Project Report ID2216 HT19 (Developing Mobile Applications)

App name: Shoppy

Group members: Tom Axberg, Edward Chancellor, George Kotsiopoulos, Jing Li

https://github.com/edchancellor/ShoppyFinal

Introduction

Our goals for this Mobile Applications project were both related to our product and to our own development as Information Technology students. On the one hand, our highest priority was to create a functional application, either web or native, which would tackle a pertinent and relevant issue (in our case, this was to make the grocery shopping experience simpler). It's main concern should be that it is immediately usable. The app should be designed in such a way that it has opportunities for future expansion and/or monetisation. On the other hand, we were also keen to learn new skills about conducting a project through this process. In particular, we wanted to improve our project management skills, task allocation, coding, and market research capabilities.

App Summary

Our proposed app is intended to help you do your weekly/monthly shop. Users can save and store individual meals by inputting all the ingredients needed for that recipe. Then, users can also create shopping lists, where all they have to do is select which pre-created recipes they want to cook that week. The app will then automatically check what ingredients are needed for all those recipes and create a shopping list accordingly. The app will contain four main functions: "New Recipe", "New Shopping List", "My Recipes", and "My Shopping Lists".

Our future intention for bringing this to market is to make it practical and profitable. The app could be made freely downloadable on the App Store or Android Store. Then, users can create their own recipes/shopping lists for free in the standard version of the app, but they will be charged a small amount of fee if they want to download premade recipes online. More information and discussion on this can be found in the Evaluation at the end of the report.

Assignment 1

Pre-observation discussions

After debating different project ideas, our group decided to create an app which would make the weekly shop easier. While there are pre-existing apps which help you to organise shopping lists, such as the ICA app, these did not allow you to create/add your own recipes, and were also very cumbersome and complicated to use (not to mention they were only relevant to one particular supermarket). Our aim was to create an easy to use tool for creating recipes and shopping lists.

After deciding our subject area, we came up with some important questions relating to this, which were as follows: How often do you go food shopping? How do you normally go about doing your weekly/monthly shop? How do you decide what meals you are going to cook? Where do you normally decide what food you are going to buy? Do you use any apps to help you organise your shopping? If so, which ones? If you were to use an app, what functions might be useful? Would you find such an app useful?

We began by asking these questions amongst ourselves, recording our responses. Some of the common themes we came across included: It would be useful to have a very simple tool for making shopping lists; Writing up a shopping list could be time consuming; It is sometimes difficult to decide what to cook, or to remember everything you need to buy; We liked to have variety in what foods we bought; It was common for us to do on average 3-4 big food shops per month, as well as more regular visits to the supermarket for things we had forgotten.

After this, we drew up a preliminary sketch of how our app could look on a whiteboard (Fig 1). While doing so, we noted down any features which we felt could be useful, arriving at the following conclusions: The main features of the app should be readily accessible from the start-up; Large tiles might be a good way to separate the app functions - this is a tried, tested and effective User Interface; The number of functions should be kept to a minimum, in order to make the app simpler. Such notions of design simplicity are supported by Brian Fling.[1]

Observations 1: Paper Prototyping in Electrum

Having formulated a basic menu system for our app, we drew out on paper a prototype version of this - an activity greatly encouraged in the literature (Fig 2).[2] We wanted to avoid what Holtzblatt and Beyer term the "design from the I" – that is, only basing our designs on what we personally would like – so knew it was time to do some market research.[3] At this stage it was important for us to use a very simplistic aesthetic design, so users could focus more on the UX and layout rather than visual design.[4] We then asked some KTH students to test out using this preliminary version, by explaining the basic functions of the app and observing them as they tried to use it themselves (Fig 3). We learned a few things from this:

- The placement of the tiles on the app was important. "New Recipe" needed to be placed to the left of "New Shopping List" since it makes more sense to first create a recipe, then a shopping list. "Hierarchies of tasks" are an integral part of app navigation architecture to consider.[5]
- We needed to reduce the number of functions further. Originally we wanted to include functions to create a random shopping list of recipes, but this complicated the app. We decided it may be best to only include four main functions: "New Recipe", "New Shopping List", "My Recipes", and "My Shopping Lists".
- We needed to make it very clear what the difference between a "Shopping List" and a "Recipe" was for our app to be clear. At the time we thought a brief tutorial might be a solution to this?
- It might be useful to be able to specify how many people each recipe was for, and to be able to scale this when creating a shopping list.

Observations 2: Paper Prototyping in ICA

After making some revisions to our plan as listed above, we redrew our basic menu system, and then went out to ICA at Kista Galleria to make some observations of people as they were shopping and conduct some interviews. The writings of Holtzblatt and Beyer convinced us of the importance of "[immersing ourselves] in the life of individual users through field visits… to show a big picture of the whole market".[6] Additionally, by simply watching people "while they engage in their activities, then people do not have to articulate their practices".[7]

Some of our main observations at ICA included:

[1] Fling, 2009, p.67
[2] Holtzblatt and Beyer, 2014, pp.67-8
[3] Holtzblatt and Beyer, 2014, p.3
[4] Newman and Landay, 2000, pp.263-274
[5] Fling, 2009, p.85
[6] Holtzblatt and Beyer, 2014, p.1
[7] Holtzblatt and Beyer, 2014, p.11

- While we saw one or two people using physical shopping lists, there were far more people with their phones out at the supermarket. However, this was not necessarily for checking what to buy (some were playing "Pokemon Go" or taking selfies).
- While the supermarket did provide some devices for scanning items and adding them to some kind of digital shopping list, we did not see many people using this.

We also conducted two interviews, one with a middle-aged man (interview 1), and one with a young man and young woman (interview 2). We asked the same questions as before, and gained the following responses:

Interview 1

- Did food shops around 3 times per month.
- Normally decided what to buy in his head, as he was at the supermarket.
- Normally picked the easiest, fastest things to cook.
- It would be useful to know where in the shop items were located.

Interview 2

- Did food shops more than twice per week.
- Buys what is needed for the few days immediately in the future.
- Often decides what to buy in store or before going out on the food shop.
- Simplicity would be very important for an app to help with food shopping.
- Pre-installed recipes would be a great way to save time for users.

In summary, we found that the people we interviewed did not make shopping lists. They decided what they wanted to buy in the store. However, they would like help with their shopping if it would make their shopping easier and faster. We derived the themes as simplicity and helpfulness.

With this information, we concluded that it would be very important for our app to work while people were on the go, as this was often when people decided what to buy. This was one aspect which later influenced our decision to make a native app, since users may not always have an internet connection if they are out at the supermarket.[8] Additionally the app should be incredibly simple to understand and use, and should if possible already include some pre-installed recipes and shopping lists, so that the user can instantly start using it before even adding their own meal plans.

Assignment 2

Second Paper Prototype - MarvelApp

Considering the user feedback gained during our initial paper prototyping in Assignment 1, we started work on a more in-depth paper prototype. We redesigned our app using software called MarvelApp, which enabled us to create a more aesthetically pleasing paper version of our app. As can be seen, we were keen not to overload the user with too many options per page, as Fling considers this poor UX design.[9] We also were intent on using bright, contrasting colours in our user interface, since this increases outside readability (as well as giving the app a playful aesthetic).[10] Figures 4-7 show screenshots of a few of the intended pages of our app created on Marvel, to give an impression of what we were testing..

Having made this site map, we decided that we would interview fellow students from KTH. Our plan was to give users very little explanation regarding how our app worked, but instead to just ask them to create a

[8] Fling, 2009, p.149
[9] Fling, 2009, p.86
[10] Fling, 2009, p.133

recipe and a shopping list and see how they responded. We then asked them various questions about our app (listed below), as well as recording their clickstream as they navigated through the app.

The questions we asked users were: Do you understand how the app works? Would you know what to do if asked to create a shopping list/recipe? What didn't you like about the app? Are there any changes you would make to the aesthetics of the app? On a scale of 1-5 how easy did you find the app to navigate?

From the interviews we conducted, we got the following general results:

- Everyone we interviewed understood easily how the app worked, rating it at 5 out of 5.
- People liked the simple aesthetic design and user interface.
- We needed to make changes to the layout of the new shopping list page so that you could alter how many people each individual recipe was for (rather than change this for all the recipes at once).
- It would be useful to have an option to add individual ingredients to a shopping list, without needing to create a new recipe to do so (e.g. if you wanted to add a single banana to the shopping list).
- The Help page was not really necessary, as users never clicked on it (and besides, our app should be so self-explanatory that a tutorial is not required. Holtzblatt and Meyer suggest making "product use so direct that there's nothing to learn").[11] This goes to show that an important part of the development process is knowing when to drop ideas if they are unsuccessful.

JQuery Mobile Prototype

Having conducted these interviews, we were now ready to begin creating our web prototype, taking into account suggestions which were made. We decided to use JQuery Mobile as a starting point for us to begin developing our app, and so our prototype makes use of the JQuery libraries and CSS code. In order to be able to perform adequate tests with this prototype, we decided beforehand that we would need to:

- Implement a functioning site map, with a suitable aesthetic for our product. It should be simple to navigate and clear how to get between pages.
- Create mock-ups on these pages of the forms which we would use for our app. These forms will not necessarily be able to save data at this point in the development process, but should make it clear to the user how such a form would work.

With these goals in mind, we set to work on creating the html file which would act as the backbone of our app. Within this html file, we made five separate pages, which served as the "Main Menu", "New Recipe", "New Shopping List", "My Recipes", and "My Shopping Lists" pages:

- **Main Menu: (Fig 8)** We ensured that navigation between pages was simple by using the Main Menu as a central hub, from which you can access all the other pages easily (as well as also being able to return to the main menu with just one button click).[12] To make our design more aesthetically pleasing, we created .png images to use as buttons for accessing other pages, as well as for the company logo. Although our app admittedly requires a lot of text input, the use of icons for these buttons may make things easier for people with poorer eyesight or reading abilities.[13] The JQuery toolkit also made it simpler for us to create a more pleasing aesthetic for our app, particularly with the colour scheme, buttons and toggle sliders.
- **New Recipe: (Fig 9)** The focus on this page was to make it easy to type in the ingredients one by one in a table. The user gets a field where to type in the name of the recipe and three rows of ingredients to start with. Each row contains three fields: Ingredient name, quantity and type. We

[11] Holtzblatt and Beyer, 2014, p.10
[12] Brian Fling advises us to "limit users' options" when it comes to app navigation architecture, to ensure that users don't get lost: Fling, 2009, p.97
[13] Chipchase J, 2008, pp.81-2

also made sure that the user could easily add multiple rows and remove those that were not desirable. At the end of the page there is an optional field to type in the method of the recipe if that is desirable and of course a button for adding the recipe to "my recipes".

- **New Shopping List: (Fig 10)** In this page, users can create their own shopping lists by choosing from existing recipes. Also, the number of people for the selected recipe can be manually added in order to calculate the amounts of ingredients. Additionally, some individual items are allowed to be added at the end of the page (same as in New Recipe page) if they were forgotten to be included in the recipe.
- **My Recipes: (Fig 11)** This page works in a similar manner to "My Shopping Lists". At present, we use collapsible boxes to denote Recipes/Shopping Lists, which will show their content when clicked upon. These can also be deleted at the press of a button. In a future version of these pages, we will also implement a means of editing these collapsibles, although this was not possible given time constraints.
- **My Shopping Lists: (Fig 12)** See "My Recipes" above, since the same applies to "My Shopping Lists".

To be able to store data we decided to create a server on one of our computers and then use PHP to be able to send and retrieve it.

Web Prototype Testing

Once again, we conducted user interviews with our web prototype, to gain insights into how to develop it further. What follows are accounts of these two user interviews:

When we tested our web-based app on our first test subject, we saw that the app was somewhat complicated to understand. However, we understood that this was due to it being an early, incomplete version (and indeed, we are even advised that "a complete design is not necessary to start this validation process").[14] When we explained in more detail how the app should work, our test subject understood how everything worked and gave us positive feedback.

In the second interview, we talked more about what the intention was with the app before the person tested it. The result was a more fruitful conversation and we received several comments on how we could make the app better (listed below).

In summary, we needed to work more on the app's features and appearance and create a way to save data and retrieve it. We can summarize the interview feedback as follows:

- The layout of the create recipe page is the most important one to get right, as it is here where the user has to input the most info.
- Font size is in general too large, and distracts from where you have to input info.
- There needs to be a method for adding individual items to shopping list (this is something we were already wanting to do, it just highlights the importance of this feature).
- There needs to be a popup confirming when recipes/shopping lists have been created
- Some of the layouts need tweaking, like removing the unnecessary 'add it' on one of the pages, but overall good.
- We need to only allow integer values for quantities of ingredients.
- We need a separate box for specifying units (maybe a dropdown box for g, kg, dl etc.)

Moving forward with a mobile web app?

We considered all the feedback we received to be of great importance, so we developed our prototype (and indeed all future versions of the app) so that they included these suggestions. Now we had a much more functional and stylish app. The web app is also now available through the web and can be used by

[14] Holtzblatt and Beyer, 2014, p.67

users on their own devices. Were we to move forward with the web app, we would need to make the pages functional and implement the PHP solution mentioned above. Our decision between web and native is detailed later in this report.

<u>Assignment 3</u>

Android Prototype

Now that we had completed our web prototype, we had a model with which we could begin creating an android prototype using Android Studio. We decided that we would split up the work so that one group member was working on one of the 4 main pages or our app: Tom implemented “My Recipes”, Edward implemented “My Shopping Lists”, George implemented “New Shopping List”, and Jing implemented “New Recipe”. We knew that at this stage we would not yet give these pages functionality with use of a database, although we worked on the assumption that user data would be stored locally using an SQLite database. Images of the Android prototype can be seen in Figs 14 - 19.

Here is a brief overview of how we decided to implement the respective page we worked on:

Main Menu: This activity acted as the hub for our app, similarly to in our web prototype. It was the parent activity to My Recipes, My Shopping Lists, New Recipe, and New Shopping List. As can be seen in the photos, it contains 4 buttons, which when clicked will initiate the corresponding activity. Once we have set up our database, we will decide how exactly data will be sent between these pages.

My Recipes (Tom): For the “My recipes” activity the thought was to gather all recipes in an easy and accessible way. The idea was to have a scroll view where all recipes lined up in two columns. To manage this, a set of cards were set up with a picture of the actual meal indicating the recipe. When a card was pressed, a list of info would appear and show details of the recipe.

My Shopping Lists (Edward): For the “My Shopping List” activity, I made it so that this page would contain an empty table within a scrollview, which would be populated with data dynamically when the activity is created. Each row of the table contained the shopping list name, a button to view it, and a button to delete it. The view button takes us to another activity called “My List View” which displays the shopping list name as well as the contents of the list within a scrollview (these Strings are both provided for the new activity through intents). The delete button removes the given row from the table.

New Shopping List (George): On “New Shopping List” activity, I used an edit text field for the name of the shopping list and then one Scrollview containing edit text fields for the user to input relevant recipes and quantities. Below this, we have an area for adding individual items to the shopping list, with two edit text views, a spinner and a button. The first edit text is used to enter the name of the individual item that you might want to buy. The second edit text is used to enter the quantity of this item, this edit text only accepts integer numbers. Lastly we use the spinner to select the unit for our specified quantity. Once we have added all the details needed for the individual item, we just press the button next to the spinner which will add it to a table below. Each entry on the table has a remove button for removing entries. Finally, there is an edit text for adding an optional comment. After entering all the information, pressing “Create New Shopping List” will save the shopping list on a local database.

New Recipe (Jing): The “New Recipe” page contains a text editor on the top where you can enter the name of your recipe. Below, there is a Scrollview for recording all the ingredients and quantities needed for the created recipe. Users can also delete any ingredient by clicking on the delete button of that table row, or add another ingredient if needed. Additionally, a multiline text area is given at the bottom of the page, where users can write the method of the recipe if they want (optional functionality). After all necessary information is written for the recipe, users can click on the “Add recipe to My Recipes!” button to save their recipe.

User Feedback

After we had joined all our work together in the prototype, we performed some interviews to see what users thought about our app (Fig 13). Like before, we gave them a simple task to try to create a new recipe and a new shopping list, then observed as they attempted the task (noting anything they found difficult with our design). We had read that it might be more useful to let these interviews develop organically, although we still considered it important to ask if there was anything they would change with our app.[15] Since we had not implemented the database yet in our design, it was difficult for us to really test the functionality of our app. However, some of the feedback we gained included:

- Pages need to be more consistent (e.g. My Recipes and My Shopping Lists should look similar and have similar functionality). More generally, we should have more stylistic unity between all the pages.
- The use of photos to describe recipes might make it difficult to determine between recipes without some additional text.
- An edit button on the My Recipes and My Shopping Lists page would be useful.
- New Recipes page needs to be scrollable (Users often tried to scroll the page, but nothing happened).
- Some minor bugs need to be fixed, such as the forms randomly deleting user input.
- Could try to make the aesthetic design more interesting if possible, rather than just using the standard Android theme. That said, it is worth noting that having quite a standard layout could also work to our advantage: using the generic Android aesthetic may cause users to associate the same Android level of quality with our app.[16] As a group we later decided that our main priority should be usability and information layout, rather than the thematic design, since these are often what the user notices more.[17]

Assignment 4

Deciding whether to do web or native

After we had a working web based application we started to talk about doing a native app. We concluded that a native app is the best decision for the following reasons: The application must be as easy as possible. It should be easy to access and it's visual design must be interactive in a natural way. The app must be responsive and the content easily accessible. When we talked about how one wants to interact with the application, we concluded that a native app will be both easier to use than the web application and faster. First, because of the tools provided by the android studio makes it easier to develop an easy app and secondly that the number of html requests could be reduced which makes the response faster for the user.[18] We also concluded that we will reach more users with an native app because the users of web based sites is just approximately 10% of the users of mobile devices globally .[19] It would also be helpful for our business model, as it is possible to charge for native apps.[20]

Post-Assignment 3: Setting up the database before starting on the mashup

Before starting on our mashup, we first had to make sure that our app was in a functional state for people to effectively use it - that is, to store user inputs in a database. What follows is a brief description of how we set up this database:

[15] Holtzblatt and Beyer, 2014, p.13
[16] Fling, 2009, p.119
[17] Fling, 2009, p.107
[18] Souders, 2007, Ch. 3
[19] Jobe, 2013, p.27
[20] Fling, 2009, p.79

From the many database options we found, we went with SQLite. We chose SQLite because it is lightweight and easy to use as it is discussed in the referenced research paper.[21] Also, since each SQLite database is an integrated part of the application it is associated with, we are able to reduce latency (thus improving UX).[22]

We created a database named “ShoppyDB” along with some tables where we are going to store our data. We have 6 tables, “Recipes”, “Ingredients”, “ShoppingLists”, “ShoppingListItems”, “ShoppingListRecipes” and “Options”. Due to the fact that it is not possible to save arrays on SQLite we had to make the “Ingredients” and “ShoppingListItems” tables which serve as arrays separated by the first column which is the name of the recipe or the shopping list to which the ingredients and items belong to.

Each table serves a unique purpose. “Recipes” holds all the basic information about the recipes like name, number of people, and the recipe method. In “Ingredients” we store all the ingredients for each recipe. “ShoppingLists” contains data similar to “Recipes” but about the shopping lists (e.g. name and comment), while “ShoppingListItems” contains all the ingredients and individual items from all the included recipes. “ShoppingListRecipes” holds all the information about the selected recipes which are included on the shopping lists such as name of the recipe and the amount of people we want to cook it for. Finally we have the “Options” table which for now serves only one purpose, holding a Boolean variable whether the program has been executed for the first time or not.

Brainstorming ideas for mashup

Now that the core of our app was functional, we were ready to start work on making a creative mash-up for it. Since none of us had any previous experience with utilising online APIs in this way, we quickly decided that it would be best if we first started to learn the basics using HTML, before trying to implement something into our Android app.

We decided that each team member would do some experimenting with possible API mashups, and then we would come together and decide which mashup we would like to implement into our final Android app. We agreed that some type of visual “Consumer Mashup”, as defined by Viedma, would be most appropriate for our customer orientated design.[23] As a result, we came up with various mashup prototypes using various different methods, which are now listed.

Tom used Node.js to execute an html mashup of the “Leaflet Map” API and “International Space Station” API, which showed the continually updating location of the ISS on a map, while also providing inspirational life advice in a popup window when the ISS icon was clicked. Ed also created an html mashup, instead using the “Google Maps” and “Flickr” API in order to allow users to check out what other people are cooking near them, and get inspiration for what recipes they might like to add to our app. This used the precise geolocation of the user, and placed the first 50 Flickr images (taken in a local radius and tagged as “food”) on the map. George also experimented with data found from JSON files able to be shown on a “Google Map”, one involving a remote server with criminal records and the other containing many recipes. Finally, Jing did some experimentation with the “Google Maps” API to find our what would be necessary for showing and updating user location on a map.

After discussion, we decided to implement the Google Maps / Flickr mashup for our final Android app, since this was most in tune with the theme of our app:

Final mashup design: Google Maps / Flickr mashup (Android)

Transferring the HTML mashup to Android proved more complicated, as we now had to ensure that the Android device made all the proper permission checks (such as accessing the internet, geolocation etc.)

[21] Bhosale, Patil and Patil, 2015, p.884
[22] Meier, 2018, p.253
[23] Viedma, 2010, p.21

before accessing the relevant APIs. Even after getting these permissions, we were also required to understand the life-cycle of classes extending AsyncTask to make sure that the app did not try to display data before it had been loaded from the internet. This is because Android does not want your app to stop functioning while accessing data, so encourages performing tasks asynchronously in the background. This is for good reason, since "in practice, users will notice input delays and UI pauses of more than a couple of hundred milliseconds."[24]

We began by setting up a Google Maps activity and giving it permission to display the user's geographical location with a marker. We implemented this by creating a function called getLastLocation(), which would be called during the onCreate() of the activity. This method, getLastLocation(), would check all relevant permissions were granted, before saving the device's latitude and longitude into two global variables.

Having obtained the latitude and longitude, we created a button on the screen which, when pressed, would call a method called locate(). This location method would use the previously stored latitude and longitude and add a marker to the map in the user's exact position. The locate method also executed an asynchronous class RetrieveFeedTask, which shall be discussed shortly:

In order to implement the flickr API into this mashup, we needed to use a few AsyncTask classes. The first of these, RetrieveFeedTask, included a protected method doInBackground() which would call on the Flickr API (in a similar manner to in HTML, as described above) and return a JSON string relating the the photos we wished to display. We decided to implement using JSON rather than XML, since the JSON parser is around 10 times faster, providing a faster and less power consuming user experience.[25] Then, the onPostExecute() method of the class would parse through this JSON String for photo URLs and location data, and for the first 50 photo objects would call upon another asynchronous class called RetrievePhoto to obtain the actual image.

RetrievePhoto had a similar life-cycle to RetrieveFeedTask. It's doInBackground() method would use the URLs parsed during RetrieveFeedTask to construct bitmap images. Then, the onPostExecute() method would use the photo coordinates parsed during RetrieveFeedTask to add a marker to the map, with the bitmap photo as its infowindow.

After both of these asynchronous classes had been executed fully, the map now contained a marker indicating where the user was standing, and 50 other markers of nearby positions containing infowindows with pictures of food from that location. Since "in a mobile environment the time taken to get an initial location can have a dramatic effect on the user experience," we made sure during testing that our app could find the user and photo locations practically instantaneously.[26] Please see Figs 27 and 28, which show the activity after the button is pressed, and after one of the markers is pressed.

User testing and feedback

Having created our mashup, we went out to test it on people. Since our app was now fully functional, we also gained feedback on the app as a whole. Our methodology was to provide the user with a physical copy of a recipe, ask them to add it to the app and then create a shopping list. We then asked them to have a go at using the mashup, with no instructions as to how it worked. Our feedback was as follows:

- Photos shown on the mashup were not always relevant in the mashup (that is, they showed images that were not strictly speaking food related). This issue was later solved by better use of tags (e.g. "breakfast", "lunch" and "dinner" work better than "food", since they are more specific).
- Privacy concerns. Since we cannot properly ensure that the photos included in our app only show food, there might be some privacy concerns surrounding our mashup if images of people are

[24] Meier, 2018, p.345
[25] Viedma, 2010, p.32
[26] Meier, 2018, p.527

shown. This is an issue which could potentially be resolved by implementing our own database of photos, where users would explicitly give consent for their photos to be used. Our concerns with the security and privacy concerns raised by mobile mashups are backed up and discussed in more detail by Maximilian in *Mobile mashups: Thoughts, directions, and challenges*.[27]

- The photos don't update frequently. You get the same photos if you try the search multiple times in the same location. This may make the mashup a bit boring and predictable. Goussis and Foukarakis encourage getting to know the lifespan of your mashup, and it appears we need to find ways to give our mashup a longer lifespan when used in a single location.[28] Perhaps an option to search in other geographic locations could remedy this?
- Overall, people didn't think that the mashup was especially useful, although some users stated that they could see how there was a market for "food porn" (that is, viewing images of food that are irresistibly tasty).

Feedback about the app as a whole:

- Slider on the "New Recipe" page is slightly hard to use, so perhaps it should be made bigger. This was later fixed to make user navigation easier.
- Overall, the app was very simple to use. One of our users claimed that they were "entirely new to mobile phones" yet were able to add a recipe to the app, view it and put it in a shopping list.
- The ability to download recipes from an online database is something which most users mentioned they would like to see in a future iteration of the app. Indeed, this is something we would possibly be able to monetise, to make our app more feasible for bringing to market. For example, users could download the app for free, but might be required to pay to download a selection of premade recipes.
- Some fonts should be changed on "My Recipes" and "My Shopping Lists" to make them more interesting.
- Might be good if you could specify your food preferences when you first load the app, so that it can suggest recipes you might like from an online database.

Project Evaluation

Our app was now in a complete and functional state, and we were ready to present it to the class. Following this presentation, we gained the following oppositional arguments from one of the other groups, which were as follows:

- "The overall idea is great! Definitely we would love to use the App, especially the idea of creating a shopping list is really useful!"
- "For the further enhancement, do you think it would be possible to get recipes online so users don't have to input the recipe ingredients by themselves? Because sometimes people might be lazy to input all the ingredients on their own to create a new recipe."
- "For the shopping list, I would suggest if the design could be more reader friendly. Maybe using a checkbox would be more user-friendly."

This feedback ties in strongly with some of the previous user suggestions we had obtained, making it clear to us that were we to move further with the project it would be necessary for us to allow an option to download recipes. As shall be discussed shortly, this feature could become part of our potential business model. As for the readability of the shopping list, this is an easy thing for us to develop in our design (indeed, we have subsequently changed the design of these pages by making the font larger and more like handwriting). Images of our final app with this feedback are shown in Figs 20 - 28.

[27] Maximilien, 2008, p.600
[28] Goussis and Foukarakis, 2010, p.1

Summary of UX, Development, Ready-to-Market

UX:

Throughout the project, we aimed to make all our design decisions motivated by user experience. Examples of this are listed in the preceding report, but a few examples include the use of a few large buttons for simple menuing, contrasting colours for ease of readability, and limiting of navigation options to prevent the user getting lost.

Upon reflection, the aesthetic design of our final app is not especially innovative, since we have tried to make everything as simple as possible. That said, through doing this we have made an app which is very intuitive to use - notably, during some of our final testing we found that users with very little prior mobile experience were able to successfully utilise the functions of our app without prompting or the use of any tutorial. This, as well as the other positive feedback we received from users, confirmed to us we had met one of our stated goals to make an immediately usable application. We feel that in limiting user actions right down to the most crucial(create recipe, view recipes, create shopping list, view shopping lists etc.) we had created a simple and understandable User Experience.

One potential flaw in the UX, brought up in the presentation opposition to our app, is that our app requires the user to input lots of data, which can prove cumbersome. This is also something Fling advises against.[29] However, this was still a considered decision for us: some of the other shopping apps which we looked at, particularly the ICA app, did not allow user inputed recipes, which might mean that this feature could be a potential advantage in the market for us. Additionally, we could later add a paid option which would allow users to gain access to lots of ready-made recipes, as opposed to the more time consuming experience gained when the app with initially downloaded (see below).

Development:

As can be seen from our detailed description of our design process, we worked iteratively (a tactic encouraged by Holtzblatt and Beyer).[30] That is, we successively produced prototypes, tested them on other students/members of the public, and used their feedback to develop a richer product for the user. A few examples include our provision of an option to add single ingredients to a shopping list, including popups to give users feedback on their inputs, and always providing easy access to the menu from each activity (all of which were ideas we gained directly from user interviews). We were also able to figure out what didn't work well in this way, such as taking away the tutorial screen and the "random shopping list" option, which was deemed unnecessary by testers.

One of our challenges with creating an app with rich features was to do so without making it over complicated - after all, one of our central goals was to make our app as simple as possible. For example, we could have added various features to the "create recipe" page, like an option to define if the recipe was vegetarian or not, but felt that this might risk adding complexity to an idea which already required a lot of user inputs. We felt that creating a rich experience is more about providing depth and polish to the features you already have, than providing lots of shallow interactions.[31]

Ready to Market:

After a couple of hours searching for relevant information about the recipe app/websites market, the only recent market report we were able to find is from this year (2019),[32] but costs thousands of dollars to buy. It's produced by a market intelligence and data analytics firm by name HTF Market Intelligence Consulting. No other report have been found that gives us insight into the market, although we found a

[29] Fling, 2009, p.84
[30] Holtzblatt and Beyer, 2014, p.67
[31] Fling, 2009, p.67
[32] https://www.htfmarketreport.com/reports/1925194-global-recipe-websites-market

website, openpr.com, that says in a press release[33] that the “market is booming worldwide” with references to the report by HTF. From this, we could assume that there is indeed a market for our service but we can’t be sure how big it is. We would like to distribute our app through Google Play because it is the largest and most frequently used Android app distributor [34] and after a while, if there is people using our app, we would like to buy one copy of the report and discuss how to develop the service further.

Since the subject area we chose to target is such an important and widely affecting one, this of course means that a lot of competition already exists. To name a few similar apps: Yummly, Chowhound, Epicurious, MyTaste, ICA, Coop. We would need to therefore focus heavily on what makes our app more unique, that is its ability for the user to create or change recipes which none of the established services provide. From the feedback we gained during this project, we found that the main functions of our app (making recipes and shoppinglists) were something that people saw as helpful and helpful tools were something that everyone saw as something they wanted in their shopping experience. From that early feedback we have been focusing on simplicity and helpfulness which have shaped the app to what it is today. As for the ecosystem, since our app is not the most revolutionary of concepts, it is hard to see how it would greatly impact the current app environment unless we provided an incredibly polished product.

There are three main ways to monetize an application through Google Play: make it a pay-to-download app, include In-App-Billing, or to display advertising. Of these options, as has been alluded to at various points in this report, the In-App-Billing option might be the most preferable, since this can be tied in well with providing users with additional downloadable recipes. Meier notes that through this monetization method, the developer will split revenue with Google Play, gaining 70% of proceeds.[35] Given the pre-existing competition of shopping apps, making our app pay-to-download may not be effective, since these tend to be free to download (e.g. the ICA app). Advertisement based revenue would be a possibility for our app in its current build (given that we have not currently implemented an online recipe database), although we would need to ensure that the ads included were as unobtrusive as possible, so as not to impair our carefully designed UX.[36] Successful apps using this model, such as the Spotify “Freemium” service, are able to strike the perfect balance between providing excellent service while occasionally deploying ads (which simultaneously encourage the user to subscribe to a premium service).[37]

We would need to ensure that our app gained a good reception upon launch, since it often hard to recover from a poor start. As Meier notes, we would therefore need to focus on polishing the features which our app already provides, and provide honest descriptions of what our app provides on the Google Play store.[38]

<u>References</u>

Aidi L, Markendahl, Tollmar, Selvakumar, Huang, Blennerud, “Mobile Music Business Models in Asia's Emerging Markets”, in *12th International Conference on Mobile Business 2013, Berlin 10-13 June 2013*, 2013.

Bhosale S. T., Patil T. and Patil P., “SQLite: Light Database System”, in *International Journal of Computer Science and Mobile Computing Vol. 4*, April 2015

[33]https://www.openpr.com/news/1765228/recipe-websites-market-is-booming-worldwide-allrecipes-foodnetwork-genius-kitchen-thekitchn.html
[34] Meier, 2018, p.774
[35] Meier, 2018, p.780
[36] Meier, 2018, p.780
[37] Aidi, Markendahl, Tollmar, Selvakumar, Huang, Blennerud, 2013, p.6
[38] Meier, 2018, p.781

Chipchase J., “Reducing Illiteracy as a Barrier to Mobile Communication”, in *Handbook of Mobile Communication Studies*, Ed Katz, MIT Press, 2008.

Fling B., *Mobile Design and Development*, O'Reilly Media, August 2009.

Goussis L. and Foukarakis I., *Aspects and Challenges of Mashup Creator Design*, 14th Panhellenic Conference on Informatics, 2010.

Holtzblatt K. and Beyer H., *Contextual Design: Evolved*, Morgan & Claypool Publishers, October 2014.

Jobe W., *Native Apps vs. Mobile Web Apps*, Stockholm University, October 2013.

Maximilien E., “Mobile mashups: Thoughts, directions, and challenges” in *Semantic Computing*, IEEE International Conference, 2008.

Meier R., *Professional Android, 4th Edition*, Wrox, Aug 2018.

Newman M. and Landay J.,. *Sitemaps, storyboards, and specifications: a sketch of Web site design practice*, DIS'00, 2000.

Souders S., *High Performance Web Sites: Essential Knowledge for Front-end Engineers*, O’Reilly Media, September 2007.

Viedma C., *Mobile Web Mashups: The Long Tail of Mobile Applications*, KTH, 2010.

Figures

Fig 1

Fig 2

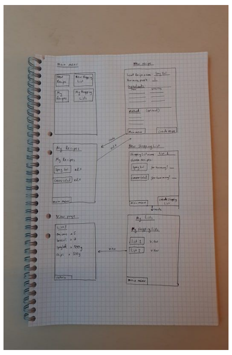


Fig 3

Fig 4

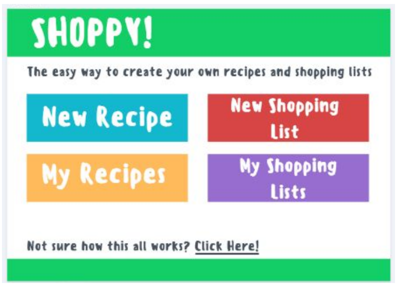

SHOPPY!
The easy way to create your own recipes and shopping lists
New Recipe
New Shopping List
My Recipes
My Shopping Lists
Not sure how this all works? Click Here!

Fig 5

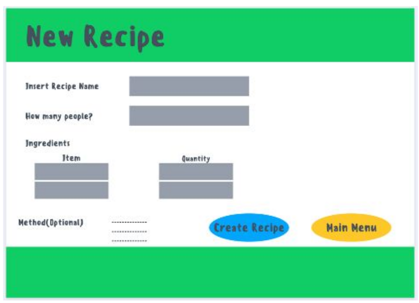


Fig 6

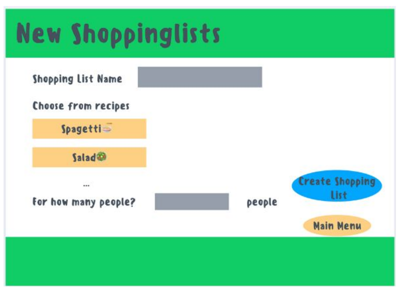
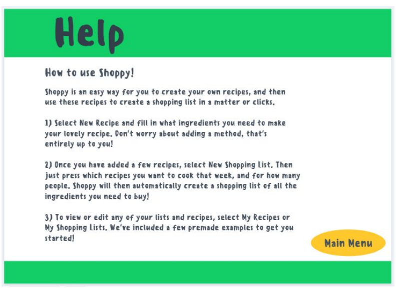


Fig 7



How to use Shoppy!

Shoppy is an easy way for you to create your own recipes, and then use these recipes to create a shopping list in a matter or clicks.

1) Select New Recipe and fill in what ingredients you need to make your lovely recipe. Don't worry about adding a method, that's entirely up to you!

2) Once you have added a few recipes, select New Shopping List. Then just press which recipes you want to cook that week, and for how many people. Shoppy will then automatically create a shopping list of all the ingredients you need to buy!

3) To view or edit any of your lists and recipes, select My Recipes or My Shopping Lists. We've included a few premade examples to get you started!

Fig 8

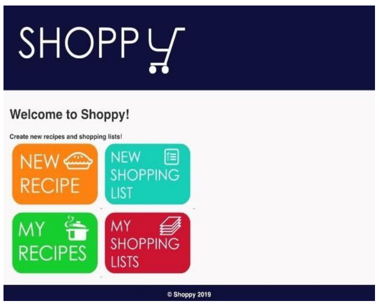


Fig 9

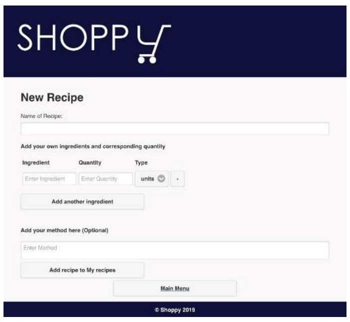


Fig 10

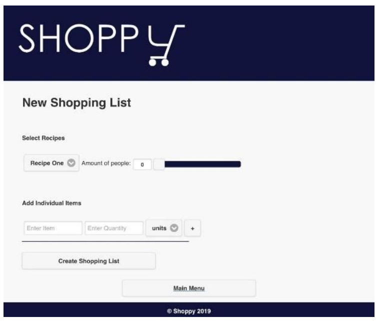


Fig 11

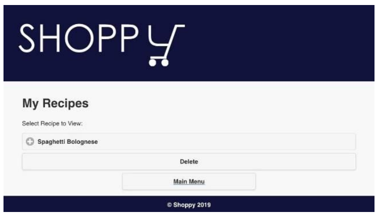


Fig 12

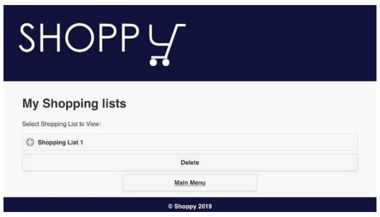


Fig 13

Fig 14

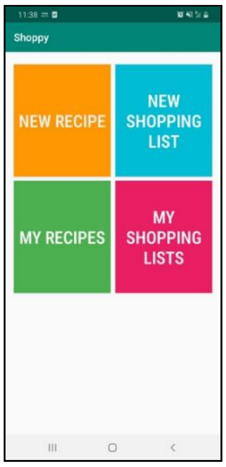


Fig 15

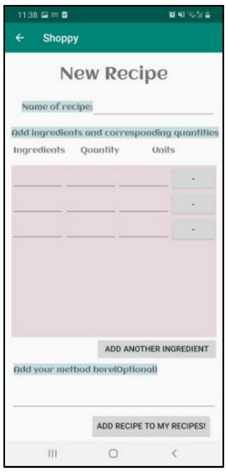

11:38
Shoppy
New Recipe
Name of recipe:
Add ingredients and corresponding quantities
Ingredients
Quantity
Units
-
-
-
ADD ANOTHER INGREDIENT
Add your method here(Optional)
ADD RECIPE TO MY RECIPES!

Fig 16

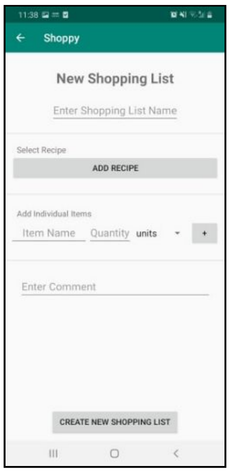


Fig 17

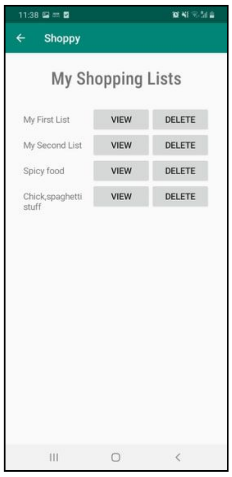
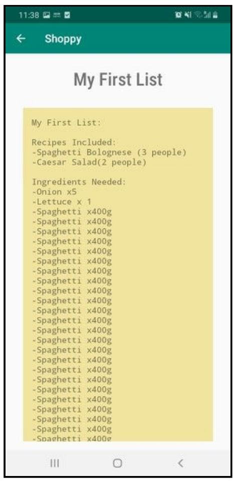


Fig 18



```
My First List:

Recipes Included:
-Spaghetti Bolognese (3 people)
-Caesar Salad(2 people)

Ingredients Needed:
-Onion x5
-Lettuce x 1
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
-Spaghetti x400g
```

Fig 19

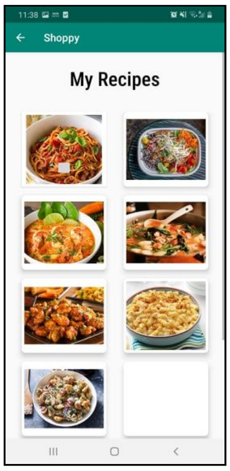


Fig 20

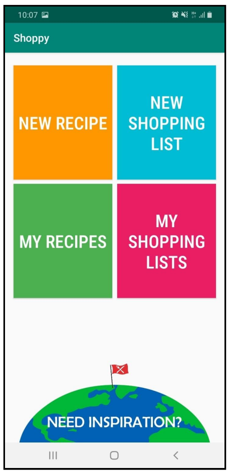


Fig 21

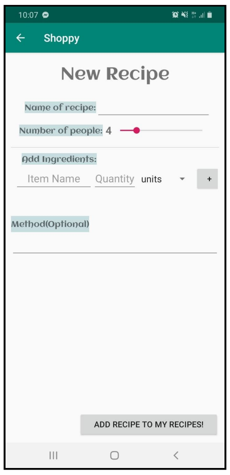


Fig 22

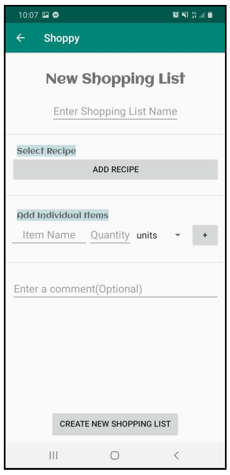


Fig 23

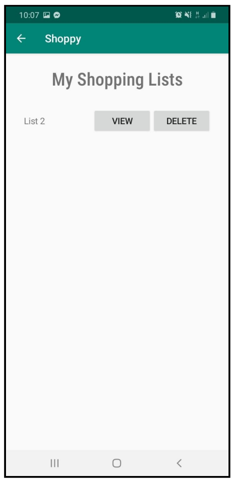
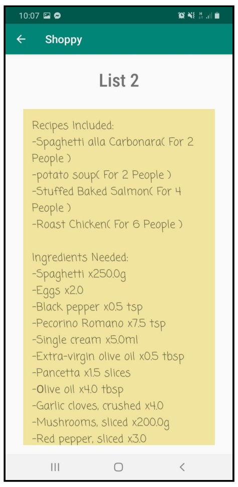


Fig 24



Recipes Included:
-Spaghetti alla Carbonara(For 2 People)
-potato soup(For 2 People)
-Stuffed Baked Salmon(For 4 People)
-Roast Chicken(For 6 People)

Ingredients Needed:
-Spaghetti x250.0g
-Eggs x2.0
-Black pepper x0.5 tsp
-Pecorino Romano x7.5 tsp
-Single cream x5.0ml
-Extra-virgin olive oil x0.5 tbsp
-Pancetta x1.5 slices
-Olive oil x4.0 tbsp
-Garlic cloves, crushed x4.0
-Mushrooms, sliced x200.0g
-Red pepper, sliced x3.0

Fig 25

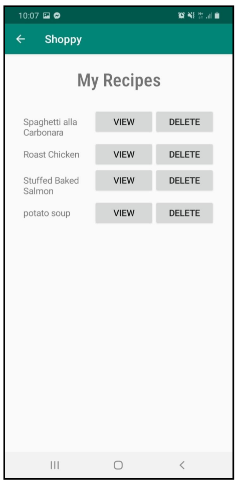
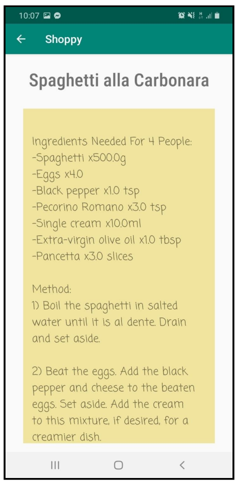


Fig 26

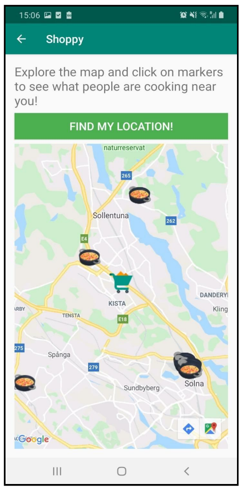


Ingredients Needed For 4 People:
-Spaghetti x500.0g
-Eggs x4.0
-Black pepper x1.0 tsp
-Pecorino Romano x3.0 tsp
-Single cream x10.0ml
-Extra-virgin olive oil x1.0 tbsp
-Pancetta x3.0 slices

Method:
1) Boil the spaghetti in salted water until it is al dente. Drain and set aside.

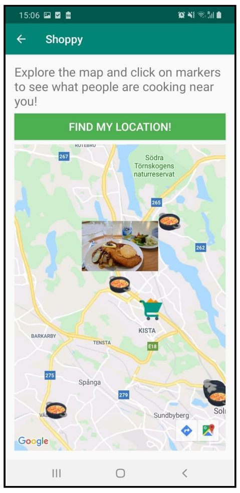
2) Beat the eggs. Add the black pepper and cheese to the beaten eggs. Set aside. Add the cream to this mixture, if desired, for a creamier dish.

Fig 27



Explore the map and click on markers to see what people are cooking near you!

Fig 28



Explore the map and click on markers to see what people are cooking near you!