **ID** | UC 8 |
| **BRIEF DESCRIPTION** | User must tap the three-bar button on the left of the search bar to access the main menu. |
| **PRIMARY ACTORS** | User |
| **SECONDARY ACTORS** | - |

| | |
|---|---|
| **USE CASE** | Selecting “Help” in Main Menu |
| **ID** | UC 8.2 |
| **BRIEF DESCRIPTION** | User gets provided with assistance methods to overcome any issues within the application. |
| **PRIMARY ACTORS** | User |
| **SECONDARY ACTORS** | Bot |
| **PRECONDITIONS** | User is in the main menu of What’s Mappin |
| **MAIN FLOW** | 1. User is in the main menu<br>2. User taps on “Help” button<br>3. User selects between Voice Assistance, Contact-a-bot (a bot that answers predefined questions) and FAQs |
| **POST CONDITIONS** | User gained access to the help |

The user must be logged in the main page of our application.

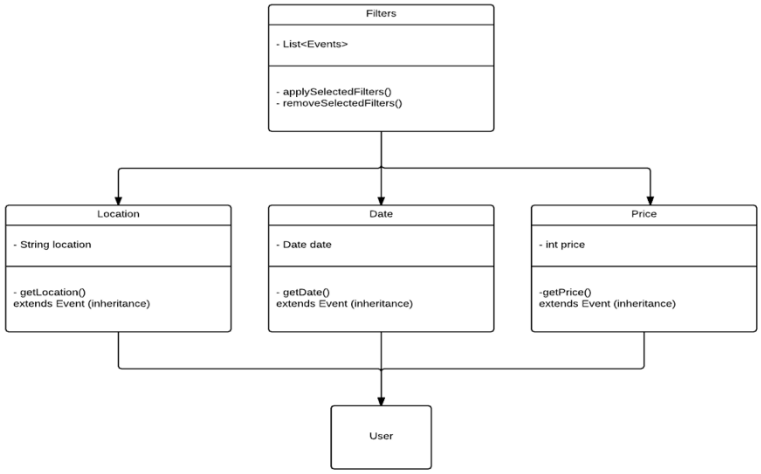
| | |
|---|---|
| | methods and their issue got resolved |
| **ALTERNATIVE FLOWS** | |
| **USE CASE** | Selecting “Wish List” in Main Menu |
| **ID** | UC 8.3 |
| **BRIEF DESCRIPTION** | User gets access to the Wish List where they can view all saved events |
| **PRIMARY ACTORS** | User |
| **SECONDARY ACTORS** | Event Holders |
| **PRECONDITIONS** | User is in the main menu of What’s Mappin |
| **MAIN FLOW** | 1. User is in the main menu<br>2. User taps on “Wish List” button<br>3. Events previously saved by the user are fetched<br>4. Saved events |

| | |
|---|---|
| | displayed in a list |
| **POST CONDITIONS** | User views previously saved events |
| **ALTERNATIVE FLOWS** | |
| **PRECONDITIONS** | |
| **MAIN FLOW** | 1. The user taps on the three bar button to present the main menu. They can access filters, account information, promotion, wish list, help and about us sections |
| **POST CONDITIONS** | User acquired access to the main menu. |
| **ALTERNATIVE FLOWS** | - |
| **USE CASE** | Selecting “About Us” in Main Menu |
| **ID** | UC 8.1 |
| **BRIEF DESCRIPTION** | User gets presented a brief description of the work done for this application. |
| **PRIMARY ACTORS** | User |
| **SECONDARY ACTORS** | - |
| **PRECONDITIONS** | User is in the main menu of What’s Mappin |
| **MAIN FLOW** | 1. User is in the main menu<br>2. User taps on “About Us” button<br>3. A brief description of the work behind the app gets displayed |

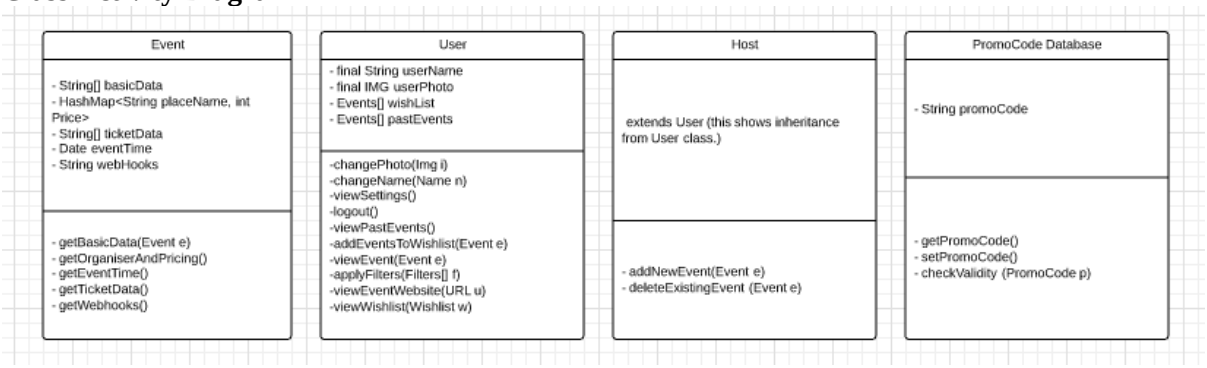
| | |
|---|---|
| **POST CONDITIONS** | User gained access to the “About Us” page |
| **ALTERNATIVE FLOWS** | - |

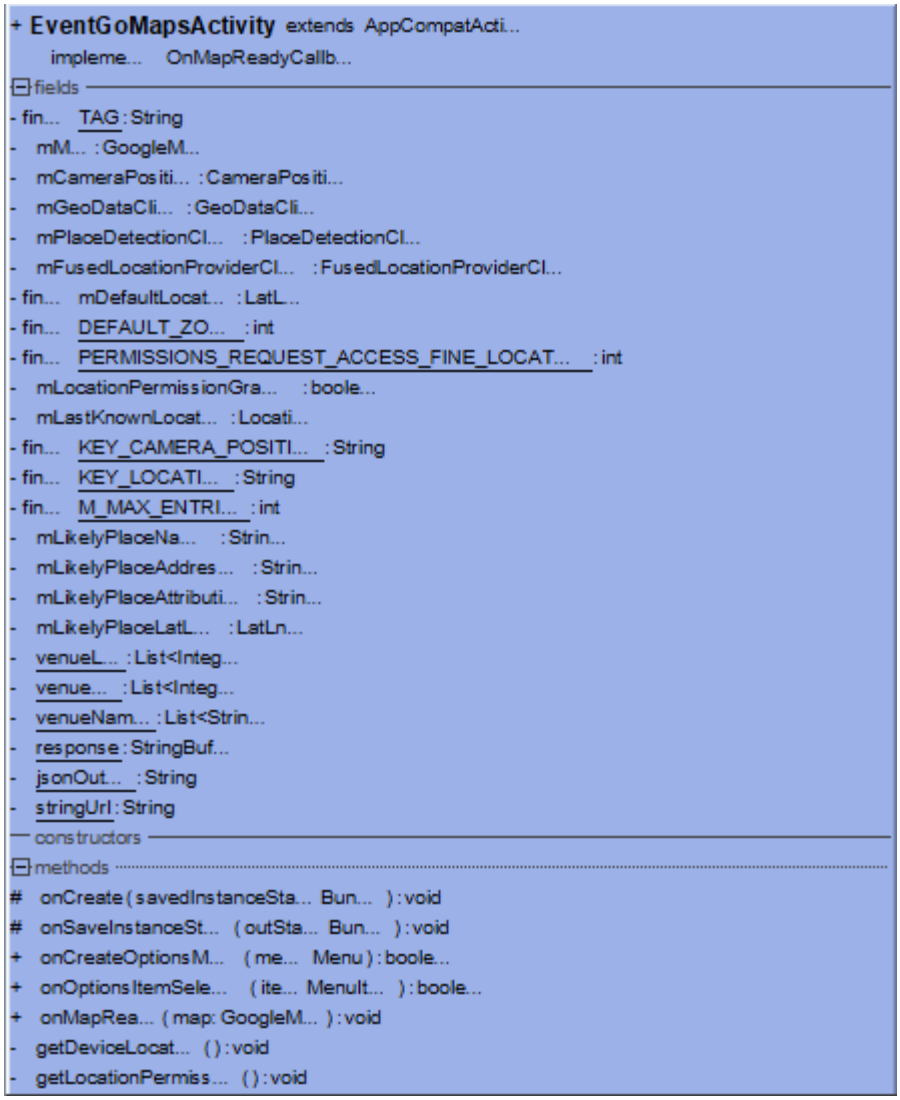
# Object-Oriented Analysis

## Class Entity Diagram

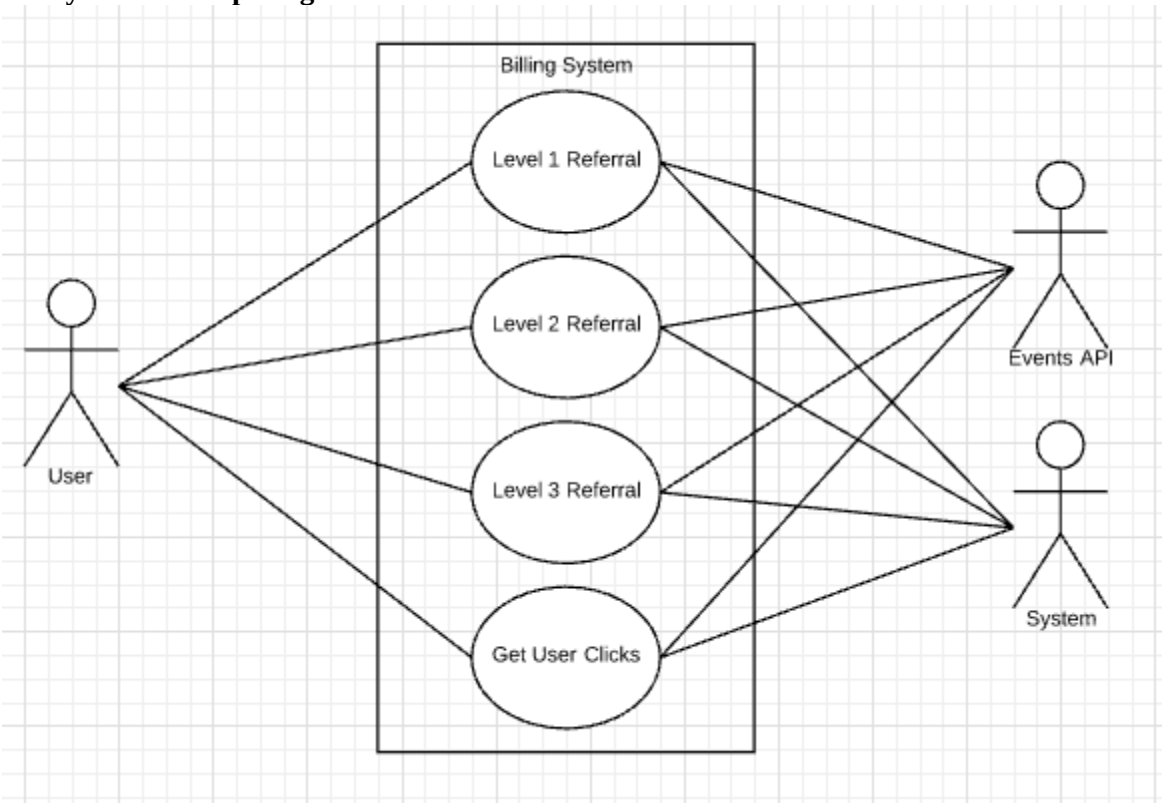



## Class Activity Diagram

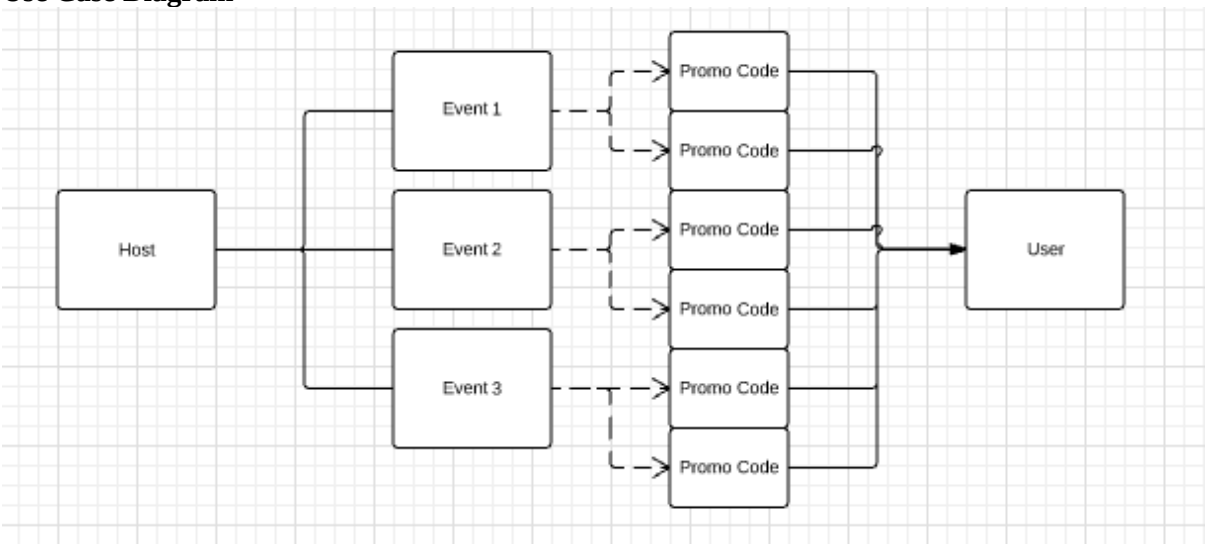


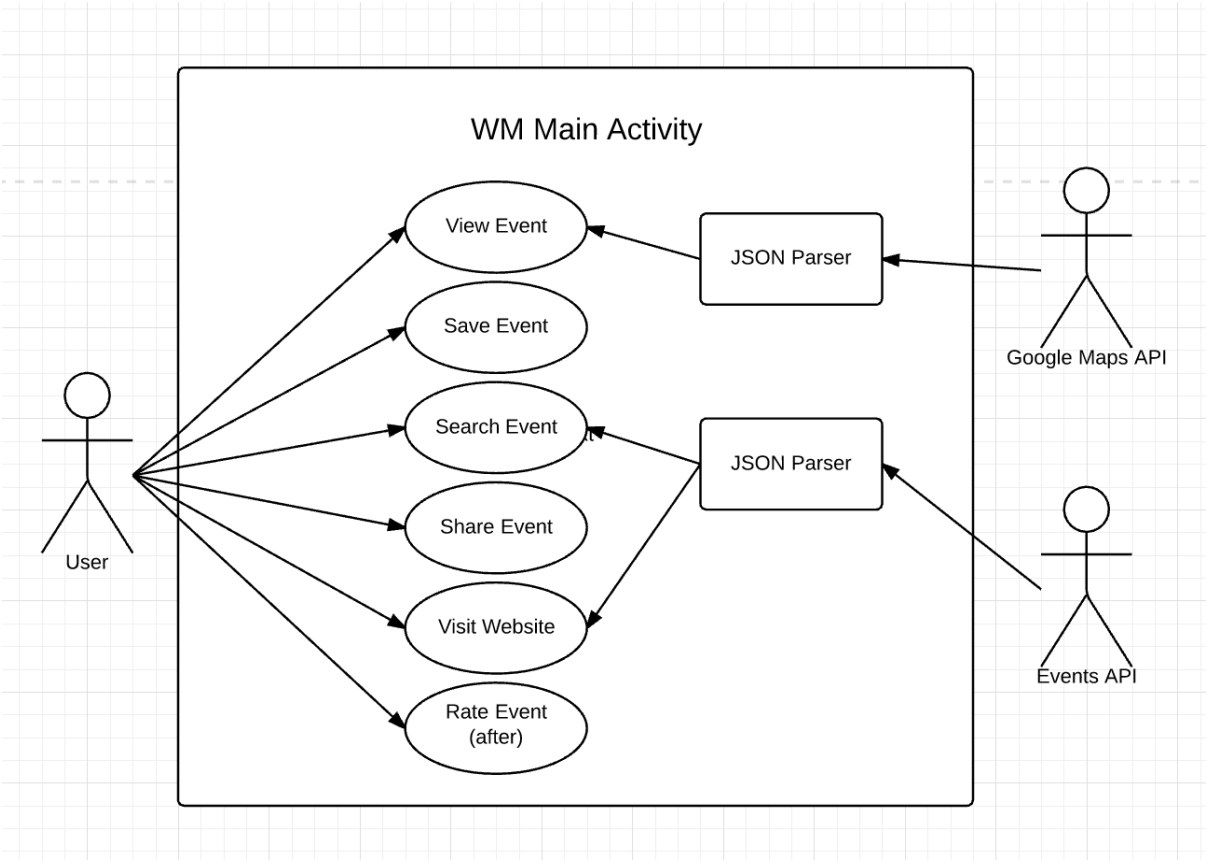

+ EventGoMapsActivity extends AppCompatActi...
impleme... OnMapReadyCallb...
fields
- fin... TAG : String
- mM... : GoogleM...
- mCameraPositi... : CameraPositi...
- mGeoDataCli... : GeoDataCli...
- mPlaceDetectionCl... : PlaceDetectionCl...
- mFusedLocationProviderCl... : FusedLocationProviderCl...
- fin... mDefaultLocat... : LatL...
- fin... DEFAULT_ZO... : int
- fin... PERMISSIONS_REQUEST_ACCESS_FINE_LOCAT... : int
- mLocationPermissionGra... : boole...
- mLastKnownLocat... : Locati...
- fin... KEY_CAMERA_POSITI... : String
- fin... KEY_LOCATI... : String
- fin... M_MAX_ENTRI... : int
- mLikelyPlaceNa... : Strin...
- mLikelyPlaceAddres... : Strin...
- mLikelyPlaceAttributi... : Strin...
- mLikelyPlaceLatL... : LatLn...
- venueL... : List<Integ...
- venue... : List<Integ...
- venueNam... : List<Strin...
- response : StringBuf...
- jsonOut... : String
- stringUrl : String
constructors
methods
# onCreate ( savedInstanceSta... Bun... ) : void
# onSaveInstanceSt... ( outSta... Bun... ) : void
+ onCreateOptionsM... ( me... Menu ) : boole...
+ onOptionsItemSele... ( ite... MenuIt... ) : boole...
+ onMapRea... ( map: GoogleM... ) : void
- getDeviceLocat... ( ) : void
- getLocationPermiss... ( ) : void

**Entity-Relationship Diagram**




**Use Case Diagram**






## Analysis Class diagrams

Class diagrams to define the structure.

- Classes, attributes and operations
- Associations and inheritance
- Data representation

## Activity and Sequence diagrams

- Data transformations
- Message call sequences
- Use case realisation

## Other architecture/UML diagrams (such as flow)

Examples:

# Design

## Design Class Diagram

## Wireframes and UI/UX

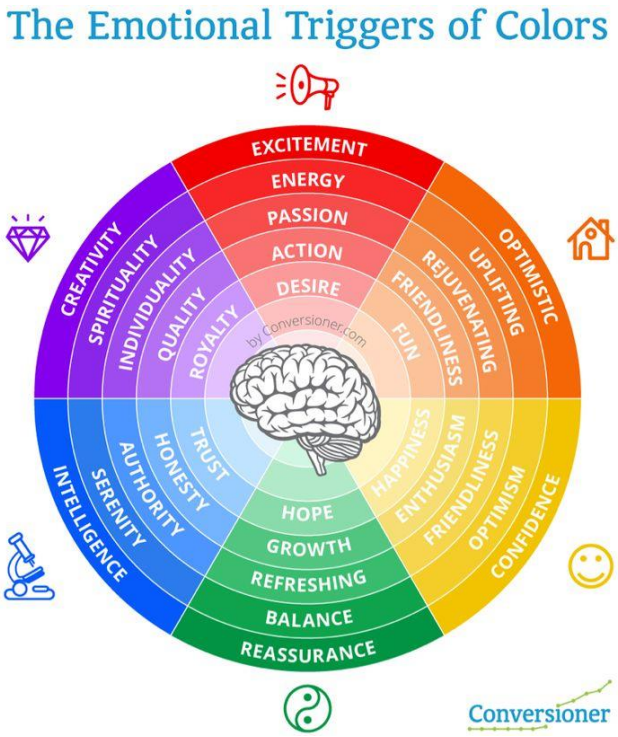
## State Machine Diagram

## Prototype development / MVP

WM considers user interface to be at the heart of our platform's functionality. Since our target demographic will likely be hurried, undecided millenials with little attention to pay to our app in their group scenario, we intend to make the user flow through as simple and intuitive as possible.

One of the primary concerns we have considered is interface familiarity. We are aware of the competitivity of the event mapping industry and although WM will optimise the experience, we will keep the basic industry structure for mapping applications as dictated by GoogleMaps.

When considering the platform's primary colour we channelled our researched into the psychological effect of color schemes. By drawing on research from industry leading companies' color choices, we devised that our primary color should lie in the spectrum of exciting reds, cheerful oranges and clear yellows, (image below left). When considering this spectrum in closer study, the third red tier of 'Passion' (image below right) stood out as a more neutral and subtle form of the red 'excitement' trajectory WM wants to convey, without pushing the full extent of the primary color into our user's screen, which might lead to eye strain and reduced time spent on our platform. Given the nature of our referral fees (detailed in business plan) it is important for WM to be able to engage their users in a way that they spend more time on our platform and explore the different events around them before making their final decision.


## App Implementation

What's Mappin is an event mapping application designed to provide users with the smoothest experience of planning a night out. Our main design target was to appeal to university students. Since all members of our scrum team belong to our target group, we could link our application to likenesses and tastes similar to ours. Having this advantage, we then researched existing mapping applications, and noted inefficiencies in terms of format and design choice. We also conducted a survey...

(**SURVEY**)

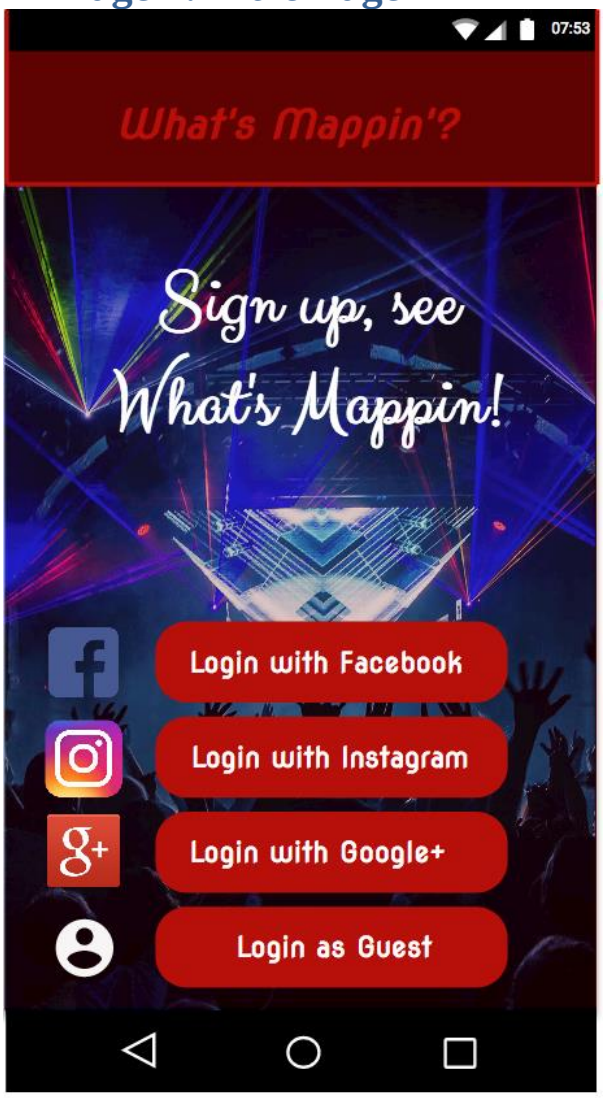
We came to the conclusion that our target group can be most effectively reached by limiting complexity and by offering a visually attractive, modernized interface.
In designing the interface of an application, an issue that needs both discussion and research are the colours which will be used. After taking into account our target group, namely university students looking forward to a night out, we aimed to look for a combination that would portray the excitement of the clubbing experience. After researching the effects of colours in our psychology, we chose the colour red, which symbolizes excitement and activity.
In the first iteration, we focused on functionality, making sure all app components are displayed to the user. We tested red as our primary colour, however we decided that it did not look as attractive and we used it as a secondary colour instead. In the second iteration, we focused to provide the user with the smoothest and easiest experience possible. Our primary colour was changed to transparent dark grey, which is less tiring for the eyes. We also restructured our app design, adding photos of numerous venues, aiming to provide the user

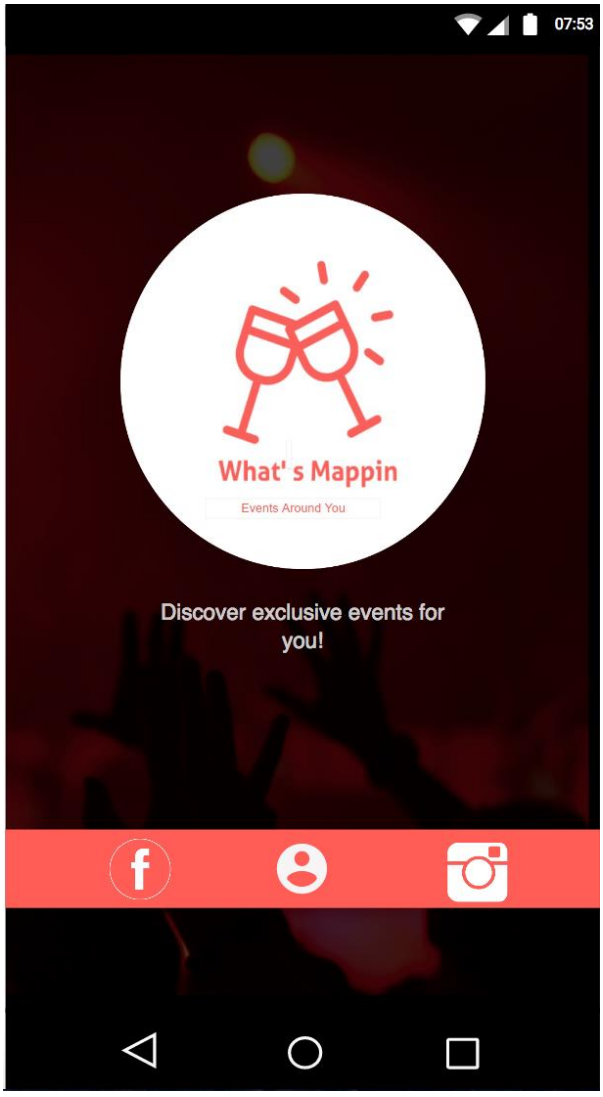
with a glimpse of the experiences that await. We also focused on fonts being clear and simple (Antic and Helvetica). Font sizes are between 16-20, so the user has no trouble reading the text, and the text itself does not cover the whole screen. We reformatted the buttons to a more modern style, by removing edges and minimizing the antithesis between them and the background. Emphasizing on simplicity, we removed unnecessary buttons; for instance we included the search filters in the main interface, making our menu less complicated. We aimed for style consistency across our app, using the same background colours, fonts and styles in every page.

# Iteration 1-2 Comparison

After conducting a survey, approaching 30 university students to gain feedback on our first iteration, our results helped us generate a number of design improvements that could elevate our application. Below is a description of how every page of our app got developed in terms of design.

## Page 1: Intro Page



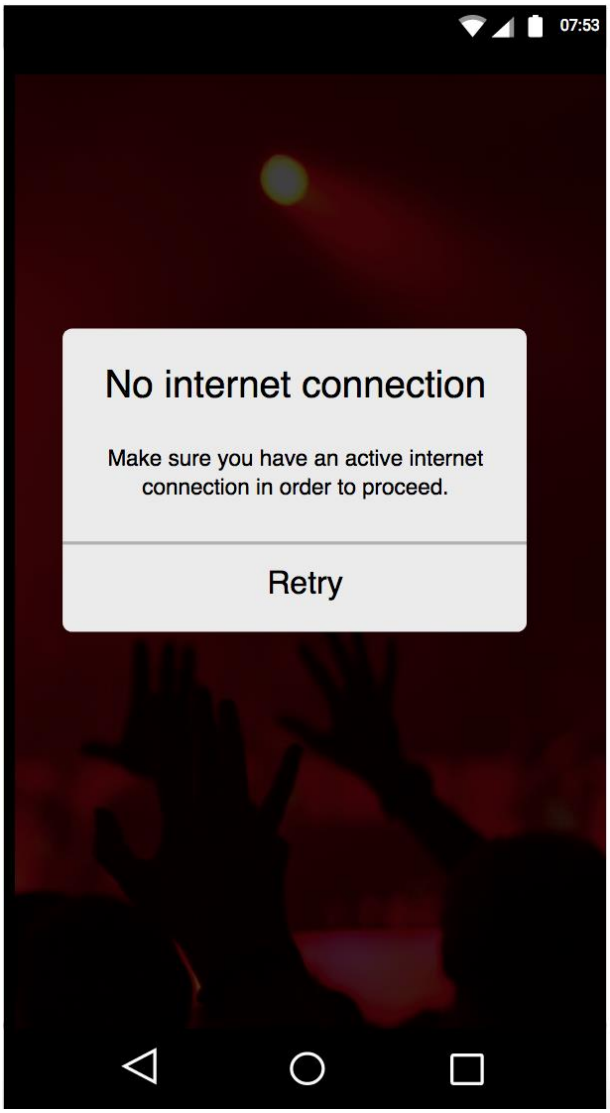

*Iteration 1*

- User is welcomed to our app
- Login options: Facebook, Instagram, Google+, guest
- Background is a blue clubbing image
- Clicker script and Nova Round fonts

*Iteration 2*

- Focus on simplicity, fewer colours, shorter text, in order to appeal.
- Background is dark grey and displays a red clubbing image with low opacity
- Red is the secondary colour
- Added our app logo-two glasses cheering portraying the nightlife.

## Page 2: No Internet

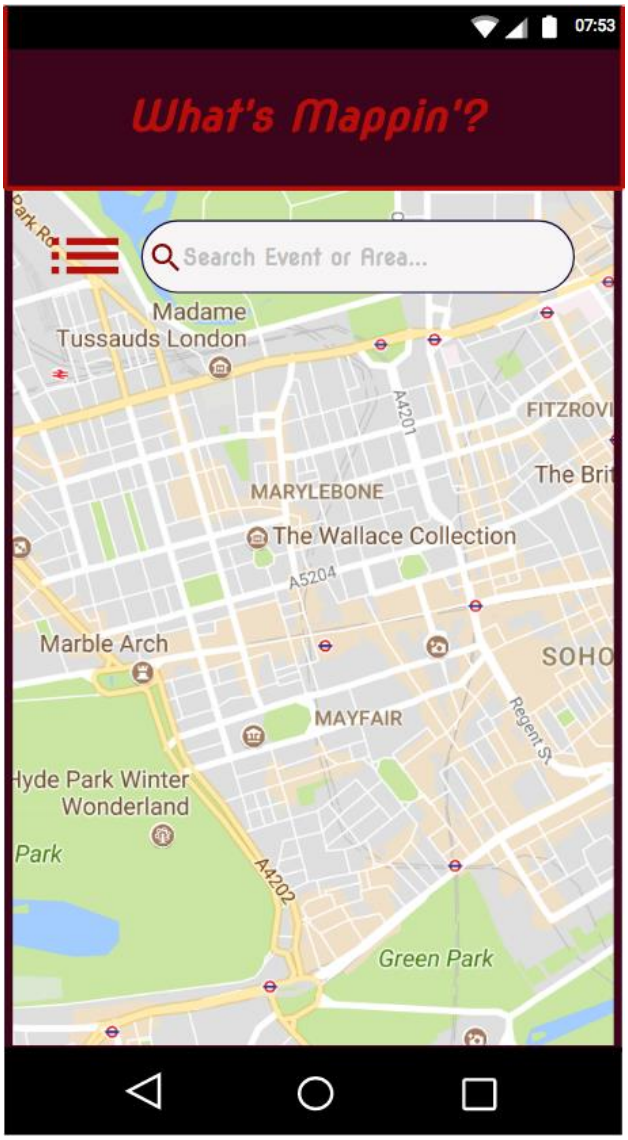

*Iteration 1*

- Not every user will be connected to the internet when opening our app, therefore this page is useful as it informs them about the case.

*Iteration 2*

- Adapted this page's style to our new design colours and features.

## Page 3: Main Interface





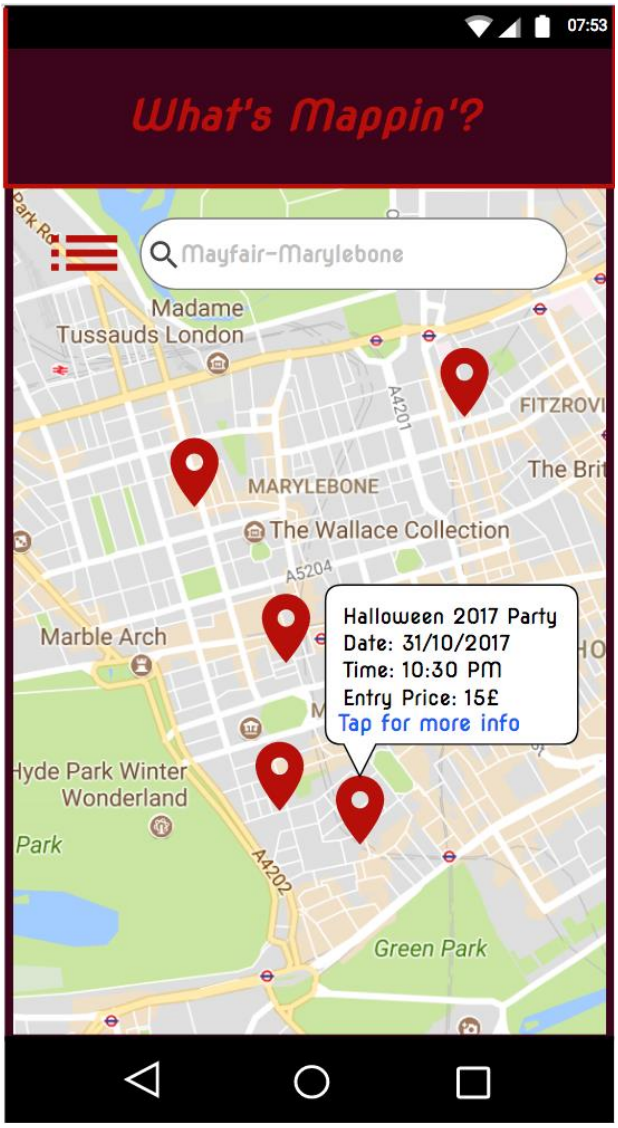
*Iteration 1*

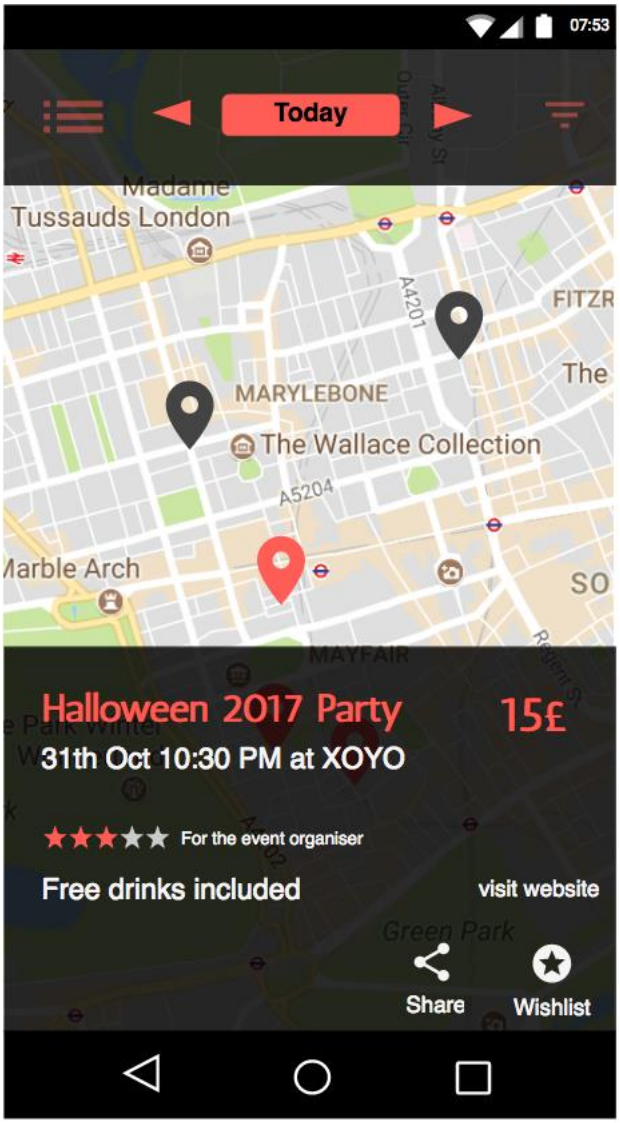
- Google Maps API displaying the map.
- Search bar on top, in order for users to search for events based on name or area.
- Three bar menu on top left redirects user to main menu.

*Iteration 2*

- Design and colour updated.
- Once map is displayed, local events are also displayed in the form of bubbles.
- Search bar replaced with date bar. We concluded that the common user is more likely to attend an event in the next few days. Search bar deemed unnecessary due to existence of the drop down menu of search filters next to the date.

## Page 4: Search Results




*Iteration 1*

- Events mapped as bubbles across the map, after user searches.
- When a bubble gets tapped, info about the specific event gets briefly presented.
- User can tap again to go to another page with specific information about the event and venue.

*Iteration 2*

- Selected event bubble coloured red compared to the rest (dark grey).
- Bottom screen part displays event info and also lets user visit the venue's website, share or add event to the wish list.
- Event organizer rating included to help users decide.

## (Page 4.1): Detailed Event Info

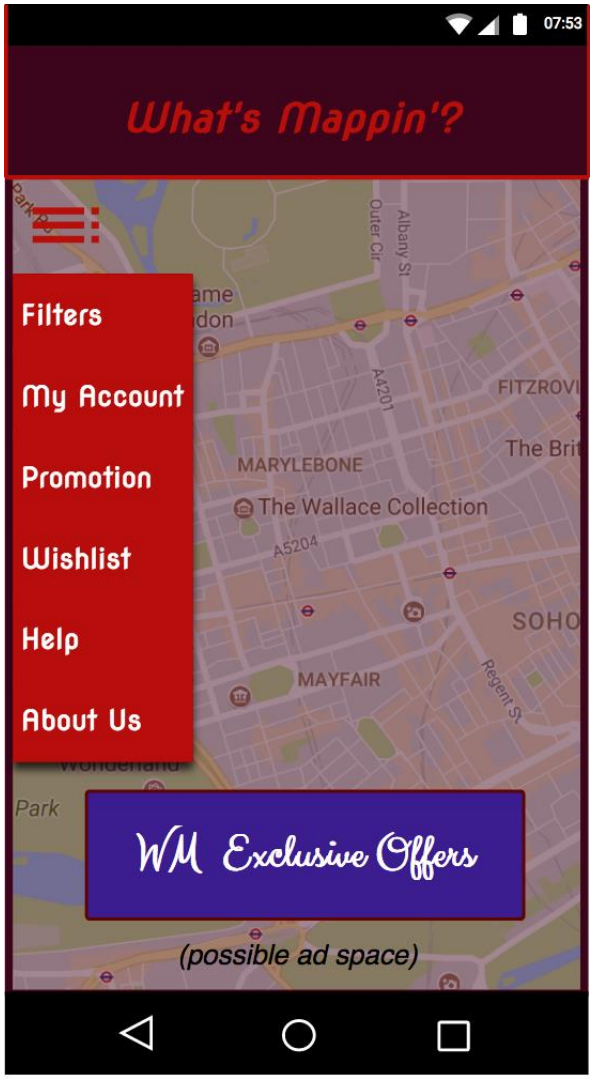

*Iteration 1*

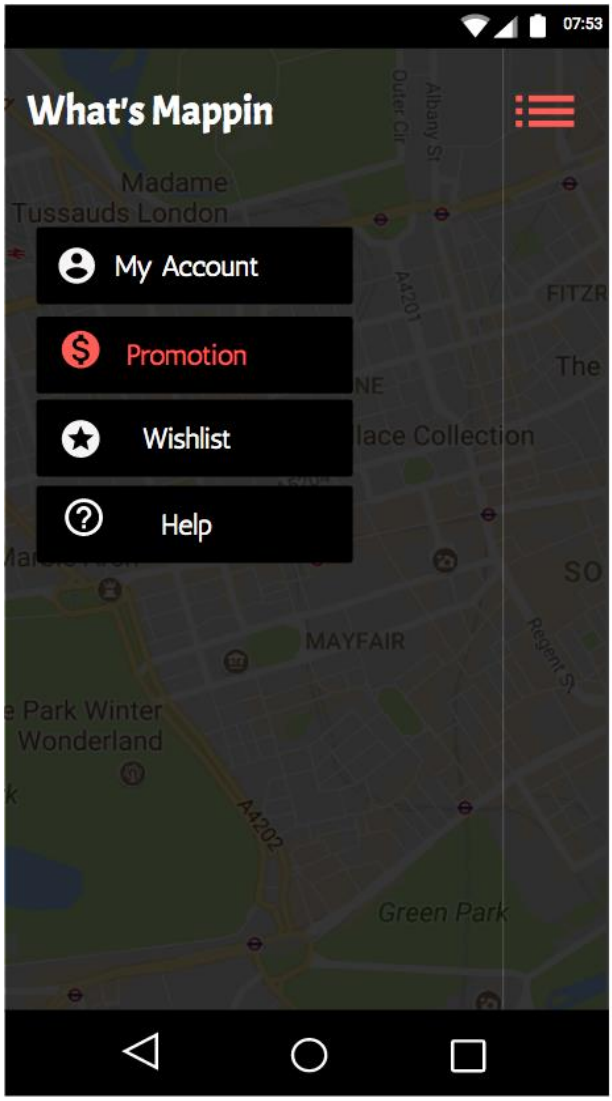
- User gets redirected here after tapping for more info on an event.
- User has access to pictures of event venue
- Event Price, date, time, location determined
- Venue Rating based on users' ratings
- User can visit event website to book tickets

*Iteration 2*

- Integrated all necessary event information in Page 4, making this page unnecessary.

## Page 5: Main Menu



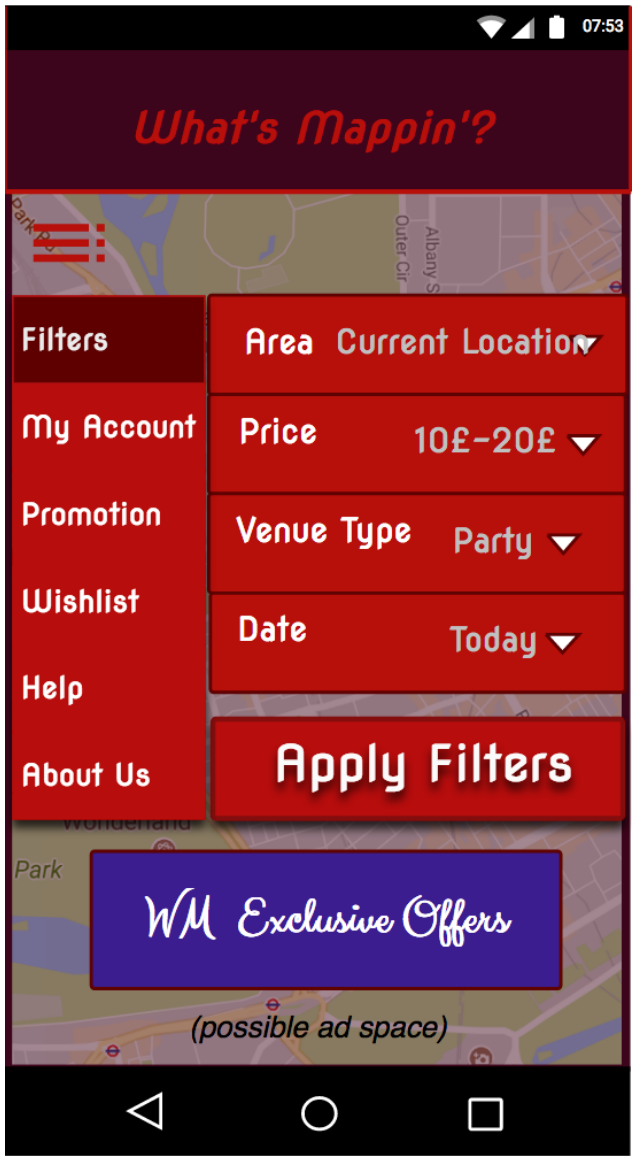

*Iteration 1*

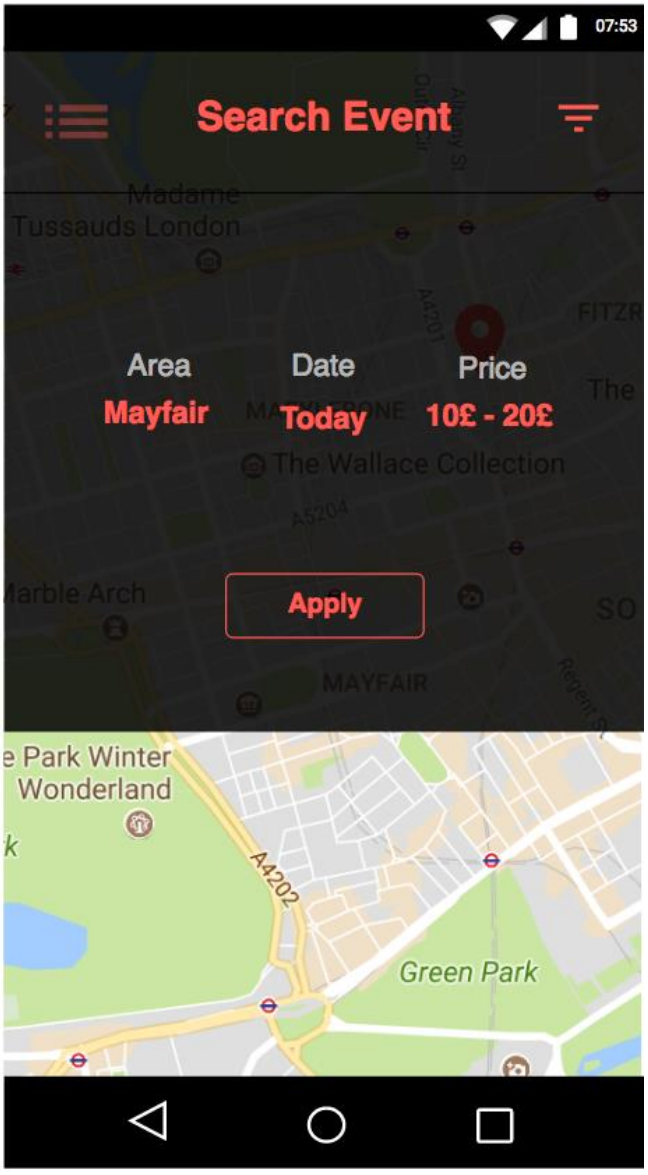
- Main menu comprises of filters, my account, promotion, wish list, help and about us sections.
- WM (What's Mappin) exclusive offers are displayed on bottom of page.
- Map opacity reduced to 52% to emphasize on the main menu.
- All menu options are designed using a common red button format.

*Iteration 2*

- Icons included next to every button making them more attractive
- Buttons and background reformatted according to new design.
- WM Exclusive offers removed due to repetition and similarity to Promotion (both offers)
- Each option lights up whenever tapped

## Page 6: Search Event (Filters)



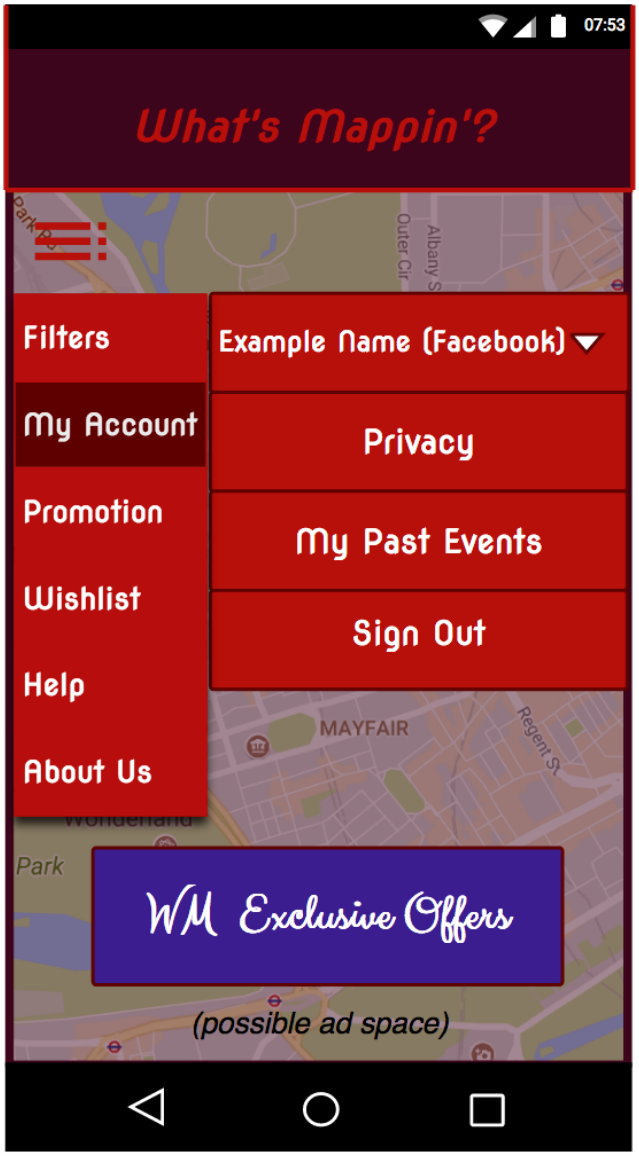

*Iteration 1*

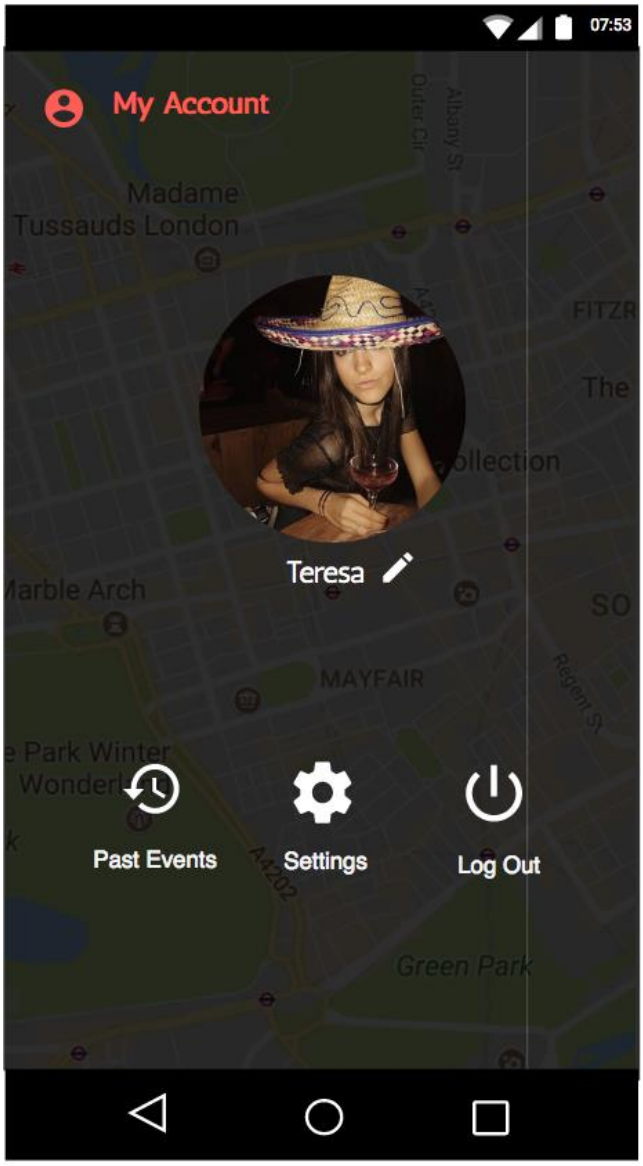
- Filters are the first option on the main menu.
- Used to search for specific events, taking into account area, price, venue type and date.
- User must tap Apply Filters for the app to start searching for events.

*Iteration 2*

- Filters replaced by Search Event.
- Included on Page 3 (Main Interface) on the top right of the screen.
- Quicker way of searching for events (no need to access the main menu)
- Search results based on area, date and price.

## Page 7: My Account



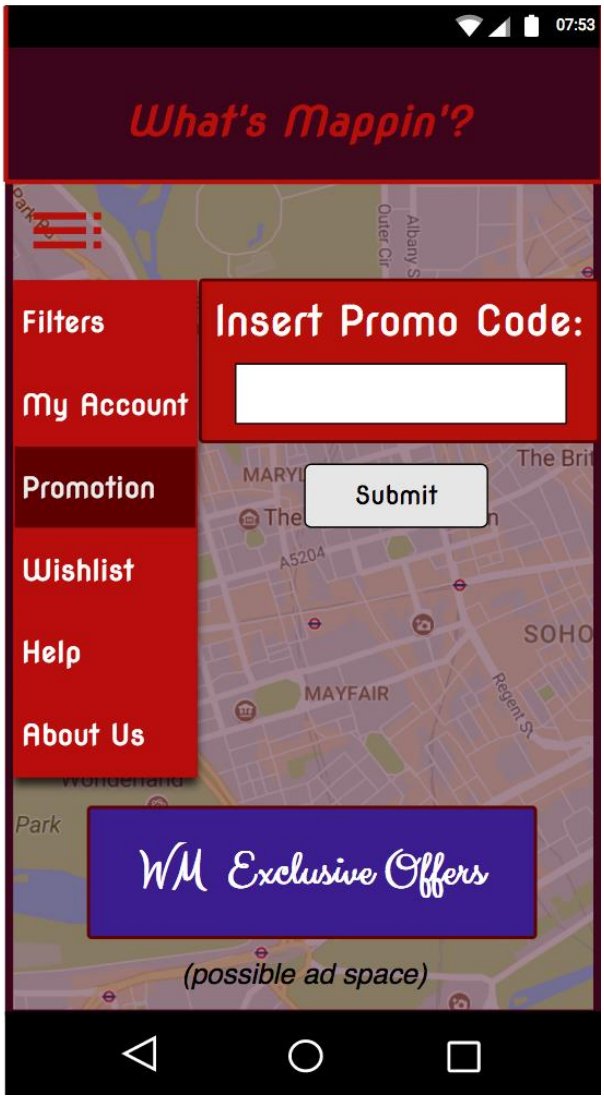

*Iteration 1*

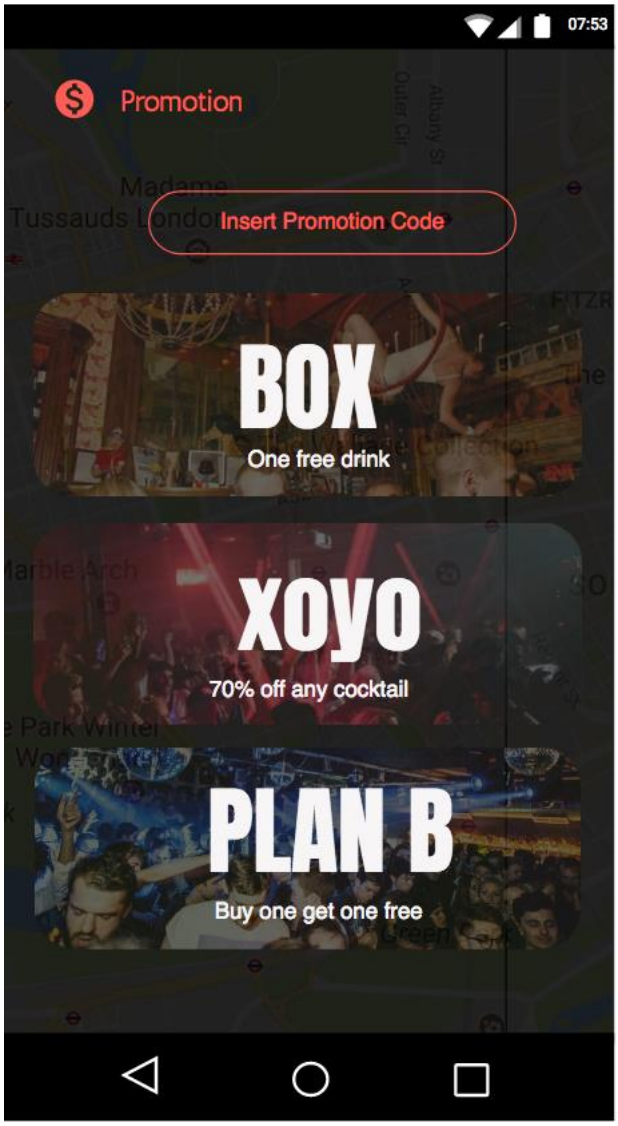
- User's social network name is displayed
- User can access privacy settings
- User can view previously attended events
- User can sign out of their social network profile

*Iteration 2*

- User's profile picture is fetched from signed in social network account. User can change/remove it if desired
- User can edit the name they use inside the app, also fetched by social network account
- User has access to past events, settings and can log out.

## Page 8: Promotion



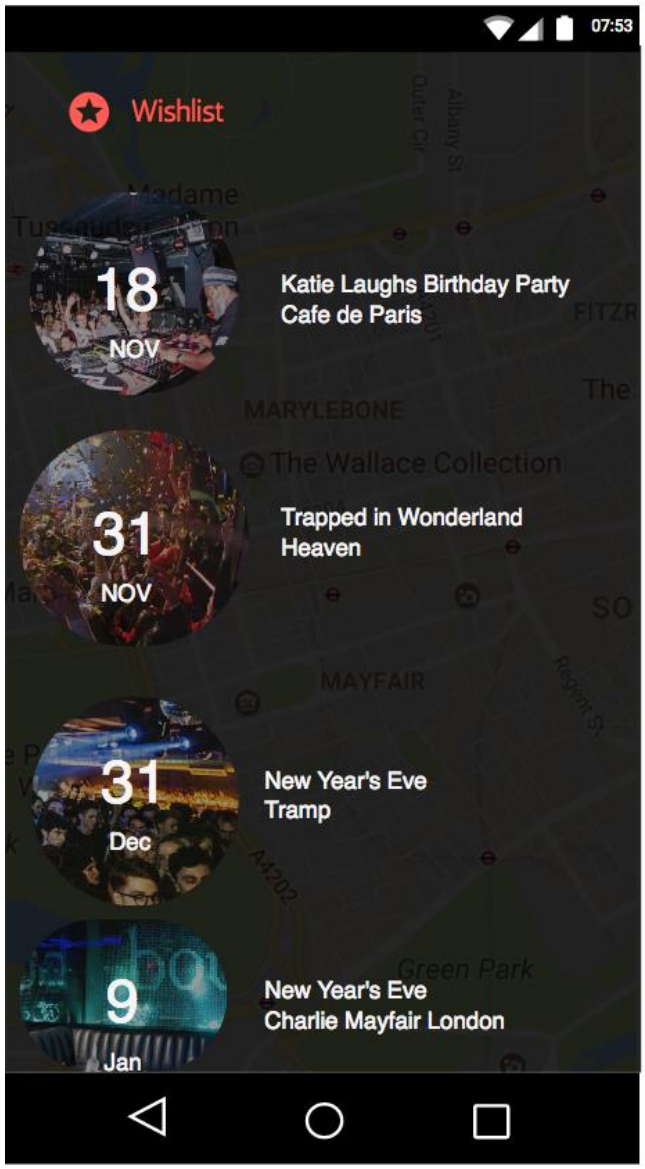

*Iteration 1*

- User can input promotion codes in order to gain discounts and offers in different venues.
- User must type the code in the white text box and tap submit to check the promo code database.

*Iteration 2*

- Insert Promotion Code button on top for the user to type in a promo code they possess.
- All offers and discounts by venues available for the user displayed with venue images individually beneath

## Page 9: Wishlist

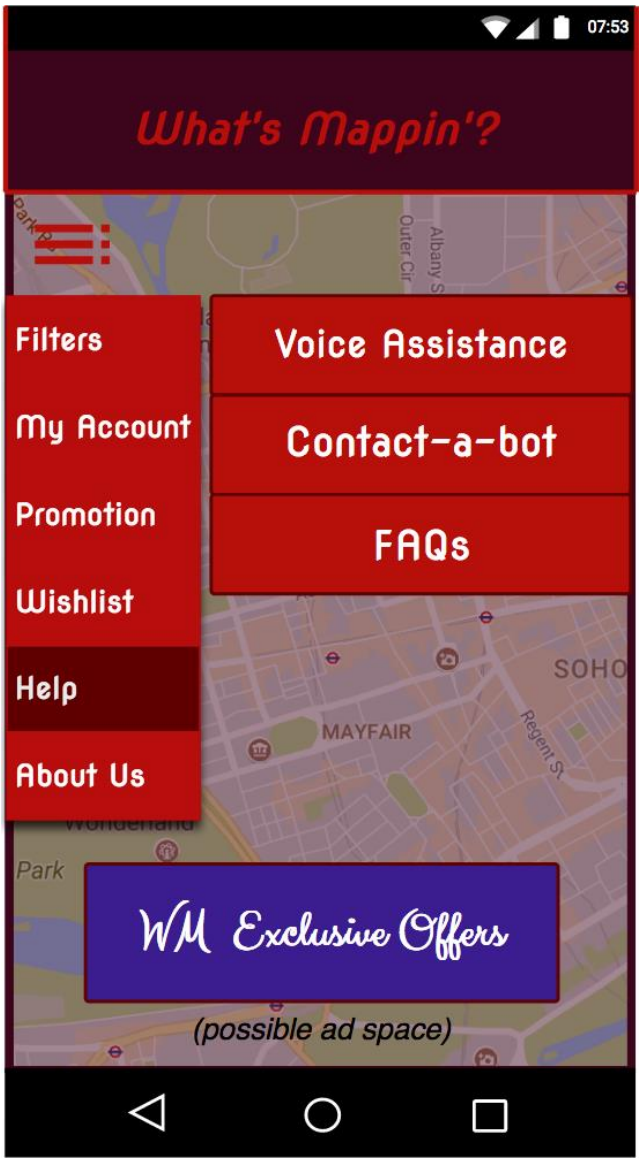

*Iteration 1*

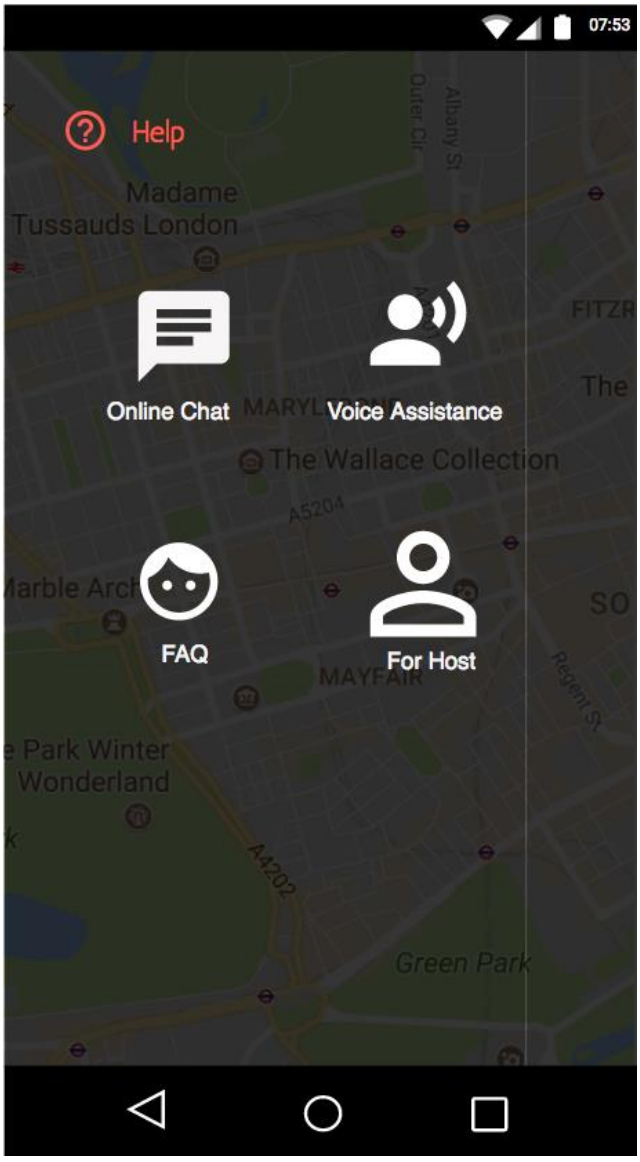
- User can access a list of all previously saved events
- User can select any event from the list to view options

*Iteration 2*

- Improvements in the design of the list
- Larger buttons for every event
- Images of venues included, with the date

## Page 10: Help



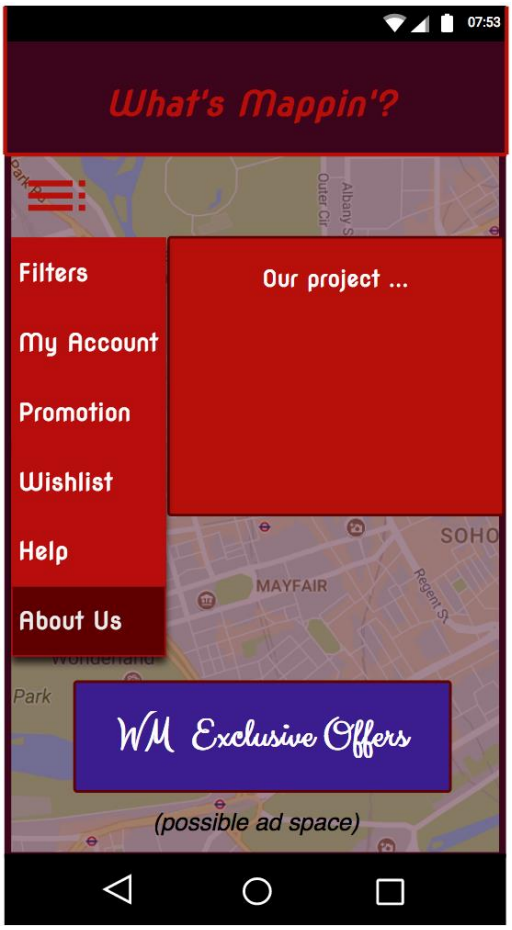

*Iteration 1*

- User can choose between voice assistance, contact-a-bot and FAQs
- Voice assistance targets individuals with vision difficulties (accessibility)

*Iteration 2*

- Icons for every help option added
- Clearer background
- For Host option included providing guidelines for event hosts
- Online chat included for instant communication with the customer

## (Page 11): About Us




*Iteration 1*

- User can access a brief description of our work for this application.

*Iteration 2*

- We wanted our interface to comprise of our app's most necessary functions, therefore we removed this page

All prototypes are made with moqups.com

# Appendix

# Meeting Notes

## Week 1

## Ideas Brainstorming

- § Davin bring up event mapping platform.
- § Yanting suggest focusing on niche market given concerns on market position
- § Teddy argue for focusing on clubbing market instead.
- § Ideas gain overall support from all members
- § Features include event filters, live chat, voice assistance, recommendation

**Technology Implementation**

- § Teddy and Onur brought up technology to form the application such as google Map API https://www.w3schools.com/graphics/google_maps_intro.asp
- § John kept log of data type required

**Event filters**

- § Event Type (e.g. concerts, speeches, internship programs, fundraiser events, entertainment etc.)
- § Date
- § Availability: Fully booked events filtered (e.g. red text) (HTML)
- § Price
- § Age Restriction (18+)
- § Reception (e.g. Today's Most viewed events, most booked events etc.)
- § Distance from current location (Prefer displaying events closer to the user)
- § Reservation Feature (or hyperlink to redirect to official booking page)

## Week 2

Business Framework generated

---

**Personas**

Yanting and Teresa make a proposal of what data need from the personas as following:

- Yanting suggest focusing on niche market given concerns on market position
- Ideas gain overall support from all members
- Features include event filters, live chat, voice assistance, recommendation

**Data log preparation**

- Teddy completed a excel database as raw material.
- Onur suggested using JavaScript .
- The database can assign recurring values to variables using the lookup function to cross-referencing events their location.
- Davin argue build the application as a mobile app, instead of a web, because it can be closer to our user and fit the user scenario. User would seek event near them when they are outside. Teresa agree this opinion as well.

# Week 3

Group Forming and Ideas Generation

---

§ The group went through the specification of the output
§ Yanting developed a mind-map to include all aspects of the report and techniques the team can used for app development.

Figure EventMap 2.0

**Technology Implementation**

Teddy made reasonable reason why we should use desktop other than Mobile from a technical perspective:

1. Writing a Mobile app would need additional knowledge of Android.
2. User can open the webpage and saved it on home screen; But user cannot open the application on desktop if we develop by mobile app

# Week 4

Scrum Implementation and Business Model Canvas

---

**Scrum Implementation**

- Invite member on invision
- Ceremonies: Brief, effective, regular sprint meeting. It shouldn't be more than 15 minute, it should be host at regular time, regular location, it should have something solved each time.
- Everyone has something to work on and report issues occurs to them each week

**Business Model Canvas**

- Davin, John and Yanting made a business model canvas for the app
- Key Parners:
    1. Event Provider
        - To provide us with the actual events.
        - To warn us about any sudden changes, and inform about capacity of events, requirements for entry etc
    2. Advertised Organizations: Our relationship with organizations which we advertise must be characterized of trust and integrity;they must not delay any payments and should be open to alterations. They should also be considerate if we propose any further deals (e.g. advertise us and we will advertise you). Existing relationships are less risky.
    3. Universities: Getting recognized by universities may lead to promotion to university students; As event provider

§ Key Activities:

1. Gather Requirements
    § Data: Seek the appropriate provider who can give us access to a large number of academic events
    § Design: A London map covering the screen, with a toolbar on the side, Search button
    § Functionality: Toolbar containing filter options for specific event search. Program searches for closest events matching criteria
2. Business Research
    § Search for most appropriate way to implement each component of our project. Refine suggestions too hard and not essential to our project.

§ Value Proposition

§ EventGo: A portable application that allows quick access to all available academic events in London, taking into account event specification, location, time, availability, requirements
§ Preparation for Professional Environment:Get to understand company's strategic, tactical and operational objectives, how each firm differs from the other
§ Collaboration: Events filled with students bearing similar interests -> potential cooperation between students;

· Customer Relationships(Student): Targeted promotion towards students seeking internship opportunities, to assemble customer base
§ Channels: University, Clubs that student will go, Societies, Facebook/Twittter share link;
§ Customer Segments: University student, Ambitious young individual, Young Adult Seeking Employment, University graduate searching for a job
§ Cost Structure : Fix cost and Variable Cost
§ Revenue Stream

1. Advertisement: Advertising third parties in our application without interfering with its functionalities (e.g. banner ads)
2. Selling Customer Data: Consumer profiles, events with highest reception, highest demand for job etc.
3. Referral fees: Promote event holders and receive a commision per every website visit they receive
4. 1£ Download/Account fee: Request an amount that seems insignificant in the users' eyes. We target the price to be multiplied by the number of users, so we gain a source of income.

# Week 5

Group Forming and Ideas Generation

---

# Confirmation

· App name: What's Mappin
· Build android app

**Business priorities:**

· No emphasis on short term profits
· Market share as many user as possible
· Club promotional, competitive.

# Research Topic

· Number of clubbing economy size
· Average spending of student

**The decision process between professional events and clubbing event:**
**Professional**

· Quantity to secure profit ~ 90,000 GBP/mth . Sufficient audiences seeking this type of event and sufficient event providers to make our app operate.
  o Bright Network is an organization in London providing those event, they host 6 event just in September, with 1500+ students for each one .
  o Goldman Sachs receives 223,849 job application in 2016 and 4000 secured an internship. Will you go to goldman Sachs Events if you want to secure a job in such competition?

Think about just earning 1 pound for each attendant, in one month we will earn 1500 * 6 = 9000 GBP conservatively from one provider. Assume there is ten event providers: ICEW, TargerJob, Universities Events, Goldman, Morgan Stanley, Brighter Future, Citi, UBS. Our revenue will be 90,000. Assume our cost of doing the app is purely labour (marketing will be mostly collaborating with universities, I will need onur or you to tell me the app cost)
90,000/6 group member = 150,000 for each people.
Keen attendees who are willing to pay and attend every event. True, in terms of number, we will have smaller audience compare to clubs people. But those people trying to land a career are driven, keen. Just ask 3rd Economics students at UCL, what are they ups to? For them, every event gives them larger chance of being employed:

- They would pay a 12 pound railway ticket;
- They would spent 4 hours writing their motivation;
- They would write thank-you emails for every HR they met. There efforts gives as an inelastic price of demand.
- The Wall Street covering letter library launch in October cost 30 pounds. In 30 days they earned 97,200 pounds.

If we are the only provider who are able to secure the network with Fortune 500 and Top Universities, shape the user habit and let them be dependent on us in every recruiting season. We will enjoy an inelastic demand, we have the pricing power.

- Event provider want more attendees, not just for them, but their rival companies. This is the fundamental difference with clubs. Clubs will be happy if everyone only come to their clubs; Company will be happy if they got the brightest people--- They are more likely to achieve that through creating a competition among attendees. Because they are getting headache in selecting candidate already. If you go to other companies event, Great, now I know you are not too keen on me ----- One application declined.

**Club**

- Clubbing & concerts (CC) demographic is similar to our's - young adults (aged 18-25), with a concentration on students (who are easier to entice and are an easier demographic to target for early technology adoption).
- CC attracts a large number of people
  - Nearly 40 per cent of Brits go on a late night out once a week, a figure which is largely consistent across age groups. - (http://www.express.co.uk/life-style/life/725719/night-out-average-spend-Britain-drink-food-clubbing-partying)
- CC attendees are impulsive (and often due to a state of intoxication & lower inhibitions) and hence can be targeted up til right before an event has begun:
  - Over a third of 18-21 year olds don't leave the house until between 10pm-12am and 88 per cent of respondents stay out for up to six hours.
- CC are high revenue events operating with high profit margins. The industry is built upon promotion (offering incentives and offers to get people through the doors).
  - "The late night economy generates £66 billion annual revenue and is responsible for one third of town turnover. With this in mind it's important we continue support and invest to ensure its future."
  - The report shows that Brits spend £58 or over on an average night out (£58.48) just on food, drinks, entry fee and transport. - (http://www.express.co.uk/life-style/life/725719/night-out-average-spend-Britain-drink-food-clubbing-partying)
  -
- Students have a high PED for CC, hence our app (working with event companies) could offer discounts & free drinks to events.
- CC has a high number of listing per evening and are highly competitive, hence our platform could be a powerful promotion tool for venues.

Eventually all member agree to focus on Club

# Week 6

# Roles

**John**: The Designer - Writing for the designing part (25%)
Give Teresa all the competitors you find via creating an excel sheet
The Client Roadmap: What will a student and client see when using the app? What pages will they go through? Registration page (Facebook and Instagram integration), Filter page, Map page, Setting page.
Textual use-cases
The user interface mark-up for each pages. Find an mark-up app that is suitable for you. Please check the example in Dribble:
https://dribbble.com/shots/2747314-Event-Discovery-App
https://dribbble.com/shots/3258253-Map
https://dribbble.com/shots/3749277-Receipt-App
Or others you have found
**Teresa**: Marketer - Writing for the analysation part (25%)
Pricing mechanism
How much is the average student spending
How to create a win-win situation for the event provider
How to attract students customer/Customer acquisition cost
**Onur**: Developer the functionalities reference is the requirement file
Route map functionalities
Time tag functionalities
Facebook and Instagram API
Write down what comes in your way and tell the group asap, the group will try remove the barriers, if not, we will record it into sprint meeting. The difficulties you face will goes into the design part.
**Teddy**: Product Owner, writing for the Business Model (20%)
Problem, Solution, Customer, Priority and competitive advantage
Business Case: Use the business model canvas, analysation framework such as Porter five forces. I think the justification you write on last week is very good, let's also put down the professional's idea, why it will not work for us and why this app will be better. Industry & market feasibility analysis
- Go-to-market strategy and revenue model: The referral fees + commision
Logo making. You can draw one. If not, specify what would it look like and upload here.
The following is the Nials note of factor he will consider:
- There is a clear problem defined. Solution presented is well-thought and of natural remedy for problem.
- Chosen business frameworks, methodologies, and project management frameworks works complementarity. It is clear why they are chosen to achieve a particular milestone.
- Industry/market analysis is detailed, with clear action plan.
- Revenue model reflects the analysis done and evidence found.
**Yanting**: ScrumMaster- Turn out there was no Scrum execution section so will take some task and writing above (25%)
Make a structure for the report
The Requirement, (check Teddy's master backlog and our previous backlog) very crucial Onur is depending on it to write his coding so be specific, find example in the 'additional' tab under moodle.
This part should be the conclusion of the analysation we have
- (Non) functional requirements, with MOSCOW
Here is Niall's comment that can help you:
- Requirements are clearly defined, adopting user-centric approach.

- Requirements are listed in a clear structure and are prioritised based on a realistic reflection of user needs. Detailed action plan can be derived from said requirements.
- Diagrams were made to help depict user interactions with the system.

Burndown chart

This week product backlog, can you put them into the excel sheet you created to have consistency? This time add a estimated hours for your and the rest of the team task. Give assumptions first and if the rest of the team things it don't fit

# Davin

1. Application Testing, work closely along with Onur
2. Create a Database that is usable for Onur

Because you were not here on Friday so this is the two task here now the team gives you. Yanting know that you talked to Teddy, has he given you extra items?

User Interface

---

# Source Code for WhatsMappin:

```
package com.example.currentplacedetailsonmap;

import android.content.DialogInterface;
import android.content.pm.PackageManager;
import android.graphics.Color;
import android.location.Location;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.os.StrictMode;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.FrameLayout;
import android.widget.TextView;

import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.location.places.GeoDataClient;
import com.google.android.gms.location.places.PlaceDetectionClient;
import com.google.android.gms.location.places.PlaceLikelihood;
import com.google.android.gms.location.places.PlaceLikelihoodBufferResponse;
import com.google.android.gms.location.places.Places;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.model.PolylineOptions;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;

import org.apache.poi.hssf.usermodel.HSSFCell;
import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import org.apache.poi.poifs.filesystem.POIFSFileSystem;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

public class EventGoMapsActivity extends AppCompatActivity
```

```
    implements OnMapReadyCallback {

private static final String TAG = EventGoMapsActivity.class.getSimpleName();
private GoogleMap mMap;
private CameraPosition mCameraPosition;

// The entry points to the Places API.
private GeoDataClient mGeoDataClient;
private PlaceDetectionClient mPlaceDetectionClient;

// The entry point to the Fused Location Provider.
private FusedLocationProviderClient mFusedLocationProviderClient;

// A default location (Sydney, Australia) and default zoom to use when location permission is
// not granted.
private final LatLng mDefaultLocation = new LatLng(-33.8523341, 151.2106085);
private static final int DEFAULT_ZOOM = 15;
private static final int PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION = 1;
private boolean mLocationPermissionGranted;

// The geographical location where the device is currently located. That is, the last-known
// location retrieved by the Fused Location Provider.
private Location mLastKnownLocation;

// Keys for storing activity state.
private static final String KEY_CAMERA_POSITION = "camera_position";
private static final String KEY_LOCATION = "location";

// Used for selecting the current place.
private static final int M_MAX_ENTRIES = 5;
private String[] mLikelyPlaceNames;
private String[] mLikelyPlaceAddresses;
private String[] mLikelyPlaceAttributions;
private LatLng[] mLikelyPlaceLatLngs;

private static List<Integer> venueLng = new ArrayList<Integer>();
private static List<Integer> venueLat = new ArrayList<Integer>();
private static List<String> venueNames = new ArrayList<String>();

private static StringBuffer response = new StringBuffer();
private static String jsonOutput;
private static String stringUrl ="https://maps.googleapis.com/maps/api/directions/json?origin="
        + "51.50987,-0.11809" +"&destination=" + "52.48948,-1.89856" +
        "&key=AIzaSyDgb1sP_RQkqr8K1USTU0B5uK_LzUHdLjM";

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Retrieve location and camera position from saved instance state.
    if (savedInstanceState != null) {
        mLastKnownLocation = savedInstanceState.getParcelable(KEY_LOCATION);
        mCameraPosition = savedInstanceState.getParcelable(KEY_CAMERA_POSITION);
    }

    // Retrieve the content view that renders the map.
    setContentView(R.layout.activity_maps);

    // Construct a GeoDataClient.
    mGeoDataClient = Places.getGeoDataClient(this, null);

    // Construct a PlaceDetectionClient.
    mPlaceDetectionClient = Places.getPlaceDetectionClient(this, null);

    // Construct a FusedLocationProviderClient.
    mFusedLocationProviderClient = LocationServices.getFusedLocationProviderClient(this);
```

```
    // Build the map.
    SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
    mapFragment.getMapAsync(this);

}

@Override
protected void onSaveInstanceState(Bundle outState) {
    if (mMap != null) {
        outState.putParcelable(KEY_CAMERA_POSITION, mMap.getCameraPosition());
        outState.putParcelable(KEY_LOCATION, mLastKnownLocation);
        super.onSaveInstanceState(outState);
    }
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.current_place_menu, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    if (item.getItemId() == R.id.option_get_place) {
        showCurrentPlace();
    }
    return true;
}

@Override
public void onMapReady(GoogleMap map) {
    mMap = map;
    // Use a custom info window adapter to handle multiple lines of text in the
    // info window contents.
    mMap.setInfoWindowAdapter(new GoogleMap.InfoWindowAdapter() {

        @Override
        // Return null here, so that getInfoContents() is called next.
        public View getInfoWindow(Marker arg0) {
            return null;
        }

        @Override
        public View getInfoContents(Marker marker) {
            // Inflate the layouts for the info window, title and snippet.
            View infoWindow = getLayoutInflater().inflate(R.layout.custom_info_contents,
                    (FrameLayout) findViewById(R.id.map), false);

            TextView title = ((TextView) infoWindow.findViewById(R.id.title));
            title.setText(marker.getTitle());

            TextView snippet = ((TextView) infoWindow.findViewById(R.id.snippet));
            snippet.setText(marker.getSnippet());

            return infoWindow;
        }
    });

    //excelReader();
    testPlotter();
    //fromExcelPlotter();
    // Prompt the user for permission.
    getLocationPermission();

    // Turn on the My Location layer and the related control on the map.
    updateLocationUI();
```

```
    // Get the current location of the device and set the position of the map.
    getDeviceLocation();

}

private void getDeviceLocation() {
    /*
     * Get the best and most recent location of the device, which may be null in rare
     * cases when a location is not available.
     */
    try {
        if (mLocationPermissionGranted) {
            Task<Location> locationResult = mFusedLocationProviderClient.getLastLocation();
            locationResult.addOnCompleteListener(this, new OnCompleteListener<Location>() {
                @Override
                public void onComplete(@NonNull Task<Location> task) {
                    if (task.isSuccessful()) {
                        // Set the map's camera position to the current location of the device.
                        mLastKnownLocation = task.getResult();
                        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(
                                new LatLng(mLastKnownLocation.getLatitude(),
                                        mLastKnownLocation.getLongitude()), DEFAULT_ZOOM));
                    } else {
                        Log.d(TAG, "Current location is null. Using defaults.");
                        Log.e(TAG, "Exception: %s", task.getException());
                        mMap.moveCamera(CameraUpdateFactory
                                .newLatLngZoom(mDefaultLocation, DEFAULT_ZOOM));
                        mMap.getUiSettings().setMyLocationButtonEnabled(false);
                    }
                }
            });
        }
    } catch (SecurityException e) {
        Log.e("Exception: %s", e.getMessage());
    }
}

private void getLocationPermission() {
    /*
     * Request location permission, so that we can get the location of the
     * device. The result of the permission request is handled by a callback,
     * onRequestPermissionsResult.
     */
    if (ContextCompat.checkSelfPermission(this.getApplicationContext(),
            android.Manifest.permission.ACCESS_FINE_LOCATION)
            == PackageManager.PERMISSION_GRANTED) {
        mLocationPermissionGranted = true;
    } else {
        ActivityCompat.requestPermissions(this,
                new String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
                PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION);
    }
}
@Override
public void onRequestPermissionsResult(int requestCode,
                                       @NonNull String permissions[],
                                       @NonNull int[] grantResults) {
    mLocationPermissionGranted = false;
    switch (requestCode) {
        case PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION: {
            // If request is cancelled, the result arrays are empty.
            if (grantResults.length > 0
                    && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                mLocationPermissionGranted = true;
            }
        }
```

```
    }
    updateLocationUI();
}

private void showCurrentPlace() {
    if (mMap == null) {
        return;
    }

    if (mLocationPermissionGranted) {
        // Get the likely places - that is, the businesses and other points of interest that
        // are the best match for the device's current location.
        @SuppressWarnings("MissingPermission") final Task<PlaceLikelihoodBufferResponse> placeResult =
                mPlaceDetectionClient.getCurrentPlace(null);
        placeResult.addOnCompleteListener
                (new OnCompleteListener<PlaceLikelihoodBufferResponse>() {
                    @Override
                    public void onComplete(@NonNull Task<PlaceLikelihoodBufferResponse> task) {
                        if (task.isSuccessful() && task.getResult() != null) {
                            PlaceLikelihoodBufferResponse likelyPlaces = task.getResult();

                            // Set the count, handling cases where less than 5 entries are returned.
                            int count;
                            if (likelyPlaces.getCount() < M_MAX_ENTRIES) {
                                count = likelyPlaces.getCount();
                            } else {
                                count = M_MAX_ENTRIES;
                            }

                            int i = 0;
                            mLikelyPlaceNames = new String[count];
                            mLikelyPlaceAddresses = new String[count];
                            mLikelyPlaceAttributions = new String[count];
                            mLikelyPlaceLatLngs = new LatLng[count];

                            for (PlaceLikelihood placeLikelihood : likelyPlaces) {
                                // Build a list of likely places to show the user.
                                mLikelyPlaceNames[i] = (String) placeLikelihood.getPlace().getName();
                                mLikelyPlaceAddresses[i] = (String) placeLikelihood.getPlace()
                                        .getAddress();
                                mLikelyPlaceAttributions[i] = (String) placeLikelihood.getPlace()
                                        .getAttributions();
                                mLikelyPlaceLatLngs[i] = placeLikelihood.getPlace().getLatLng();

                                i++;
                                if (i > (count - 1)) {
                                    break;
                                }
                            }

                            // Release the place likelihood buffer, to avoid memory leaks.
                            likelyPlaces.release();

                            // Show a dialog offering the user the list of likely places, and add a
                            // marker at the selected place.
                            openPlacesDialog();

                        } else {
                            Log.e(TAG, "Exception: %s", task.getException());
                        }
                    }
                });
    } else {
        // The user has not granted permission.
        Log.i(TAG, "The user did not grant location permission.");

        // Add a default marker, because the user hasn't selected a place.
```

```
            mMap.addMarker(new MarkerOptions()
                    .title(getString(R.string.default_info_title))
                    .position(mDefaultLocation)
                    .snippet(getString(R.string.default_info_snippet)));

            // Prompt the user for permission.
            getLocationPermission();
        }
    }

    private void openPlacesDialog() {
        // Ask the user to choose the place where they are now.
        DialogInterface.OnClickListener listener = new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                // The "which" argument contains the position of the selected item.
                LatLng markerLatLng = mLikelyPlaceLatLngs[which];
                String markerSnippet = mLikelyPlaceAddresses[which];
                if (mLikelyPlaceAttributions[which] != null) {
                    markerSnippet = markerSnippet + "\n" + mLikelyPlaceAttributions[which];
                }

                // Add a marker for the selected place, with an info window
                // showing information about that place.
                mMap.addMarker(new MarkerOptions()
                        .title(mLikelyPlaceNames[which])
                        .position(markerLatLng)
                        .snippet(markerSnippet));

                // Position the map's camera at the location of the marker.
                mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(markerLatLng,
                        DEFAULT_ZOOM));
            }
        };

        // Display the dialog.
        AlertDialog dialog = new AlertDialog.Builder(this)
                .setTitle(R.string.pick_place)
                .setItems(mLikelyPlaceNames, listener)
                .show();
    }

    private void updateLocationUI() {
        if (mMap == null) {
            return;
        }
        try {
            if (mLocationPermissionGranted) {
                mMap.setMyLocationEnabled(true);
                mMap.getUiSettings().setMyLocationButtonEnabled(true);
            } else {
                mMap.setMyLocationEnabled(false);
                mMap.getUiSettings().setMyLocationButtonEnabled(false);
                mLastKnownLocation = null;
                getLocationPermission();
            }
        } catch (SecurityException e) {
            Log.e("Exception: %s", e.getMessage());
        }
    }

    private String  getMapsApiDirectionsUrl(LatLng origin,LatLng dest) {
        // Origin of route
        String str_origin = "origin="+origin.latitude+","+origin.longitude;

        // Destination of route
        String str_dest = "destination="+dest.latitude+","+dest.longitude;
```

```
    // Sensor enabled
    String sensor = "sensor=false";

    // Building the parameters to the web service
    String parameters = str_origin+"&"+str_dest+"&"+sensor;

    // Output format
    String output = "json";

    // Building the url to the web service
    String url = "https://maps.googleapis.com/maps/api/directions/"+output+"?"+parameters;


    return url;

}

private void fromExcelPlotter() {

    for (int i = 0; i < venueNames.size(); i++) {

        mMap.addMarker(new MarkerOptions().position
                (new LatLng((double) venueLat.get(i), (double) venueLng.get(i)))
                .title(venueNames.get(i)));
    }
}

public void testPlotter() {

    StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
    StrictMode.setThreadPolicy(policy);

    LatLng origin = new LatLng(51.50987, -0.11809);
    LatLng dest = new LatLng(52.48948,-1.89856);

    String url = getMapsApiDirectionsUrl(origin, dest);
    ReadTask downloadTask = new ReadTask();

    downloadTask.execute(url);

}

private class ReadTask extends AsyncTask<String, Void , String> {

    @Override
    protected String doInBackground(String... url) {
        String data = "";
        try {
            MapHttpConnection http = new MapHttpConnection();
            data = http.readUr(url[0]);


        } catch (Exception e) {
            Log.d("Background Task", e.toString());
        }
        return data;
    }

    @Override
    protected void onPostExecute(String result) {
        super.onPostExecute(result);
        new EventGoMapsActivity.ReadTask.ParserTask().execute(result);
    }
    private class ParserTask extends AsyncTask<String,Integer, List<List<HashMap<String , String >>>> {
        @Override
```

```
        protected List<List<HashMap<String, String>>> doInBackground(
                String... jsonData) {
            // TODO Auto-generated method stub
            JSONObject jObject;
            List<List<HashMap<String, String>>> routes = null;
            try {
                jObject = new JSONObject(jsonData[0]);
                PathJSONParser parser = new PathJSONParser();
                routes = parser.parse(jObject);


            } catch (Exception e) {
                e.printStackTrace();
            }
            return routes;
        }

        @Override
        protected void onPostExecute(List<List<HashMap<String, String>>> routes) {
            ArrayList<LatLng> points = null;
            PolylineOptions polyLineOptions = null;

            // traversing through routes
            for (int i = 0; i < routes.size(); i++) {
                points = new ArrayList<LatLng>();
                polyLineOptions = new PolylineOptions();
                List<HashMap<String, String>> path = routes.get(i);

                for (int j = 0; j < path.size(); j++) {
                    HashMap<String, String> point = path.get(j);

                    double lat = Double.parseDouble(point.get("lat"));
                    double lng = Double.parseDouble(point.get("lng"));
                    LatLng position = new LatLng(lat, lng);

                    points.add(position);
                }

                polyLineOptions.addAll(points);
                polyLineOptions.width(4);
                polyLineOptions.color(Color.BLUE);
            }

            mMap.addPolyline(polyLineOptions);

        }}

    }

}
```

# Source Code for JSONParser:

```
package com.example.currentplacedetailsonmap;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;


import android.util.Log;

public class JSONParser {
    private static String url ="http://maps.googleapis.com/maps/api/geocode/json?origin="
            + "51.50987,-0.11809" +"&destination=" + "52.48948,-1.89856" +
            "&keyAIzaSyDgb1sP_RQkqr8K1USTU0B5uK_LzUHdLjM";

    static InputStream is = null;
    static JSONObject jObj = null;
    static String json = "";

    public JSONObject getJSONFromUrl(String url) {

        // Making HTTP request
        try {
            // defaultHttpClient
            DefaultHttpClient httpClient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(url);

            HttpResponse httpResponse = httpClient.execute(httpPost);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    is, "iso-8859-1"), 8);
            StringBuilder sb = new StringBuilder();
            String line = null;
            while ((line = reader.readLine()) != null) {
                sb.append(line + "n");
            }
            is.close();
            json = sb.toString();
        } catch (Exception e) {
            Log.e("Buffer Error", "Error converting result " + e.toString());
        }

        // try parse the string to a JSON object
        try {
```

```
            jObj = new JSONObject(json);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error parsing data " + e.toString());
        }

        // return JSON String
        return jObj;

    }
}
```

# Source Code for MapHTTPConnection

```
package com.example.currentplacedetailsonmap;

import android.util.Log;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

public class MapHttpConnection {
    public String readUr(String mapsApiDirectionsUrl) throws IOException {
        String data = "";
        InputStream istream = null;
        HttpURLConnection urlConnection = null;
        try {
            URL url = new URL(mapsApiDirectionsUrl);
            urlConnection = (HttpURLConnection) url.openConnection();
            urlConnection.connect();
            istream = urlConnection.getInputStream();
            BufferedReader br = new BufferedReader(new InputStreamReader(istream));
            StringBuffer sb = new StringBuffer();
            String line ="";
            while ((line = br.readLine()) != null) {
                sb.append(line);
            }
            data = sb.toString();
            br.close();


        }
        catch (Exception e) {
            Log.d("Exception:", e.toString());
        } finally {
            istream.close();
            urlConnection.disconnect();
        }
        return data;

    }
}
```

# References


All Events in the City. (2016). *About Us.* Available: https://allevents.in/blog/about/ . Last accessed 1 Dec 2017.

Amitech. (2017). *All Events in City - Discover Local Events.* Available: https://itunes.apple.com/us/app/all-events-in-city-discover-local-events/id488116646?mt=8. Last accessed 1 Dec 2017.

Antwonne D. (2017). *How to use the Eventbrite iPhone app.* Available: https://www.eventbrite.co.uk/support/articles/en_US/Multi_Group_How_To/how-to-use-the-eventbrite-iphone-app?lg=en_GB. Last accessed 1 Dec 2017.

Bandsintown. (2017). *Bandsintown Concerts.* Available: https://itunes.apple.com/gb/app/bandsintown-concerts/id471394851?mt=8. Last accessed 1 Dec 2017.

City Data Company. (2017). *Communities are our passion.* Available: https://www.walkaboutapp.com/about/. Last accessed 1 Dec 2017.

DoStuff. (2017). *DoStuff - What To Do Tonight.* Available: https://itunes.apple.com/us/app/dostuff-what-to-do-tonight/id970122355?mt=8. Last accessed 1 Dec 2017.

DoStuff. (2017). *What is DoStuff?.* Available: https://dostuffmedia.com/app-faq. Last accessed 1 Dec 2017

Eventbase. (2016). *Eventbase - the Free Event App for Everyone.*Available: https://itunes.apple.com/gb/app/eventbase-the-free-event-app-for-everyone/id580233134?mt=8. Last accessed 1 Dec 2017.

Eventbase. (2017). *About us.* Available: https://www.eventbase.com/about-us. Last accessed 1 Dec 2017.

Eventbase. (2017). *Eventbase Free.* Available: https://www.eventbase.com/free-event-app. Last accessed 1 Dec 2017.

Eventbrite. (2016). *Eventbrite.* Available: https://itunes.apple.com/gb/app/eventbrite/id487922291?mt=8. Last accessed 1 Dec 2017.

EventNook. (2017). *Event Management Platform.* Available: http://overview.eventnook.com. Last accessed 1 Dec 2017.

EventNook. (2017). *EventNook CheckIn Pro.* Available: https://itunes.apple.com/us/app/eventnook-checkin-pro/id1178895813?mt=8. Last accessed 1 Dec 2017.

Hackett, B. (2015). *TINDER'S FIRST YEAR USER GROWTH STRATEGY.*Available: https://parantap.com/tinders-first-year-growth-strategy/. Last accessed 2 Dec 2017.

Lee, S. (2016). *What Deliveroo Did To Get Traction And Become a Billion-Dollar Company.* Available: https://www.referralcandy.com/blog/deliveroo-marketing-strategy/. Last accessed 2 dec 2017.

McClelland, L. (2016). *5 Marketing Lessons Learned From UberEATS.*Available: https://www.mediapost.com/publications/article/272424/5-marketing-lessons-learned-from-ubereats.html. Last accessed 2 Dec 2017.

DoStuff. (2017). *WE ARE DOSTUFF.* Available: https://dostuffmedia.com/about. Last accessed 1 Dec 2017.

Vendini. (2001). *Mobile Apps.* Available: https://vendini.com/mobile-apps. Last accessed 1 Dec 2017.

Vendini. (2013). *Vendini TicketScan.* Available: https://itunes.apple.com/cg/app/vendini-ticketscan/id376436697?mt=8. Last accessed 1 Dec 2017.