5/4/2017

Cross-Platform Development – CMP3035M

CMP3035M

Chelsea Mcshane (MSH14537872)
UNIVERSITY OF LINCOLN

Table of Contents

List of Figures

Cross-Platform Development

1 Design

The following sections will give a brief overview of a cross platform mobile application developed using a variety of web technologies such as HTML5, CSS, Phone Gap, JavaScript and external libraries. After this a critical analysis of similar mobile applications will be discussed along with how these helped make specific design decisions towards making a different application that fills a gap in the market. Following this a user scenario will be discussed and the core requirements derived from that scenario. In addition, a walkthrough of the design process will be explained and evaluated and how the evaluations helped towards developing a better overall application experience.

1.1 The Strategy Plane

1.1.1 Overview

The developer chose to design and implement an application based on food recipes; more specifically, recipes for users that have special dietary requirements such a vegetarian recipes or recipes that exclude allergens such as dairy. This also included a feature to allow the users to add their own recipes so they could find all their recipes in one place; to be used wherever and whenever they wanted. The reason the developer chose this type of application is because a gap in the market had been identified when an analysis of similar applications had been completed, the developer found that none of these applications had this kind of feature; this will be discussed further in the next section.

1.1.2 Competitor Analysis

Recipe Book

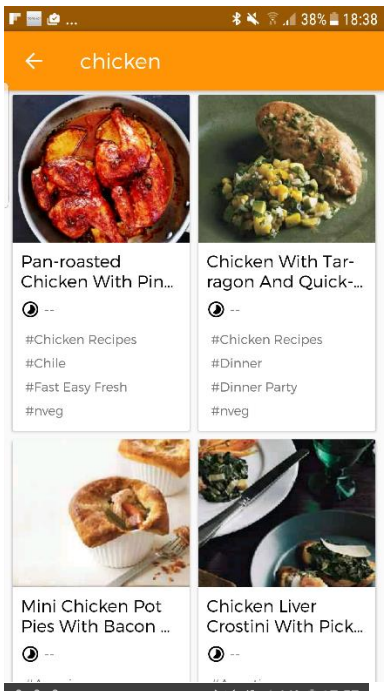
This application has comparable features to the Developer's application due to having a recipe search feature, this application allows you to search via tags so the target market they are aiming at are younger, technology enabled users. You search for a recipe or product such as "chicken" and then you pick one out of the filtered results, it then shows you a nice picture of the product you could make, and how many it serves. They convey a fresh, vibrant mood with their high-quality images of products.

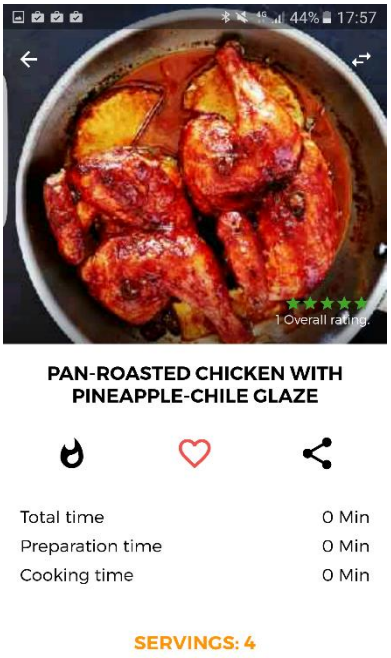
The best feature about this application is the ability to see the calorie content as well as other nutrient information on the recipe, this could be quite useful when users only want to have a certain amount of a specific nutrient. The developer especially liked

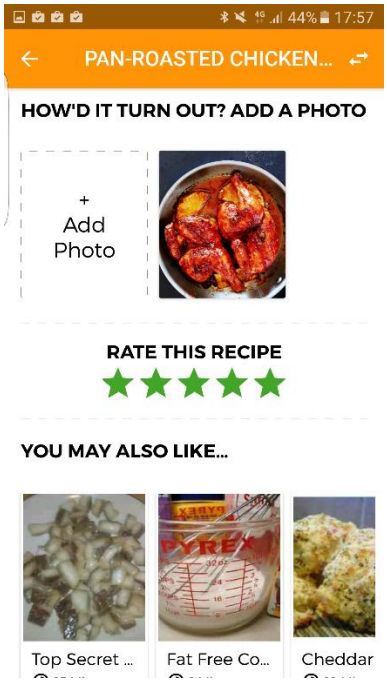
the idea of being able to take a photo of the recipe once you have cooked it to see how this turned out, however, to do this you need to setup an account.

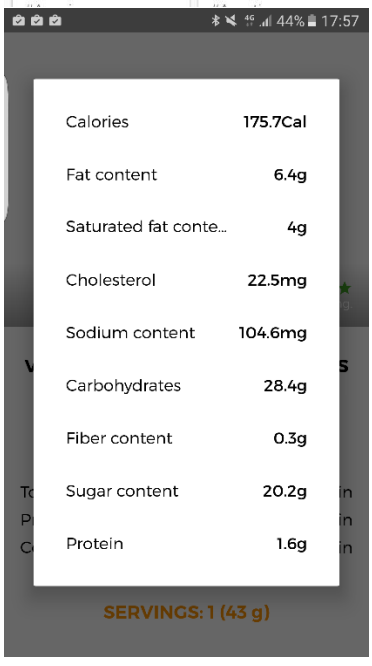
Even though the application is easy to navigate, the user experience is not so great, because of the use of high-quality images means that loading of the images takes a while to show or are not shown at all, another issue found was that some of the products do not show a cooking time, preparation time or total time which can be found to be quite frustrating and means that most of the recipes within this application cannot be used, see figure one.

1.1.2 Figure One





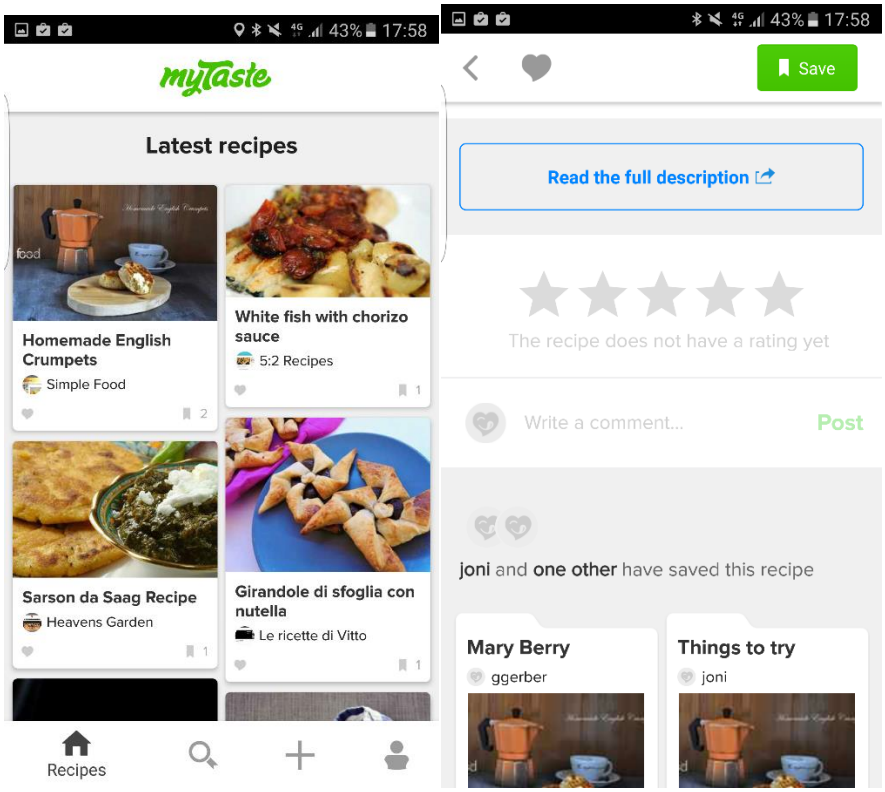


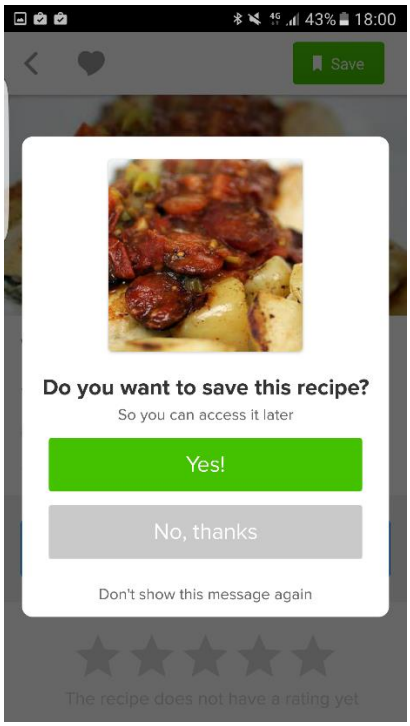


MyTaste

This application also has a search feature; however, this does not show the nutritional content that the aforementioned application had. The developer found that this application was hard to navigate, for instance once you have searched there is no visible button to go back to the home screen (you must press the back button on your device). They have implemented a blog type feature which allows you to add comments to recipes such as if you used less sugar etc. however this required an account. People with no cooking skills wanting to experience cooking for the first time would find this feature extremely useful. The application is aimed at multiple audiences such as skilled and non-skilled users. One feature that the developer found annoying was that a popup appears on every recipe you click asking you to save it, which gives a bad user experience, see figure two.

1.1.2 Figure Two





From the analysis of the above applications, the application to be developed identifies and fills a gap in the market by offering this to users who wish to be able to search for dietary and allergen specific recipes, something that none of the above-mentioned applications are capable of. Therefore, this application would cater to a niche market, any technology enabled users who want to cook diet and allergen specific recipes would find this application to be highly useful. This could be users finding out they are eating too much carbohydrates so want to eat a more balanced diet or they want to learn how to cook but have an allergy such a peanuts or fish.

1.2 The Scope Plane

1.2.1 User Scenario

From analysing the market a user scenario has been created which shows what core features the application may implement and how the end user will ultimately use the application.

Joanne wants to find recipes with specific dietary and allergen requirements for her and her fiancé, so she can try them out later. She loads the application and enters a search term "noodles", she then clicks the filter button, which then drops down to allow her to pick between a variety of diet options and allergen options. After she is finished applying filters, she clicks the save filters button then clicks search. The application shows a loading sign while it searches for the recipe requested, then a list of recipes along with any images if they are available are shown. Joanne can then browse these recipes, when she finds a recipe that she likes the sound or look of, she can click the recipe and this takes her to a page where she can view the image enlarged, the recipe name, calories, serving size, ingredients and instructions. She then clicks the recipe

page where she can add her own recipes to the application and these then show up in the recipe page to be viewed at any time. Joanne clicks to add her own recipe and this takes her to a page where she fills in the title, ingredients and method, she then clicks the save recipe button, then a notification tells Joanne that her recipe has been saved.

1.2.2 Requirements

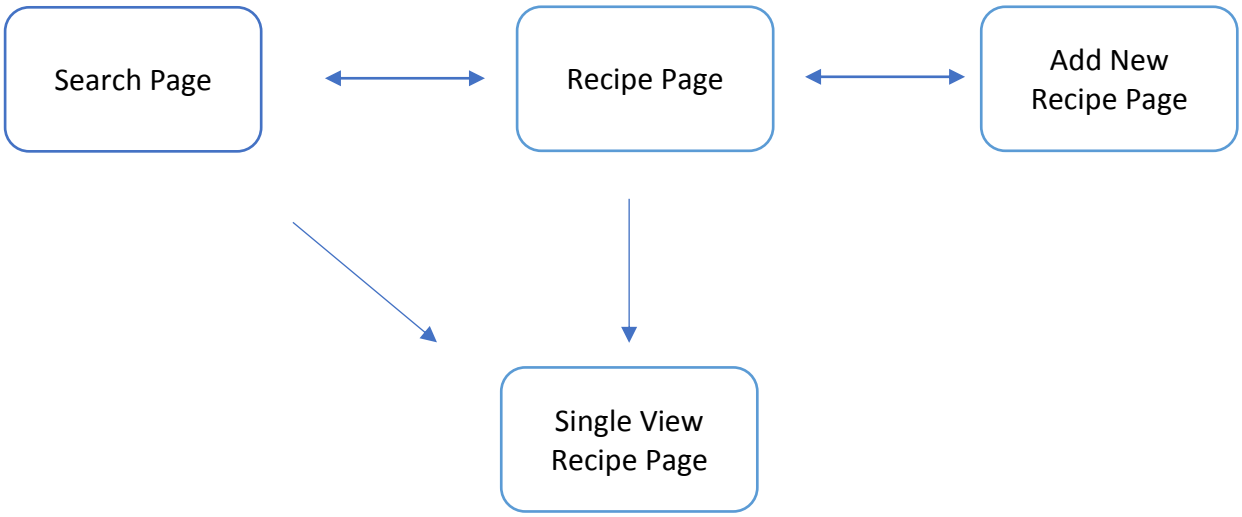
From the above scenario, the core requirements were derived for the application which are:

- Search for dietary and allergen specific recipes
  - Enter search term
  - Apply filters
  - View a list of recipes
- View each recipe
  - View recipe ingredients
  - View recipe instructions
  - View nutritional information about recipe
- Add own recipes and save to view later
  - Add recipe information
  - Save recipe information
  - View saved recipe information

1.3 The Structure Plane

From the derived requirements from the user scenario within the scope plane, the structure of the application was designed, see figure three. This shows how the user would reach certain pages. From the search page, the recipe page and the single view of searched recipes can be viewed and accessed. It also shows that the add recipe page can be accessed from the recipe page.

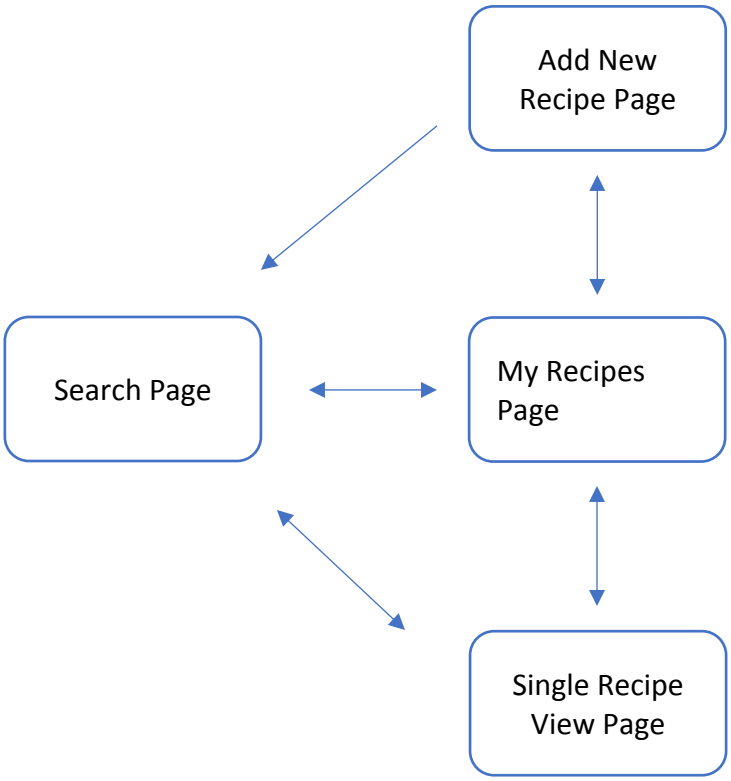
1.3 Figure Three



After the initial hi-fidelity design of the application had been designed, some changes were made to the structure of the application, this was because after observing two people interacting with the application, after going to the “add recipe page” they would then try and go to search for a recipe but this could not be accessed from the “add recipe page”.

In the next iteration of the design, the structure was redesigned so that the user could go back to the search or saved recipe page from the single recipe view or the add recipe view, see figure four.

1.3 Figure Four

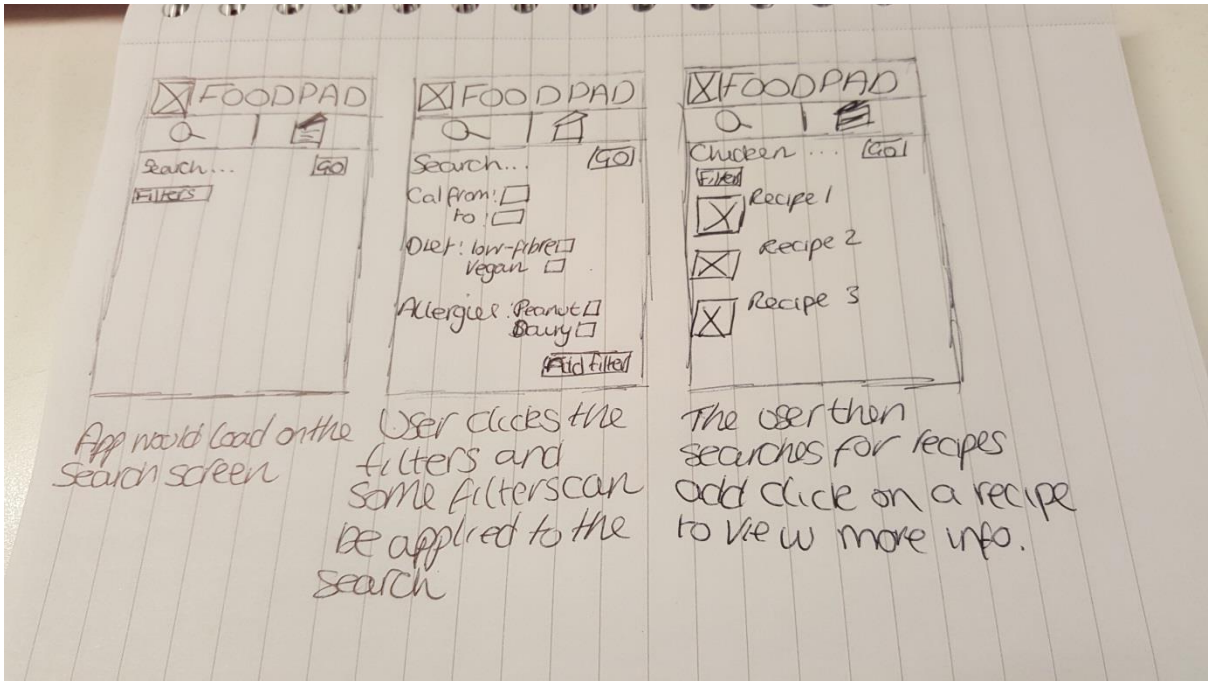


1.4 The Skeleton Plane

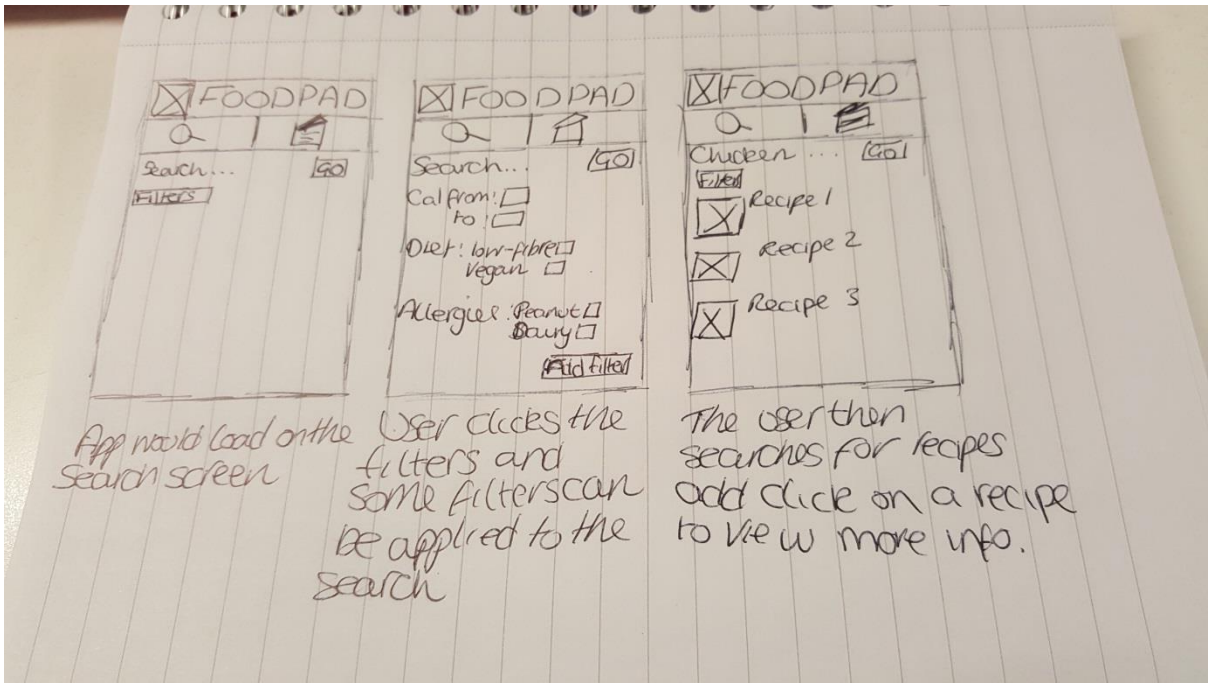
A low-fi prototype using pen and paper was developed, see figure five to figure seven. These show the basic actions and possible layout of the application. Throughout the design of the application the developer set up meetings with five potential users of the application to get their input and feedback on the design decisions being made. The low-fi prototype was then shown to the users to talk them through the core features that would be implemented and how they would use and interact with these features. The feedback from this meeting was mostly positive, but until a prototype is designed that allow the user to interact with the application it was hard to get a feel of how the overall user experience would be.

A digital skeleton was developed that showed the placement of features and after the second meeting where this prototype was evaluated, it was found that the users took a dislike to the name of the application so this was discussed and a name was agreed upon called “COOKPAD”, another digital skeleton was developed which shows the placement in more detail, see appendix one.

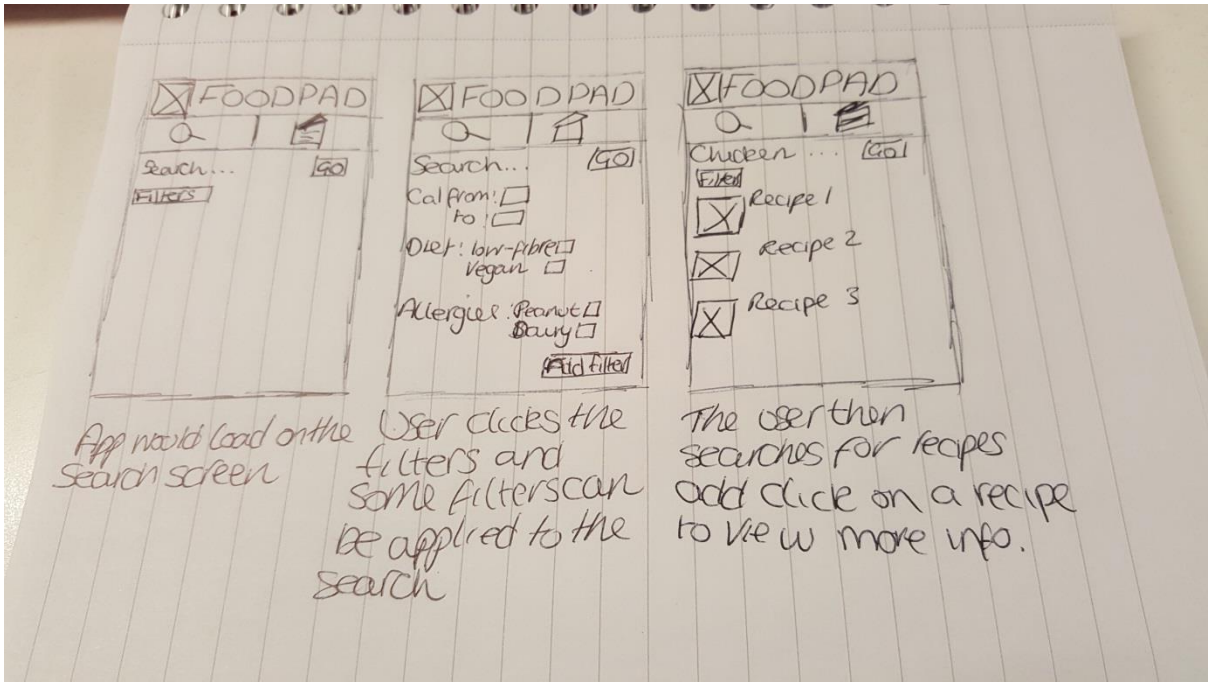
1.4 Figure Five



1.4 Figure Six



1.4 Figure Seven

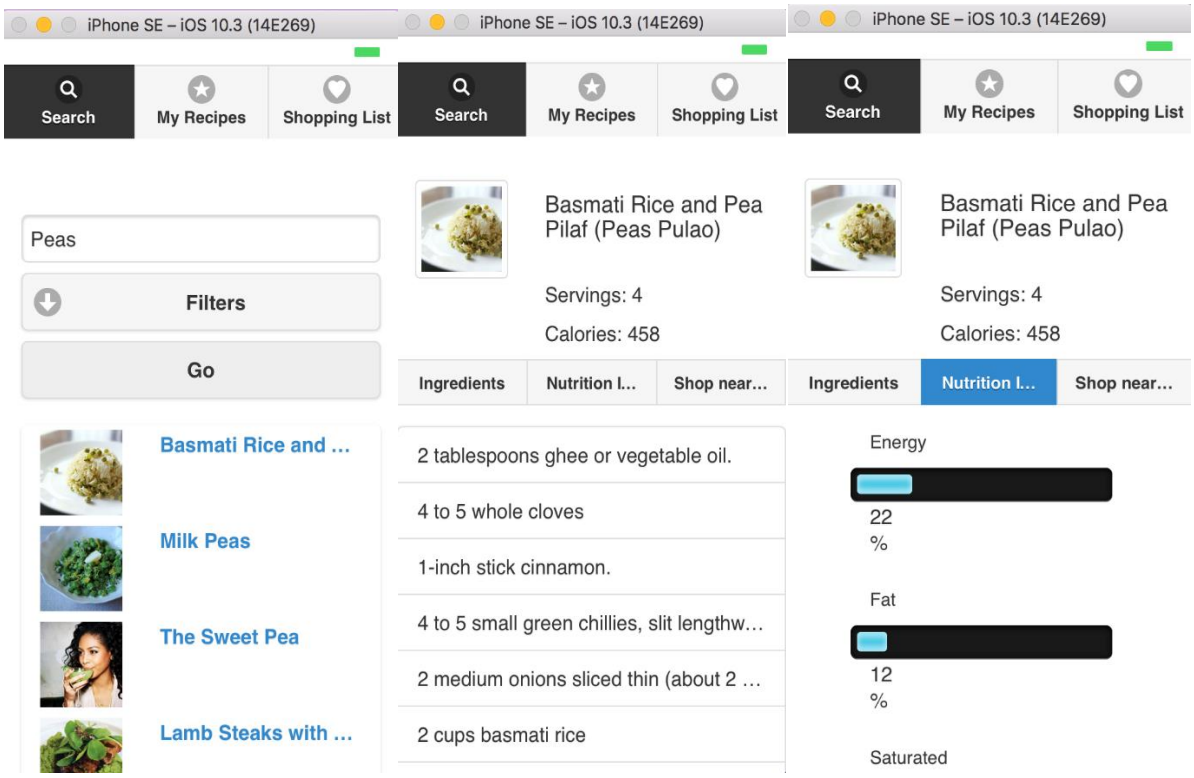


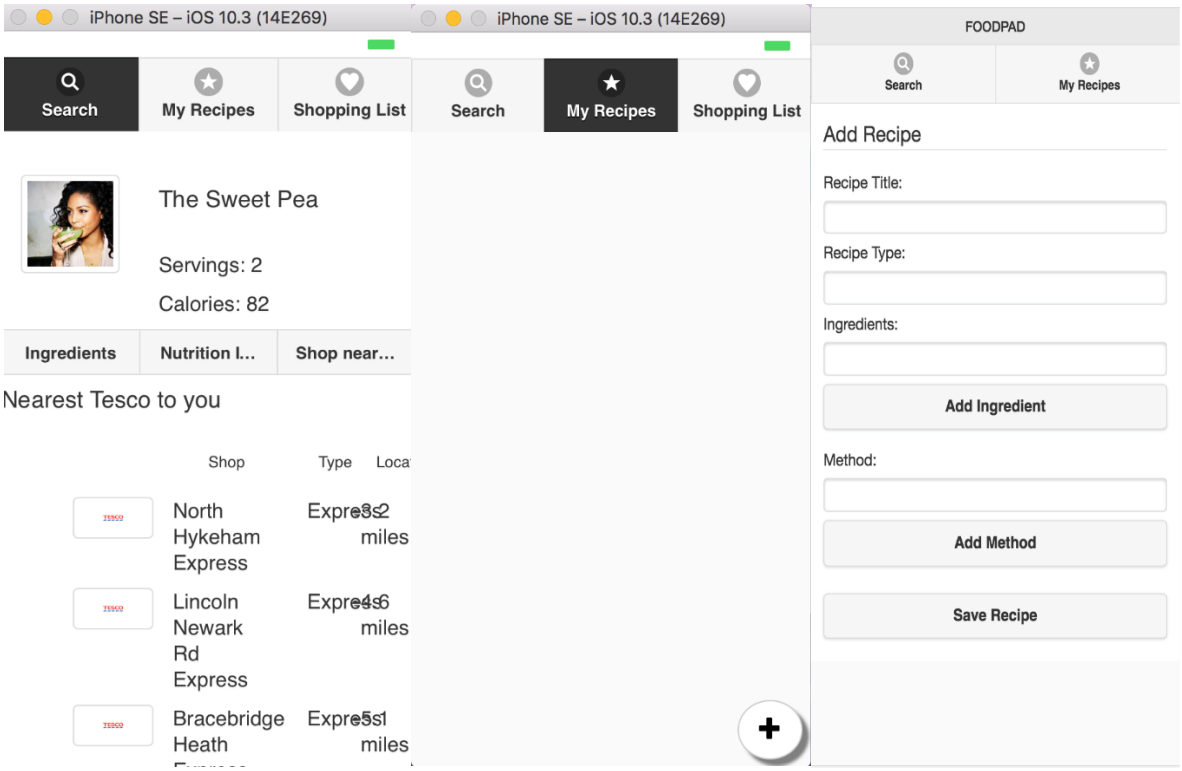
1.5 The Surface Plane

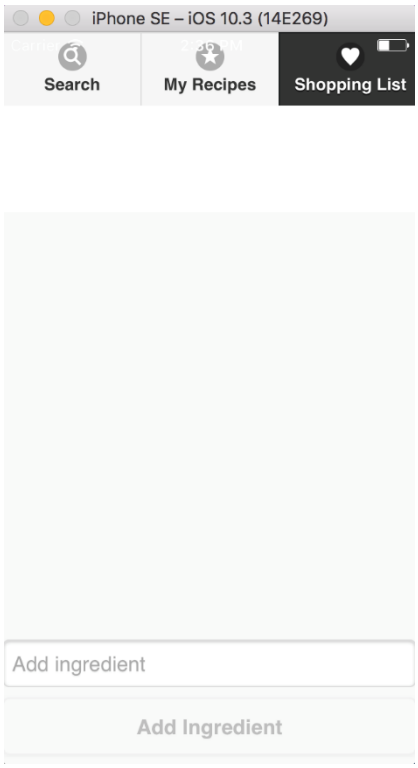
After this a hi-fi prototype was created, this was done with HTML5, CSS3 and JavaScript so the users could better visualise the actions and features of the application as well as get a feel for the user experience. In the third meeting two users thought that a shopping list should be added so that they could add items that they needed to, either off the recipes or unrelated items. Four users also said that they would like to be able to find the nearest shops to them so that they could go get the recipes if they were on the go using the application.

The developer changed the recipe view so that when a recipe is clicked, the ingredient list automatically loads and then the user can click to view the method or nutrients of that recipe, this enables a better user experience for the user as before the ingredients and instructions were all on one page meaning the user had to scroll to see the information, by sectioning it out the user can choose what information they want to see without being overloaded with information, see figure eight.

1.5 Figure Eight







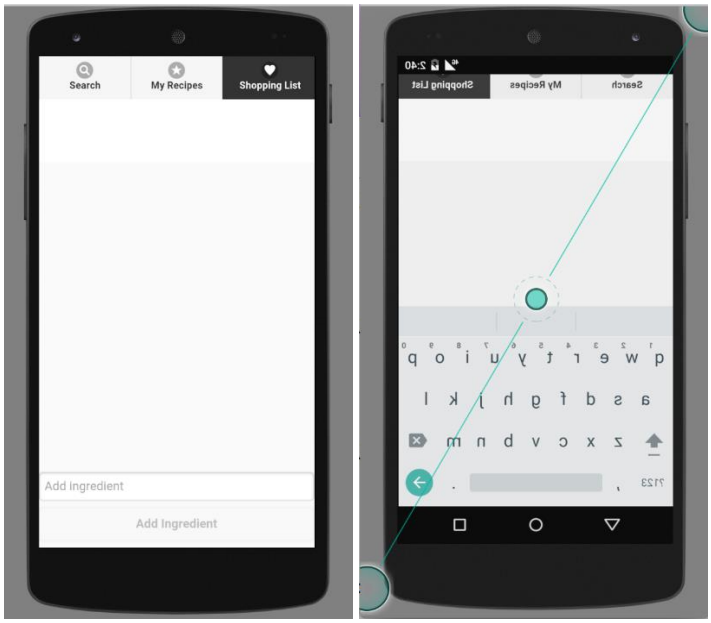
The feedback from observing the all users with the application was that they needed to be told when a recipe had been saved as they didn't know if this action was being implemented or not, see figure nine.

1.5 Figure Nine

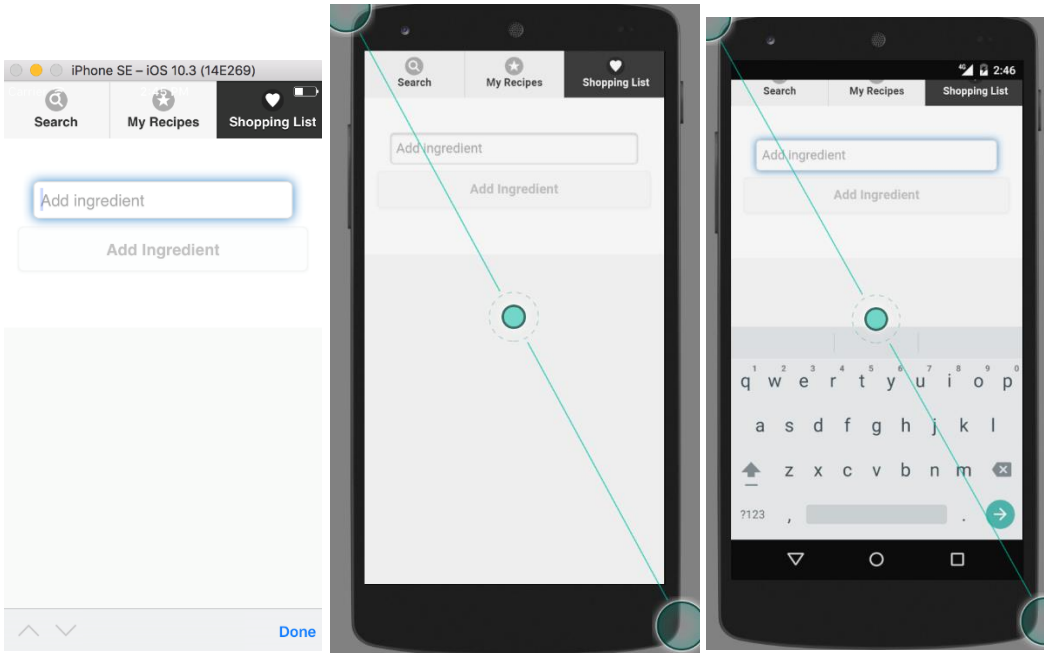
What the above shows is that when a user taps to save a recipe or to save their own recipe, a notification will appear telling the user that the action was completed successfully or not.

Another issue that was found from observing the users was that when a user when to add a product to the shopping list, the button would disappear behind the keyboard, making it inaccessible and giving a bad user experience, see figure ten. This is shown on an android virtual device, which shows that the keyboard covers the button. This was improved upon in the next iteration, see figure eleven, this shows that the button was moved to underneath the input instead of fixed at the bottom of the page so that when the keyboard was in use, the user could still access the button to add the product to the list, this was tested an android virtual device and an apple virtual device.

1.5 Figure Ten

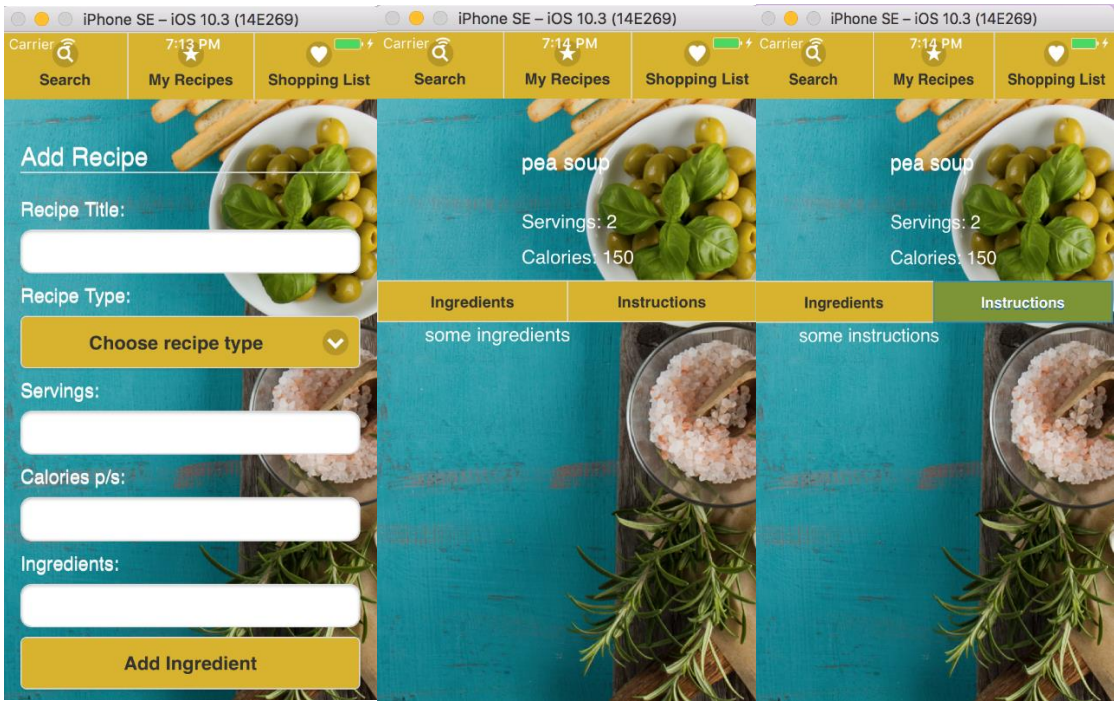


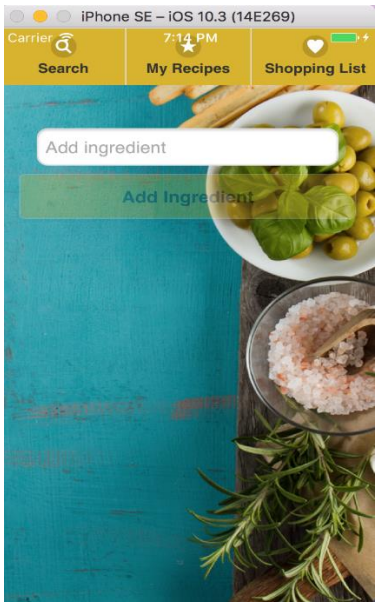
1.5 Figure Eleven



In the next meeting with the end-users, it was found that the round circular button to add the recipes didn't look appealing. Two users commented saying that the application did not look very aesthetically pleasing and if it came down to a good-looking application and this one, they would ultimately choose the good-looking application. To try and correct this images and better looking aesthetics were applied to the application, see figure below. Upon observing the users will the application it came to light that an option to apply no filters on the filters needed to be added as when a user accidentally clicked an option there was no way to remove it, this was the last feature to be implemented, see appendix two.

1.5 Figure Twelve


iPhone SE – iOS 10.3 (14E269)
Carrier
7:13 PM
Search
My Recipes
Shopping List
Add Recipe
Recipe Title:
Recipe Type:
Choose recipe type
Servings:
Calories p/s:
Ingredients:
Add Ingredient
iPhone SE – iOS 10.3 (14E269)
Carrier
7:14 PM
Search
My Recipes
Shopping List
pea soup
Servings: 2
Calories: 150
Ingredients
Instructions
some ingredients
iPhone SE – iOS 10.3 (14E269)
Carrier
7:14 PM
Search
My Recipes
Shopping List
pea soup
Servings: 2
Calories: 150
Ingredients
Instructions
some instructions


iPhone SE – iOS 10.3 (14E269)
Carrier
7:14 PM
Search
My Recipes
Shopping List
Add ingredient
Add Ingredient

1.6 Evaluation

Using users from the target group during the whole design and development process allowed the developer to improve the application iteratively and gained insight into what the users wanted from the application, a reflection of this will take place later in this work.

The developer conducted two evaluations on two applications where they, analysed the design, features and overall user experience of the application to help the developers enhance their application, see appendix three and four.

2 Cross-Platform Implementation

This chapter will walk through the implementation process of each feature and how these were tested.

2.1 Recipe Search

This feature was implemented by checking the connection by using the “cordova-plugin-network-information” which makes sure there is an internet connection before connecting to a free restful api called “edamam”.

The service allowed the developer to pull recipe information into the application in json format, this was then iterated through and shown in a list view to the user, see appendix five. This service permitted the user to add filters such as dietary requirements and how many calories in the recipe, however when implementing and testing the calories filter, the call was too narrow so no results were found, this led to the calorie filter being removed from the application, see appendix six which shows this feature on different devices, screen sizes and operating systems.

2.2 Recipe Information

Ingredients

After searching for a recipe, the user would then click on a recipe which would take them to another page where another call would be made to the service to retrieve specific recipe information in json format. The ingredients were then shown in a list view to the user, see appendix seven.

Instructions

When trying to implement the instructions for the recipe, an issue occurred; the instructions for the recipe could not be retrieved unless the service used was being paid for, in light of this, a button was added that would take the user to the webpage of the recipe to use the instructions. This was implemented with the “cordova-plugin-inappbrowser” plugin, see appendix eight.

Nutrients

The daily amount of nutrients within the recipe was shown here, originally this was just shown in text format, but it was pointed out that this did not look very appealing so visual bars were used to make the information more visually pleasing, see appendix nine. The progress bars were adapted from a third party that created different types of CSS3 progress bars (Coyier, 2011).

2.3 Shops near you

The closest shops near the user was implemented by using three different services, the user's latitude and longitude was retrieved from the device using a plugin called "cordova-plugin-geolocation", these co-ordinates were then used to make a call to a service that reverse geocoded the co-ordinates into a postcode. Once the postcode was retrieved, it was then used to make a call to a Tesco service that would send all the nearest Tesco shop locations which was then shown to the user with the shop names and how far away each shop was to the user's location, see appendix ten.

2.4 Add Recipe

This feature was implemented by using a form to gather all the recipe information and the use of a sqlite database to store all the data that could then be used later to retrieve and view specific recipe information. The reason for using a sqlite database instead of local storage was because the developer needed a more complex way to store the recipe information and the key-value used for local storage would not suffice. The developer used the "cordova-sqlite-evcore-extbuild-free" plugin to create and setup the database when the application opened, see appendix eleven for the table designs. The plugin was used to generate queries to store and retrieve the recipe information that the user would like to view, see appendix twelve.

2.5 Shopping List

The developer adapted code from JqueryScript which shows a bootstrap list using local storage (2017), an ingredient is typed into the input box which is then added to a list, this list is then stored in local storage. When the user comes off this page and returns, the ingredients are retrieved from local storage and shown in a list view with a cross button that allows the user to delete items, see appendix thirteen.

3 Reflection

Using users throughout the design and development process proved to be an insightful experience and was shown to be a success; this allowed the developer to design and develop to the exact requirements for the end user instead of designing and developing features the developer "thought" the end user's might like. They also demonstrated to be a good tool when it came to designing for a good user experience, as without this, the developer would never have known about the issues identified which ultimately led to a better experience for the user.

There were many issues encountered through the development of the prototype such as when implementing the instructions feature for the recipe, it was found that this could not be implemented unless the service being used was paid for, instead of showing the instructions within the application itself, a button was design which took the user to the web-page where the instructions could be found. For the future, the application could offer a paid version which would pay for the service, allowing the user to view the instructions within the application.

Another issue that was found in the early stages of development was when trying to deploy to a device; the application would install successfully but when the developer tried to open the application it would crash. Manual debugging of the plugins was needed to find out which plugin was crashing the application, this was done by adding a plugin one at a time and testing the application. Further inspection found that the splash screen plugin was causing the plugin to crash on the device which was a Samsung S6 Edge. However, this plugin would work on other android devices. It remains unclear why the plugin crashes on the device used, it could be the specific android version or the browser version but there was not enough time to investigate this further.

Another issue found was that when implementing the SQLite database and testing with an android emulator, this plugin would not work with versions Marshmallow or Lollipop on Nexus devices 5 and 6. It did work on a physical device but this was not helpful for debugging purposes when creating the tables as the database could not be accessed without the device being rooted. To work around this issue, this part of the process was debugged with an IOS Simulator, which allowed the developer to view the database and its tables. However, when using the simulator to test the application it was found that dynamically added click events on span elements did not work, this was fixed by adding a dummy on click event with a void parameter so that IOS could recognise that the element is clickable.

The developer found that developing a cross platform that would create the same user experience across multiple devices and operating systems was a difficult process and that due to the many android version and devices, it was hard to maintain the same user experience across these. Yalantis (2017) states that end users get familiar with one platform and each platform has its own way of delivering a great user experience which can be hard to achieve through cross-platform development. This means that what an IOS user and Android user expect in respect to a good user experience may be different and using cross platform to accommodate both these may be difficult to implement. With that being said, the developer enjoyed learning Phonegap and

creating a cross platform application. The developer found it exasperating to have to keep uploading to Phonegap to test on a device, this was because the upload could take anywhere from one minute to fifteen minutes to upload. It would have also been nice to further test the application on a windows operating system too.

When an application is needed for multiple platforms and for little costs then cross platform development could be a good choice due to only have one code base, whereas done natively this would have to be developed for one platform and then adapted and re-coded for another platform. However, if an application needed full hardware access and high performance then a native application may be a better option. “native apps provide the richest user experience. Source code is efficient, with fast performance, consistent look and feel and full access to the underlying platform hardware and data” (Xanthopoulos and Xinogalos, 2013, 213).

Overall the developer is happy with the results but if there had been more time and resources they would research more into Android and Apple conventions to further customise the user experience for both these platforms. Furthermore, the developer would add a few more features such as editing of the custom recipes, once the user has viewed the nearest shops; they could then click on one which would then bring up directions to that shop and a rating system of the recipes would also be included.

4 References

Coyier, C. (2011) CSS3 Progress Bars. CSS-TRICKS. Available from https://css-tricks.com/css3-progress-bars/ [accessed 29/4/17].

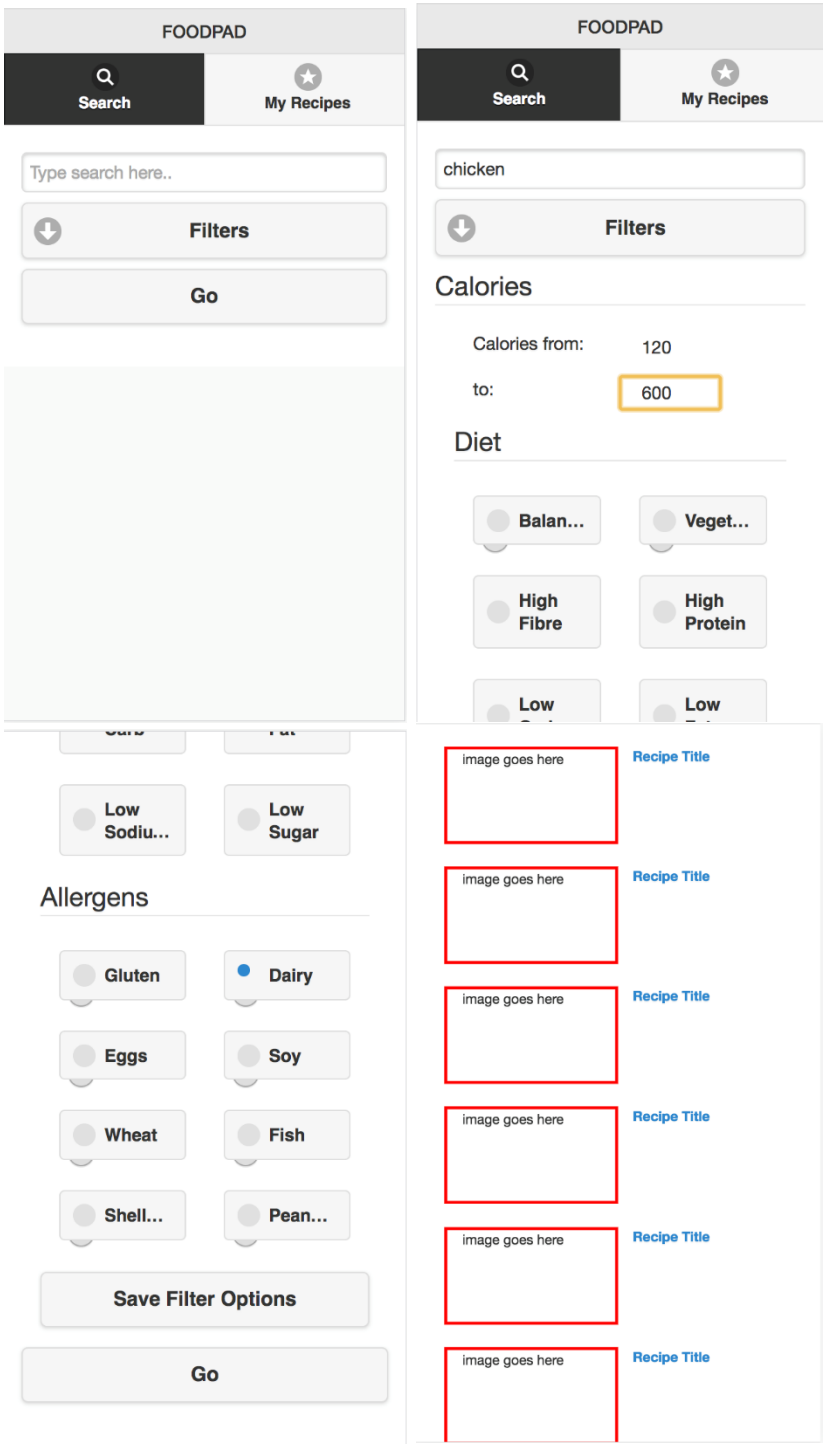
JqueryScript (2017) Bootstrap To Do List.

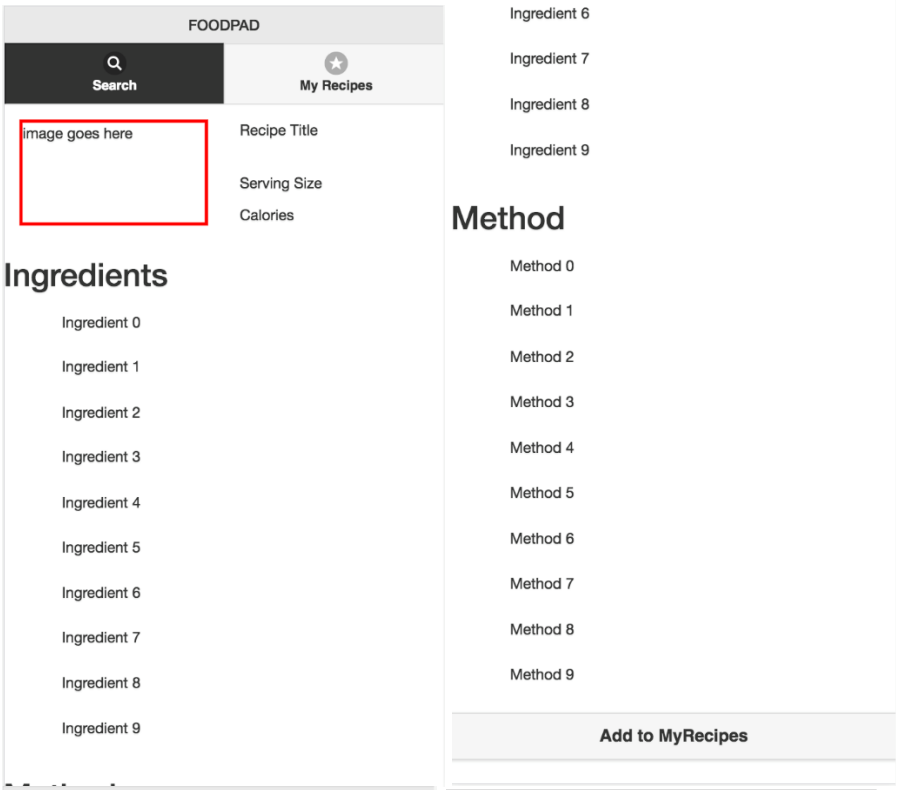
Xanthopoulos, S. and Xinogalos, S. (2013) A comparative analysis of cross-platform development approaches for mobile applications. In: Proceedings of the 6th Balkan Conference in Informatics: ACM, 213-220.

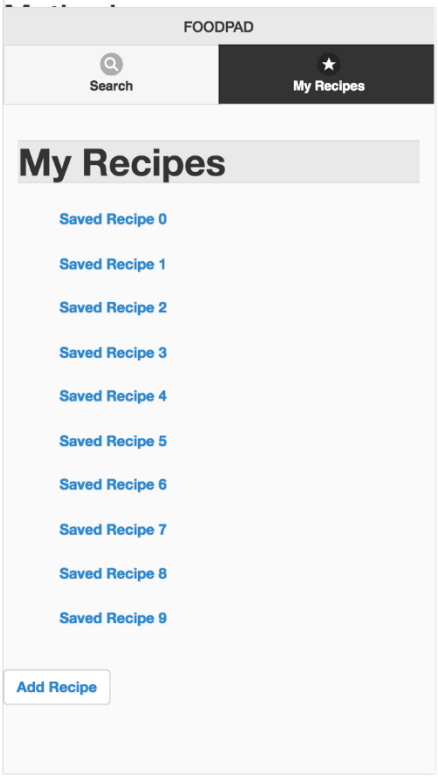
Yalantis. (2017) Native vs Cross Platform. Yalantis. Available from https://yalantis.com/blog/native-vs-cross-platform-app-development-shouldnt-work-cross-platform/ [accessed 29/4/17].

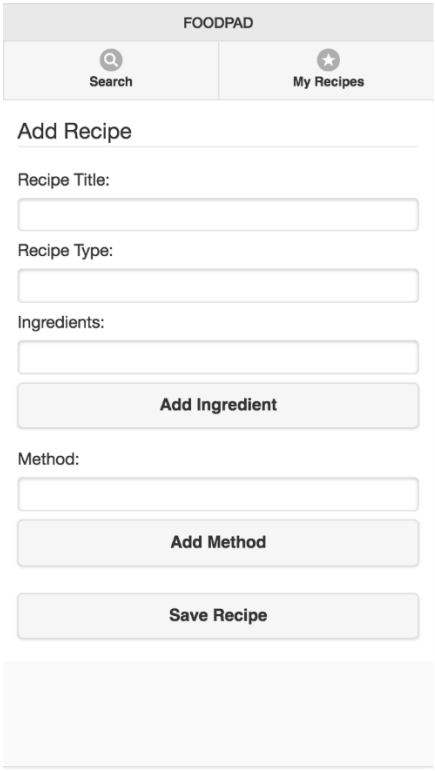
5 Appendices

Appendix 1

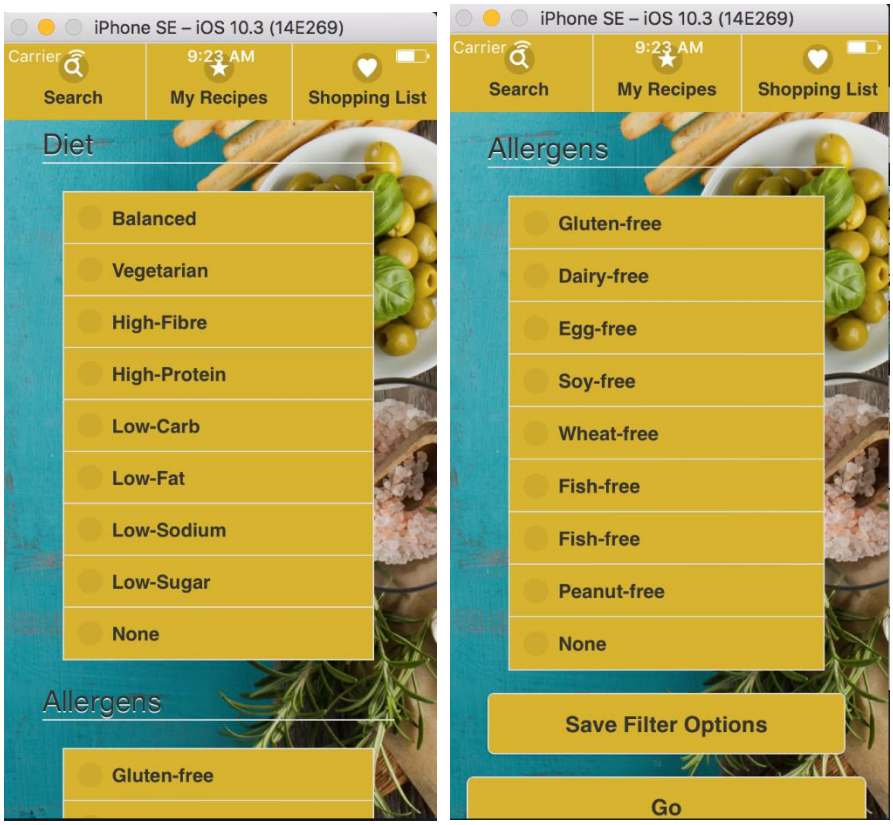



FOODPAD
Search
My Recipes
image goes here
Recipe Title
Serving Size
Calories
Ingredients
Ingredient 0
Ingredient 1
Ingredient 2
Ingredient 3
Ingredient 4
Ingredient 5
Ingredient 6
Ingredient 7
Ingredient 8
Ingredient 9
Ingredient 6
Ingredient 7
Ingredient 8
Ingredient 9
Method
Method 0
Method 1
Method 2
Method 3
Method 4
Method 5
Method 6
Method 7
Method 8
Method 9
Add to MyRecipes


FOODPAD
Search
My Recipes
My Recipes
Saved Recipe 0
Saved Recipe 1
Saved Recipe 2
Saved Recipe 3
Saved Recipe 4
Saved Recipe 5
Saved Recipe 6
Saved Recipe 7
Saved Recipe 8
Saved Recipe 9
Add Recipe


FOODPAD
Search
My Recipes
Add Recipe
Recipe Title:
Recipe Type:
Ingredients:
Add Ingredient
Method:
Add Method
Save Recipe

Appendix 2


iPhone SE – iOS 10.3 (14E269)
Carrier
9:23 AM
Search
My Recipes
Shopping List
Diet
Balanced
Vegetarian
High-Fibre
High-Protein
Low-Carb
Low-Fat
Low-Sodium
Low-Sugar
None
Allergens
Gluten-free
iPhone SE – iOS 10.3 (14E269)
Carrier
9:23 AM
Search
My Recipes
Shopping List
Allergens
Gluten-free
Dairy-free
Egg-free
Soy-free
Wheat-free
Fish-free
Fish-free
Peanut-free
None
Save Filter Options
Go

Appendix 3

Student being peer reviewed: Luke

App name and brief description: Lincoln Cathedral

Target group and requirements: I think the application as it is now would suit the older generation as it is not technically challenging and easy to use.

Functionality

Easy identifiable navigation with good use of icons, each button within the navigation works as expected, Home button takes you to the main page, Scan button to open the scanner, Map button that takes you to a view of a map of inside the cathedral, and the help button takes you to some helpful information.

Has a call button on the help page that opens up the dial pad on the mobile device.

Has a notification feature that notifies the user of a new service.

Able to see information upon scanning cathedral products.

User Experience and Usability

The application is easy to use, shows an error message when a non-cathedral product is scanned.

you have clear focused content within the application and your navigation is easy and simple to use which follows the best practices for mobile design guidelines.

Summary of strengths and weaknesses

Good use of colour scheme and stays the same throughout the application.

The logo and the notifications are a nice added touch to the overall application.

The background can be a bit busy, this is the first thing I look at when I open the application, also if there are several services on the home page the background can't really be seen.

When I click back from the scanner an error message is shown which is excellent but is quite generic as if I scan a non-cathedral product the same error shows, see screenshot 1.

I like the map but it's not very easily viewable, could make this more interactive, show where you are on the map or at least zoomable, see screenshot 2.

Actionable recommendations

The error message could be turned into a pop up error message so the user knows what's happened and make the error messages more defined to what the user did wrong or what has gone wrong instead of just one error message for all.

The application crashes if I try to load without my internet turned on, maybe show a error message asking the user to turn their internet on or have a refresh button?

On the homepage, there is a lot of information to process, a user may get overwhelmed by all this information so you could enhance the experience by making these sections only viewable if the user clicks that section, would also switch the notices and todays services around so that's the first thing the user reads and because the todays services will always be changing from day to day I would think this section is more important.

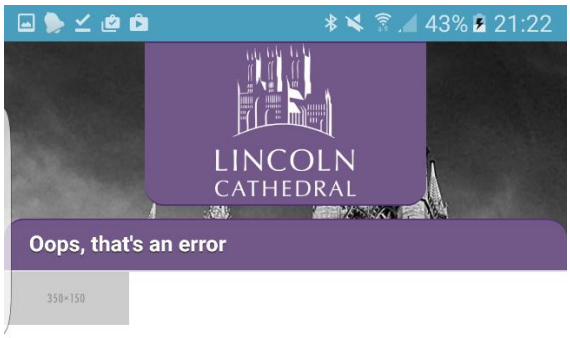
The call button on the help page could be a bit more prominent as it blends in with the section above. Also, what happens if someone using a tablet with no call feature clicks the call button?

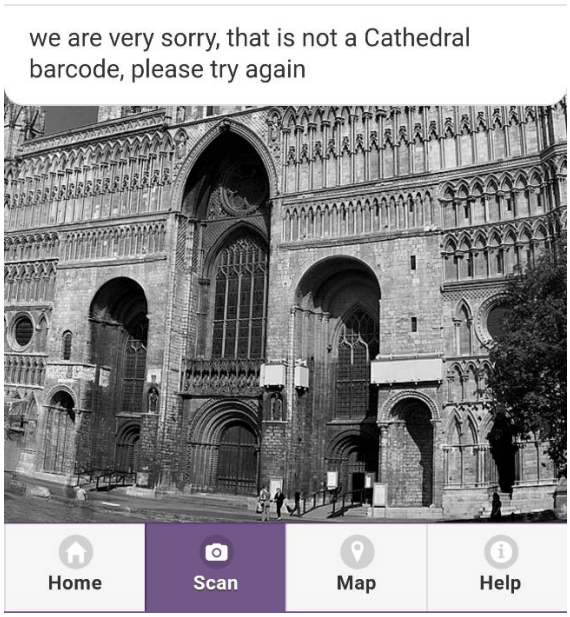
On closure and reopen of the application the service notification appears again? From a usability perspective, this would get annoying

When there's a lot of information on the page, I must scroll to use the navigation bar, could fix this to the screen so the navigation bar is always on show, see screenshot 6.

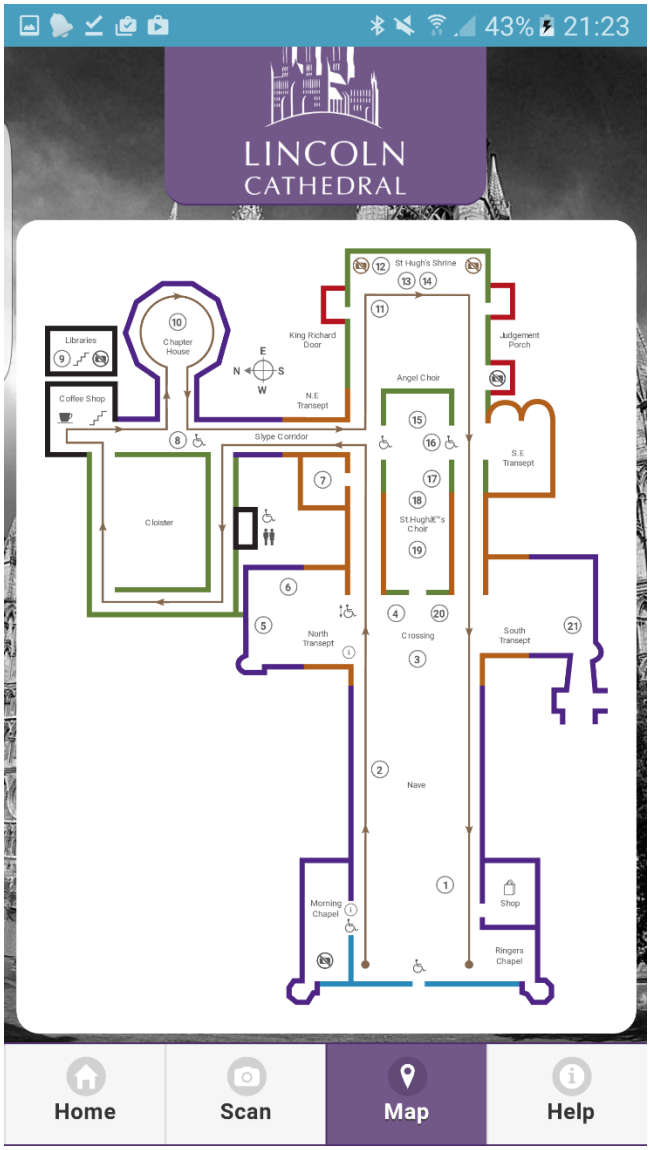
Screenshots

Screenshot 1

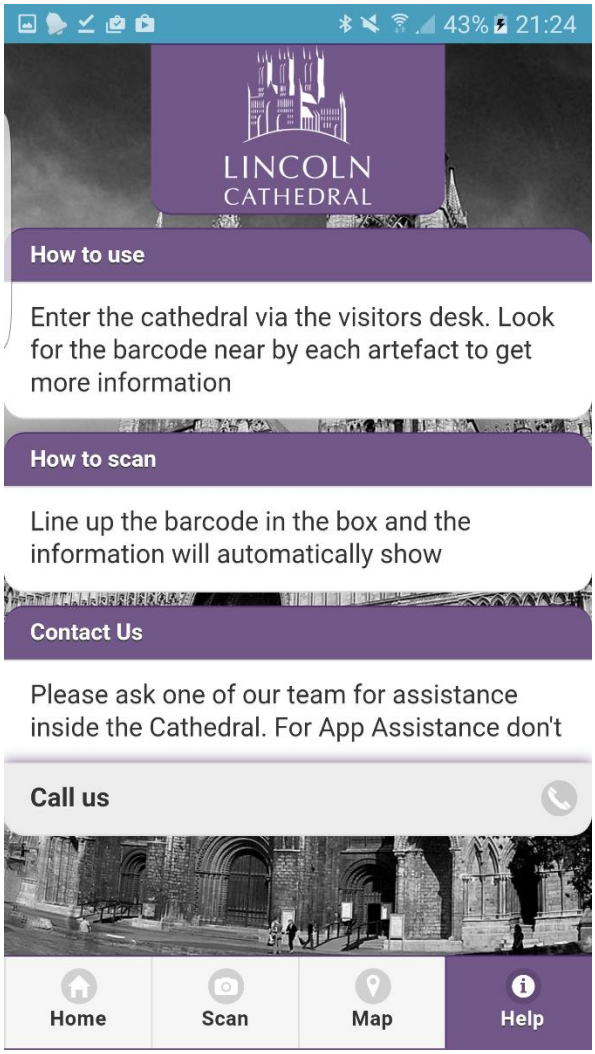




Screenshot 2

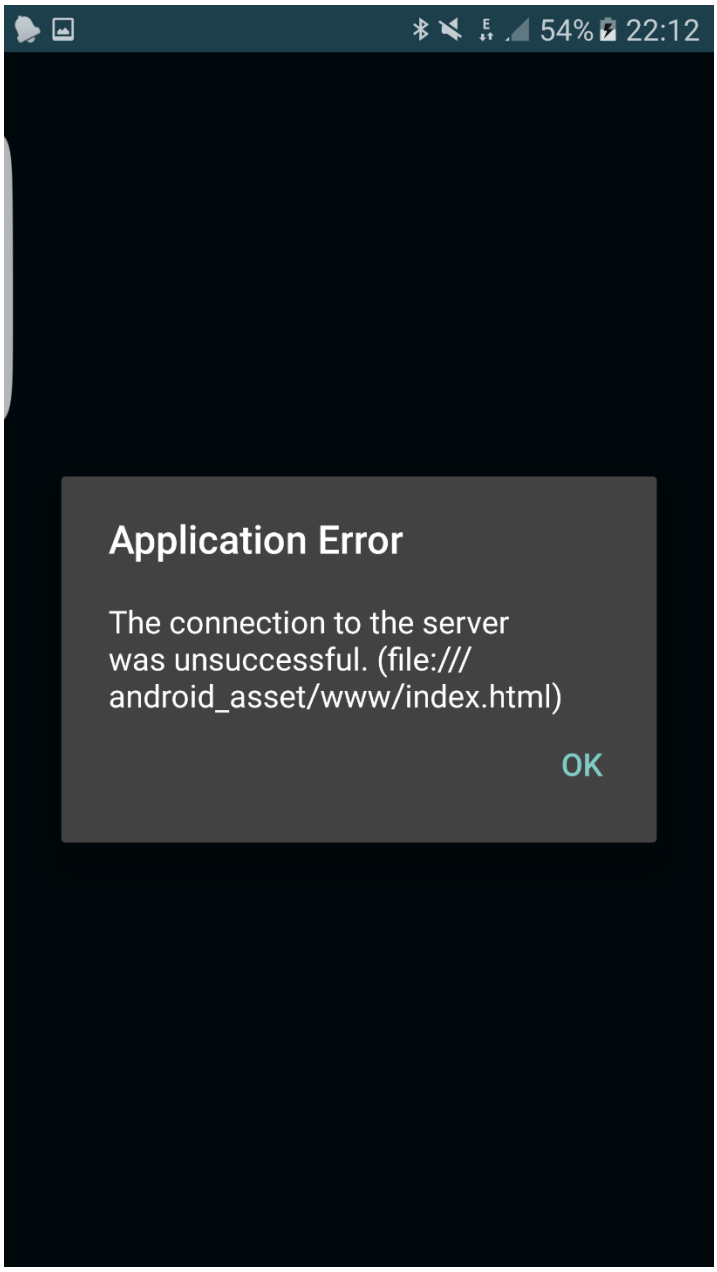


Screenshot 3

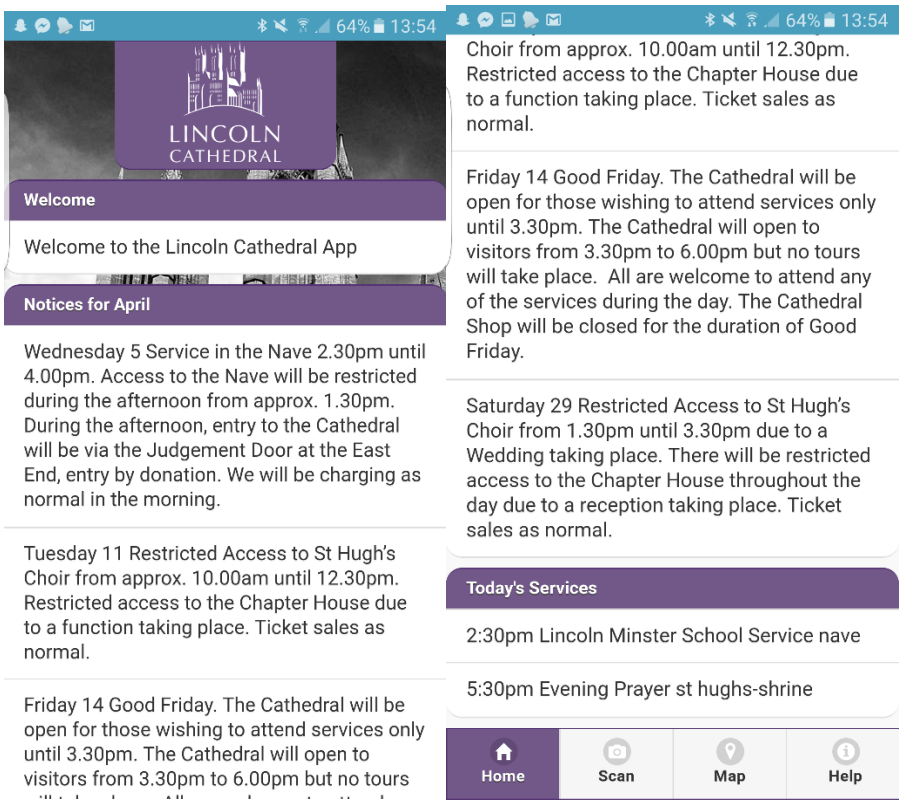


Screenshot 4

Screenshot 5



Screenshot 6



Appendix Four

Student being peer reviewed: David

App name and brief description: To Do List: an application to show singular tasks for the current day and later on in the day.

Target group and requirements

Target group of this application is aimed all students who need to organize all the tasks they need to do daily.

Functionality

Can add task name, date task needs to be completed by and can check off when done, can also add a task to be completed later, there is an edit task call to action where you can add a time to be completed, there is also a new task call to action but doesn't do anything yet.

User Experience and Usability

- The buttons and input boxes blend into the background of the application, so can be difficult to see which widgets are buttons.
- No placeholder for the inputs so unsure what to put in, there is a task name but nowhere to input the content of the task.
- Some buttons don't work yet or have no relevance to your application.

Summary of strengths and weaknesses

the layout could be better such as adding a different colour for the buttons so the user can clearly see what they are doing.

Later tasks and current tasks are on the same page, may need to separate these as can be confusing to the user.

Icons used are a little misleading, home button for the task list, star icon for the new task.

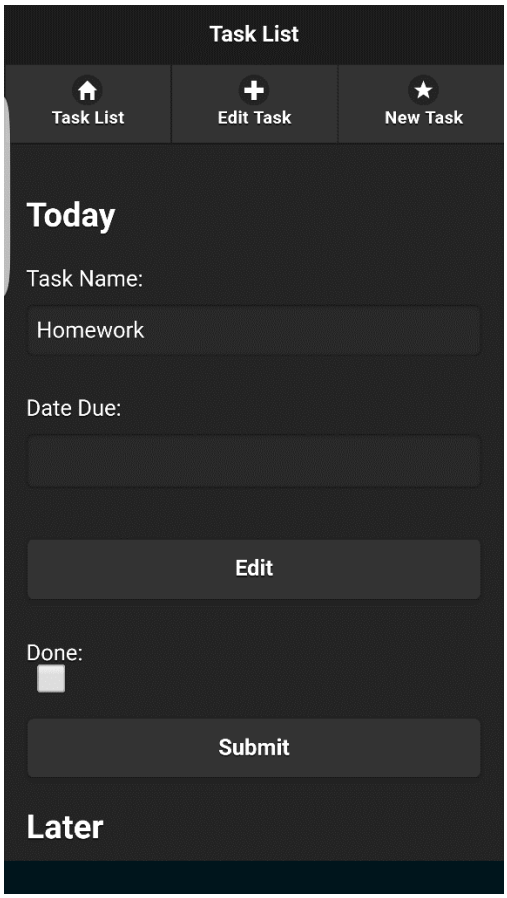
Actionable recommendations

- Change the button colours so they stand out and highlight which page the user is on, see screenshot 1
- Add a task notes section, because right now you can only add a title, see screenshot 1.
- Be able to save the tasks to a database or local storage

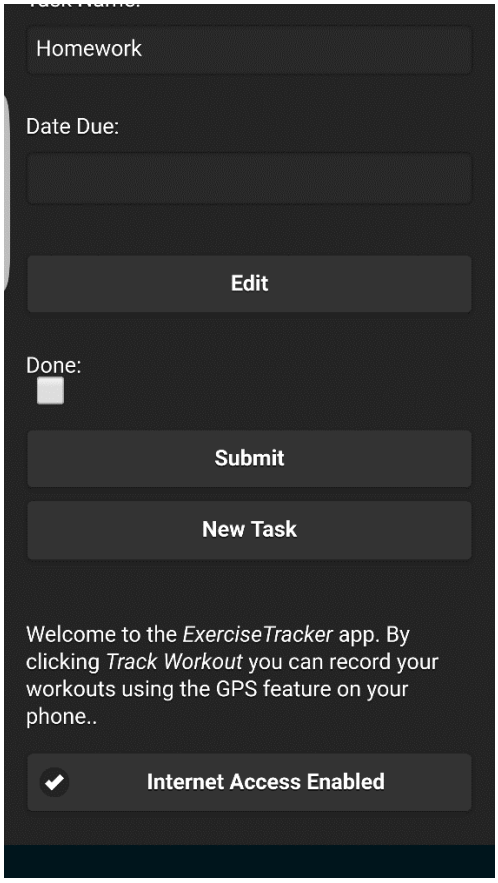
- Need to remove the geolocation logo and buttons that have no relevance to your application, see screenshot 2, 3 and 5.
- Could use a star rating system for the priorities so the user doesn't have to input anything, see screenshot 4

Screenshots

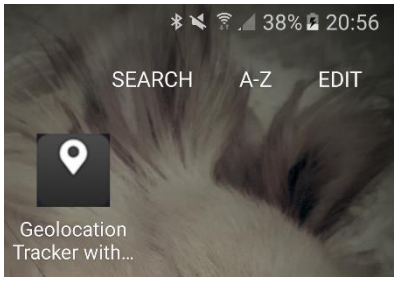
Screenshot 1



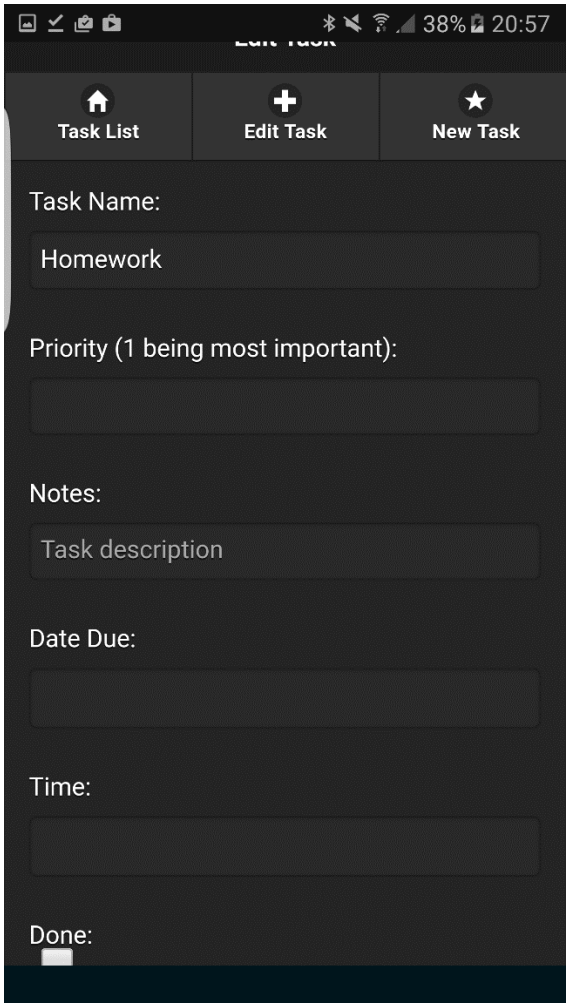
Screenshot 2



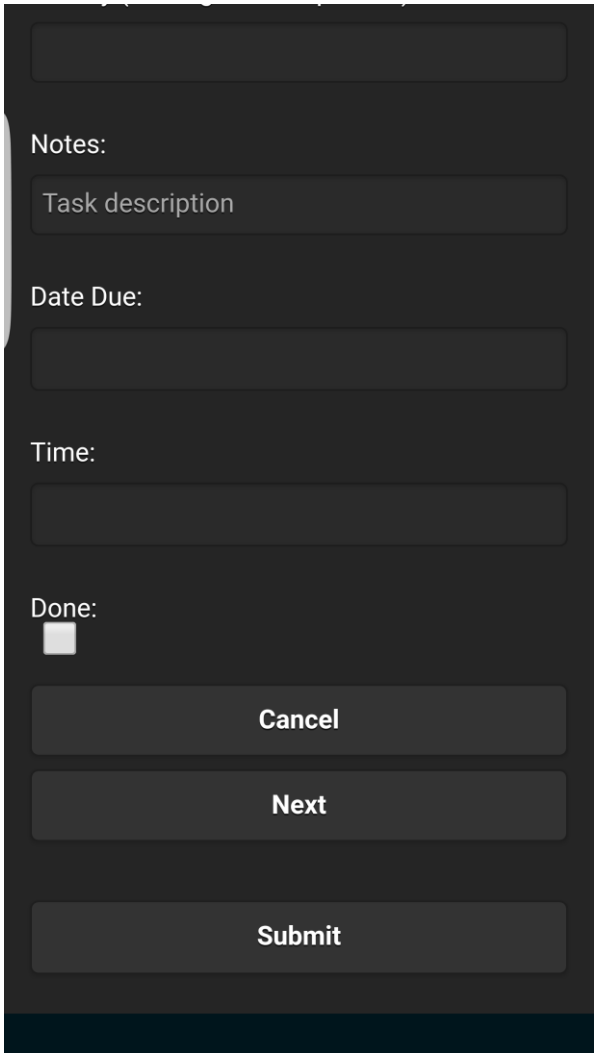
Screenshot 3



Screenshot 4

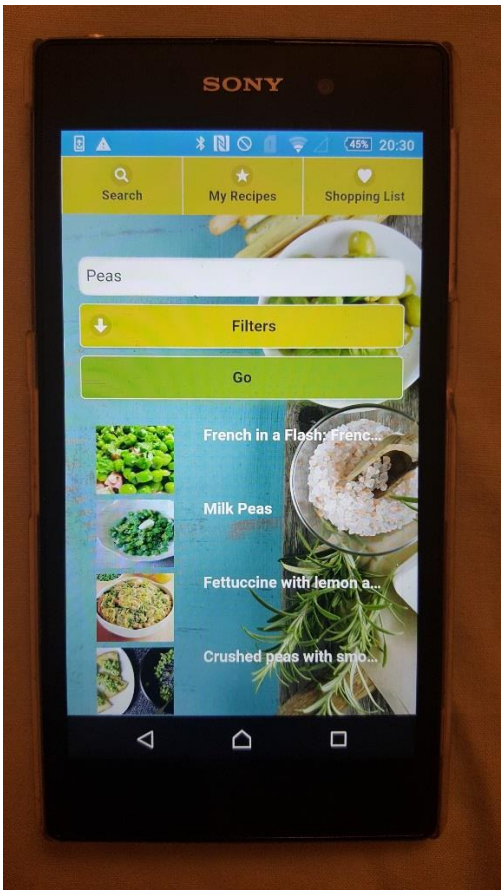


Screenshot 5

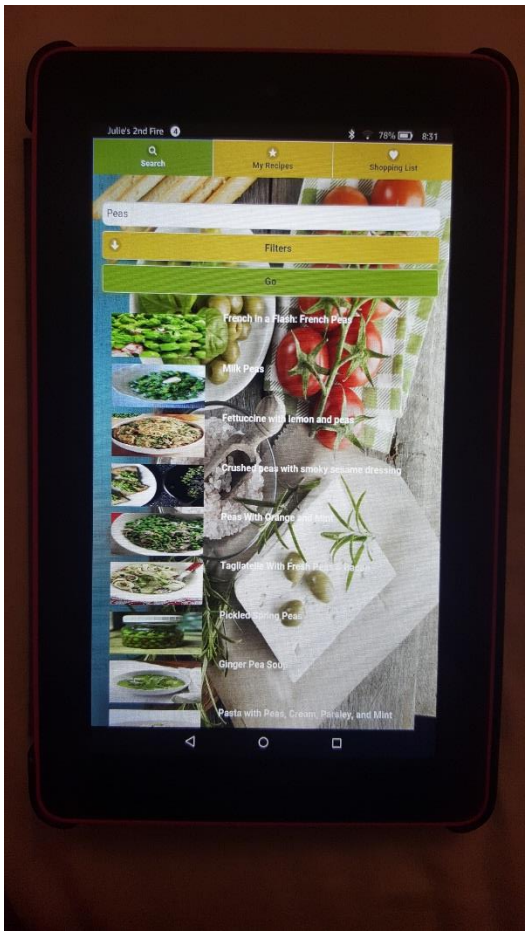


Appendix 5

Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels.

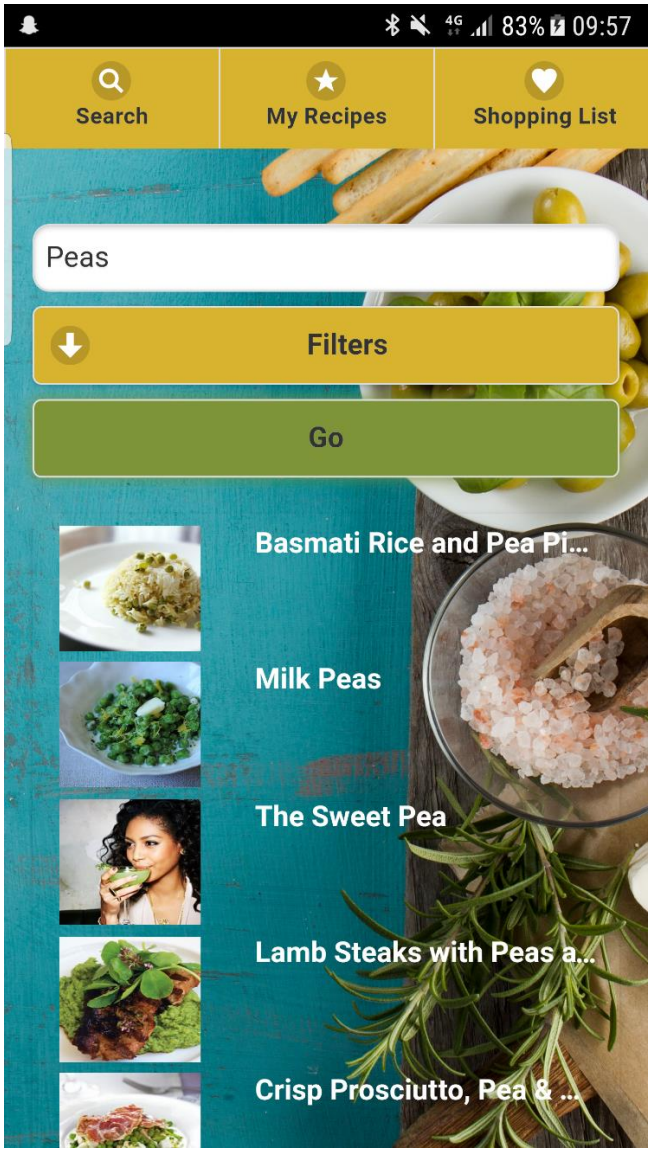


Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.

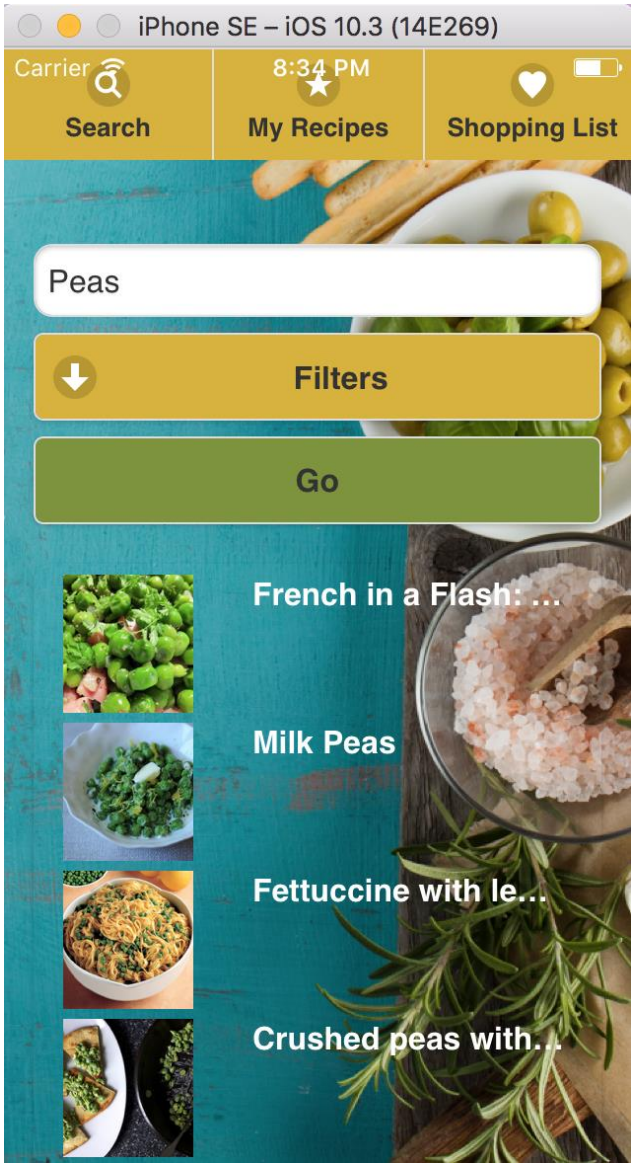


Samsung Tablet, Android 4.2.2, 7-inch display – 600x1024 pixels.

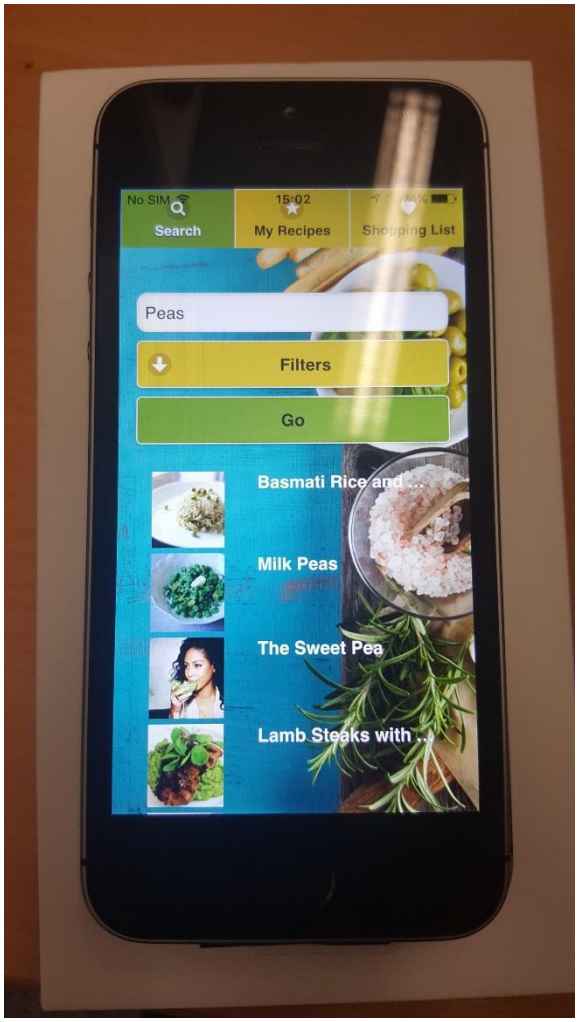
Samsung S6 Edge, Android version 7.0, 5.1-inch display – 1440x2560 pixels.



IOS SE, 10.3 (Simulator), 4-inch retina display – 1136x640 pixels.

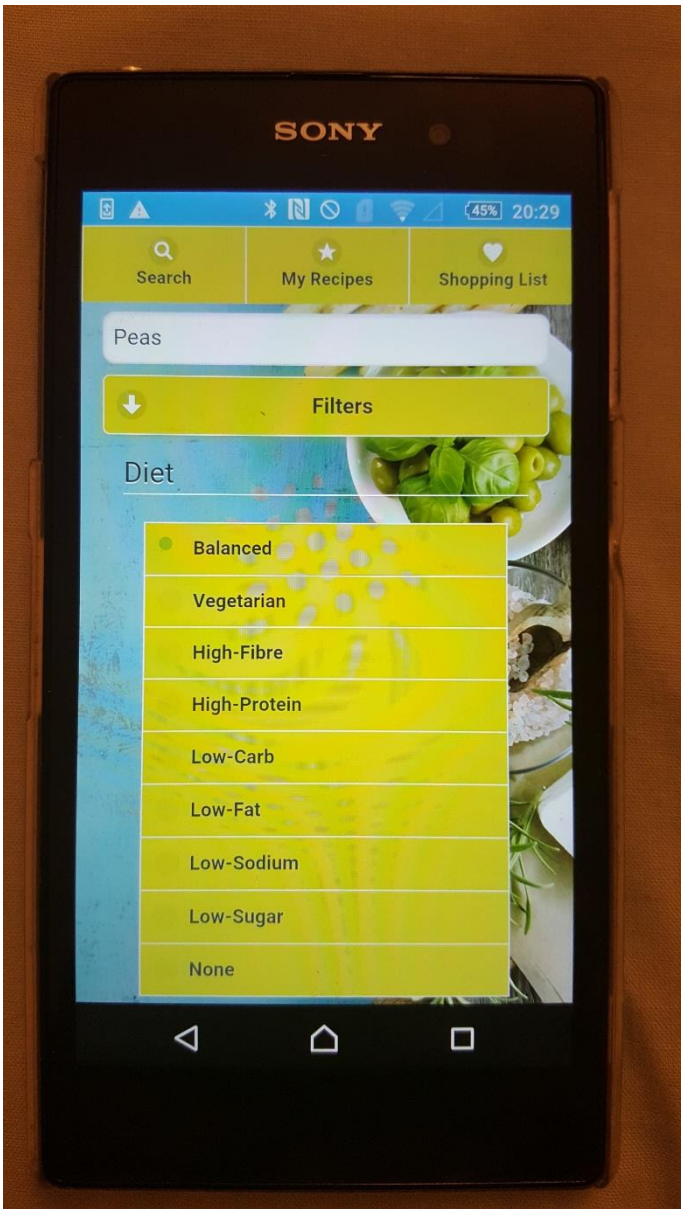


IOS IPhone SE version 9, 4-inch retina display – 1136x640 pixels.

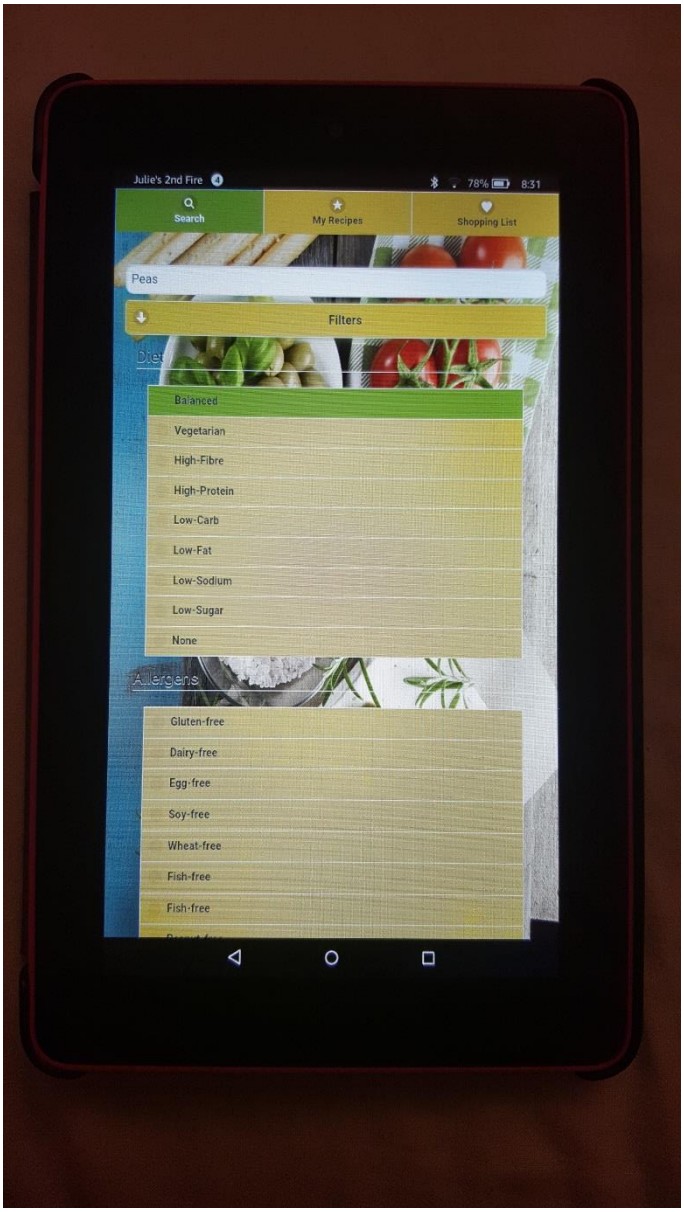


Appendix Six

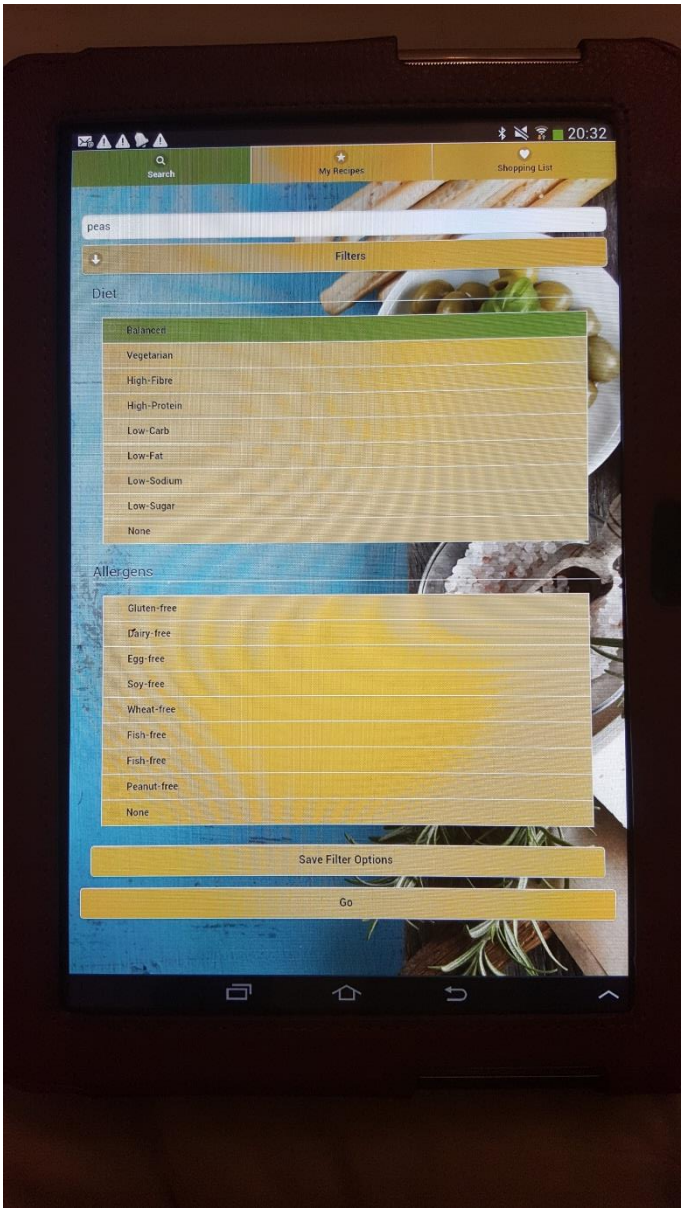
Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels



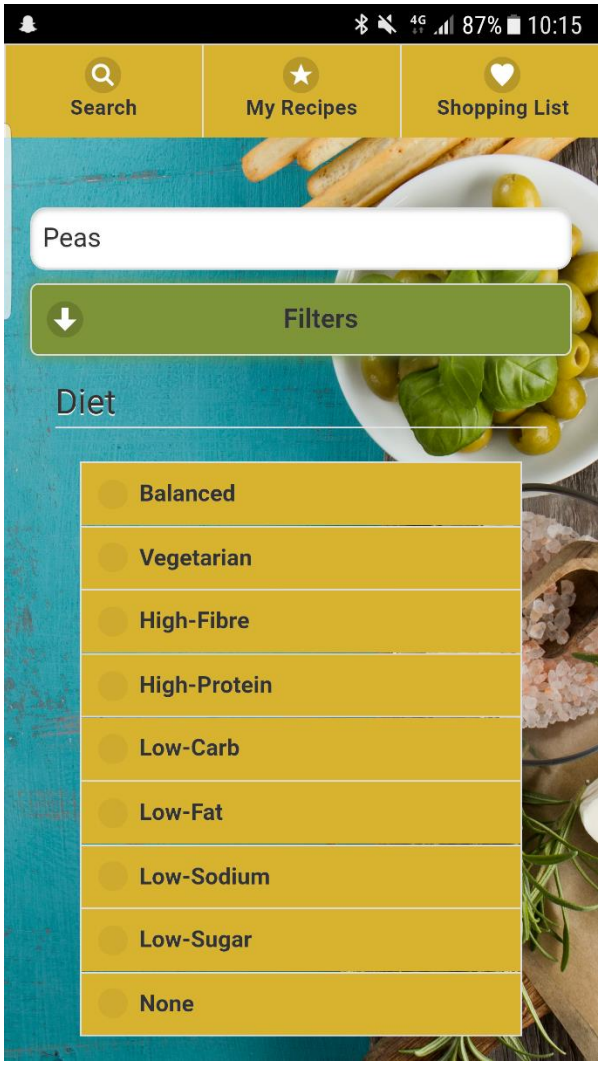
Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.



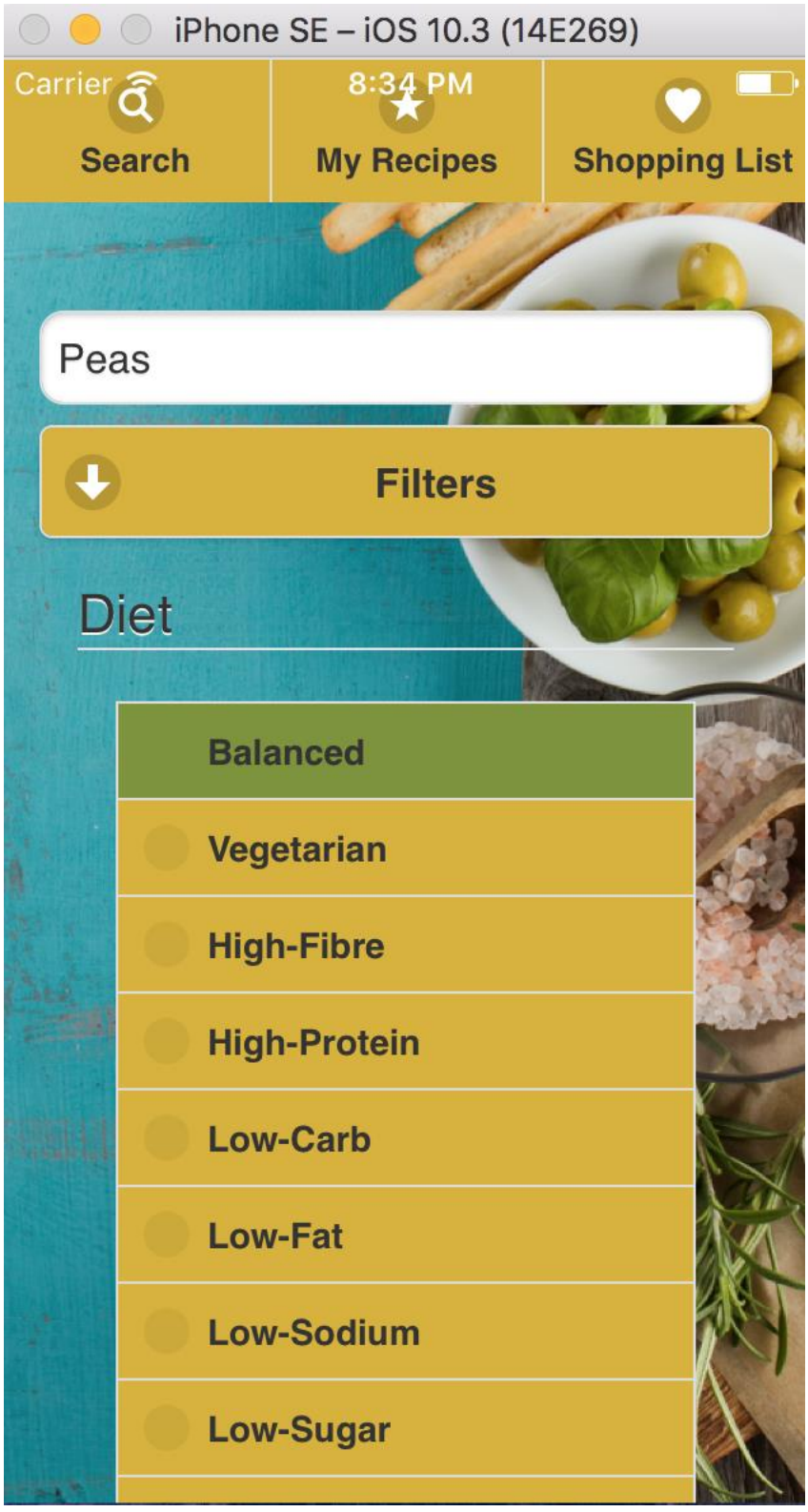
Samsung Tablet, Android 4.2.2, 7-inch display – 600x1024 pixels.



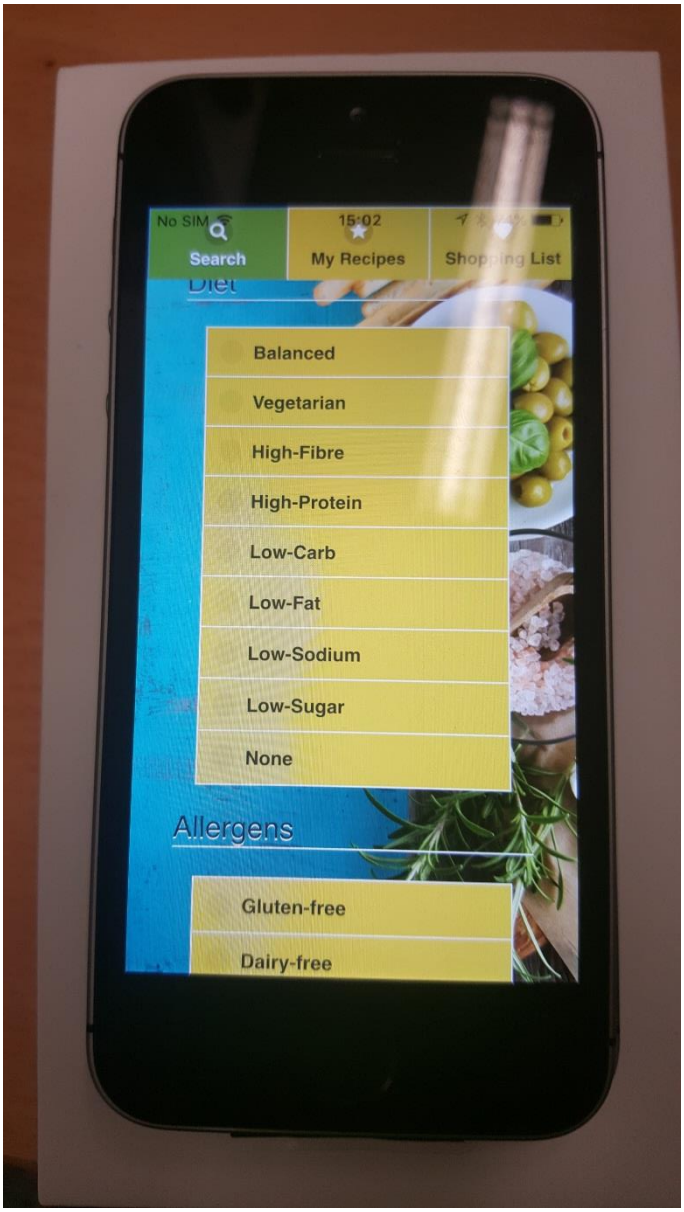
Samsung S6 Edge, Android version 7.0, 5.1-inch display – 1440x2560 pixels.



IOS SE, 10.3 (Simulator) , 4-inch retina display – 1136x640 pixels.

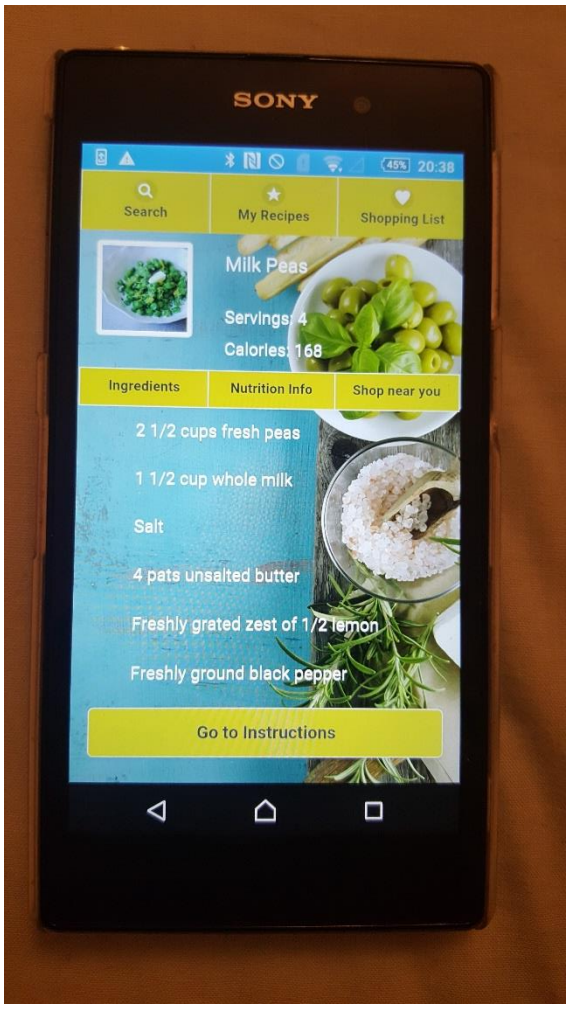


IOS IPhone SE, version 9, 4-inch retina display – 1136x640 pixels.

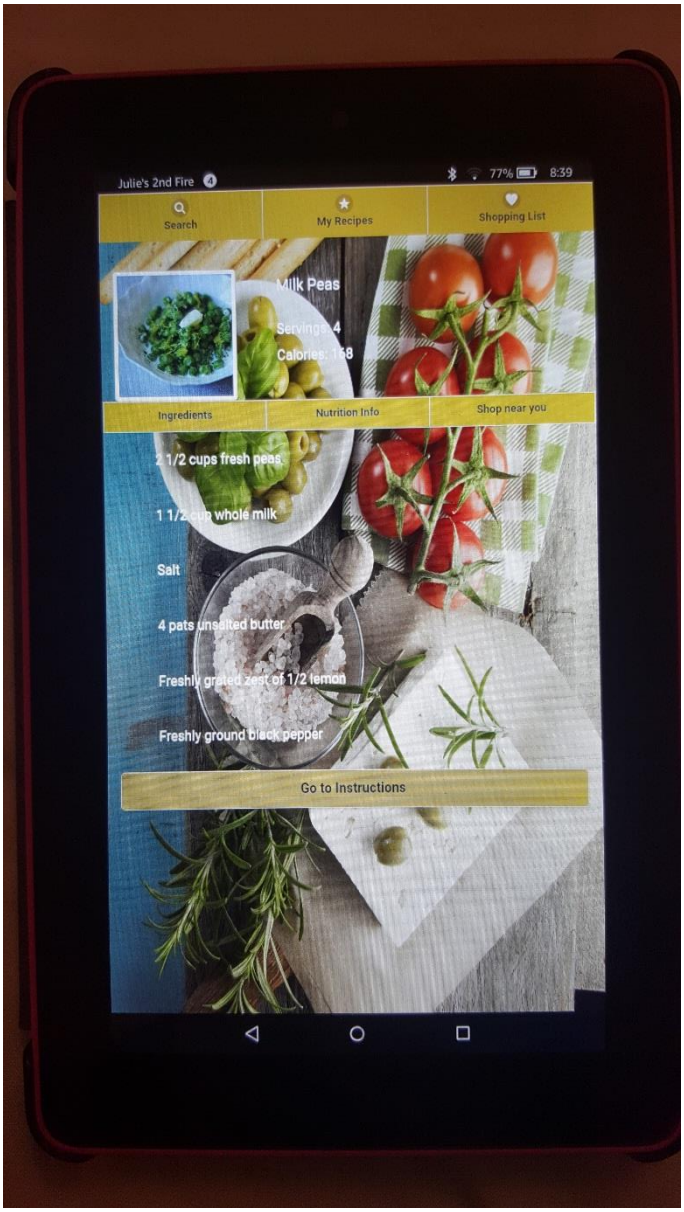


Appendix Seven

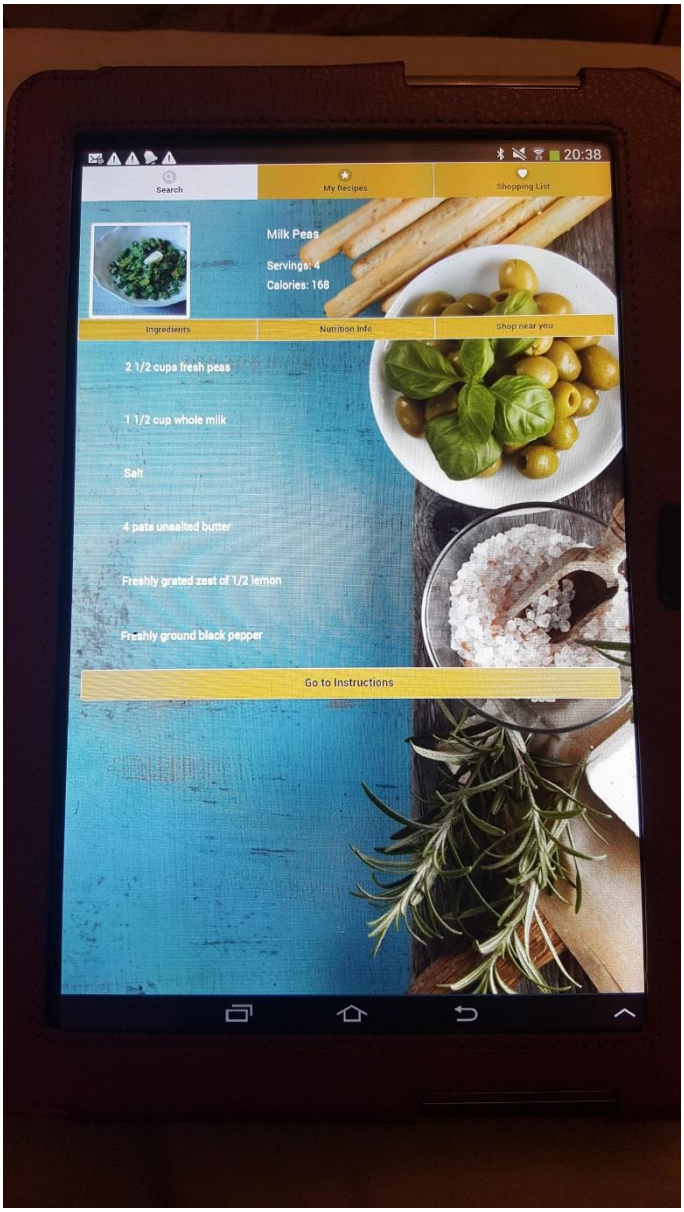
Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels



Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.



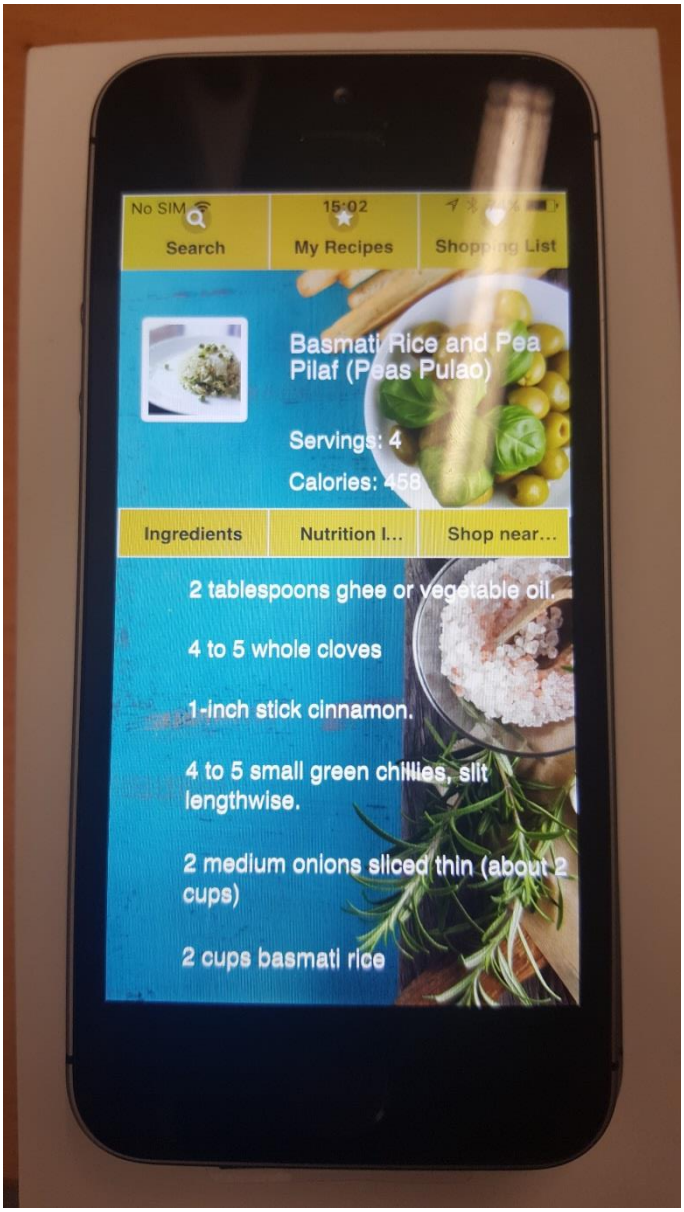
Samsung Tablet, Android 4.2.2, 7-inch display – 600x1024 pixels.



Samsung S6 Edge, Android version 7.0, 5.1-inch display – 1440x2560 pixels.

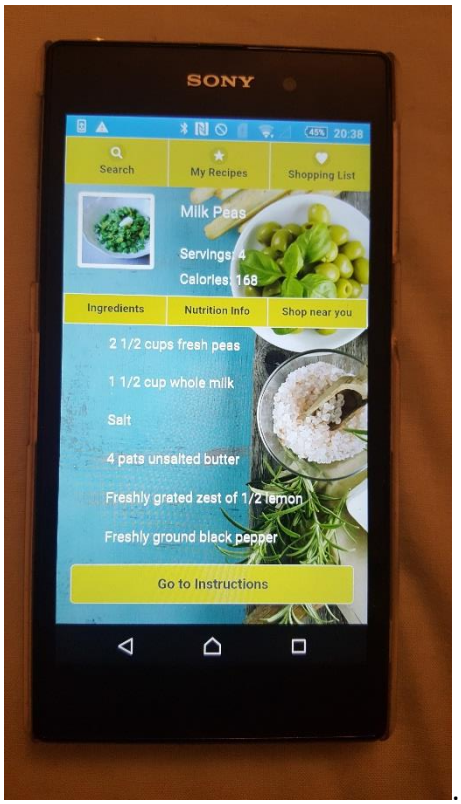
IOS SE, 10.3 (Simulator), 4-inch retina display – 1136x640 pixels.

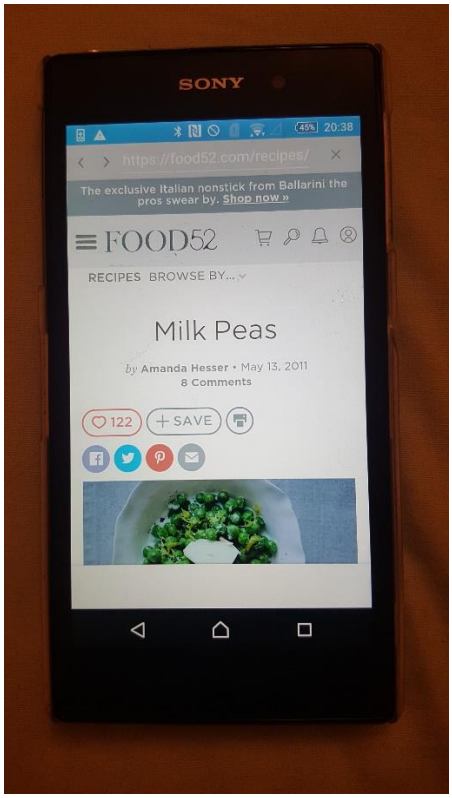
IOS IPhone SE, version 9, 4-inch retina display – 1136x640 pixels.



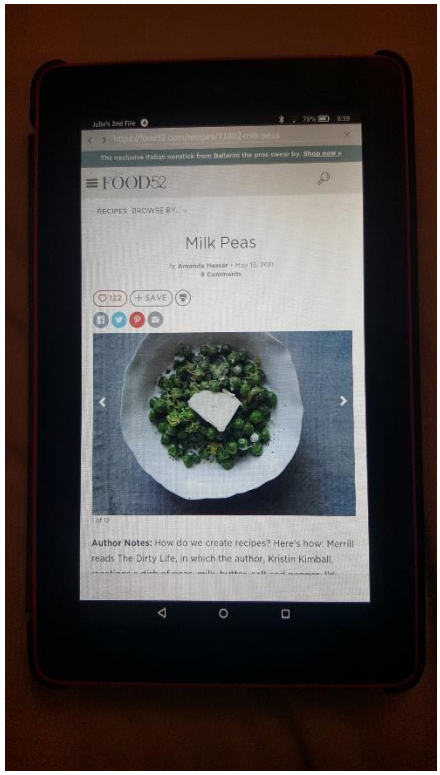
Appendix Eight

Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels

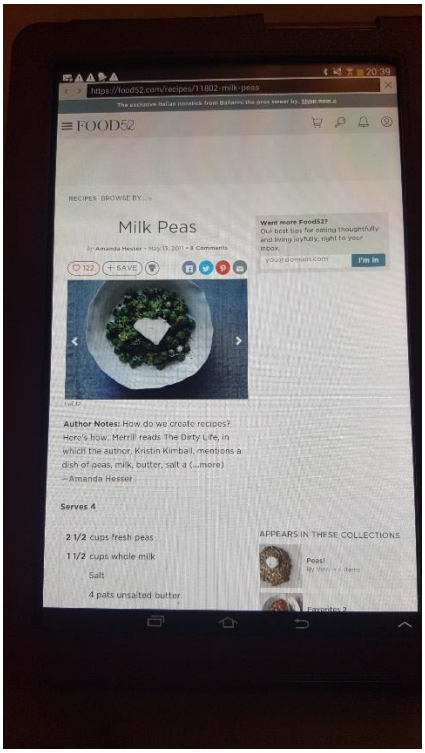




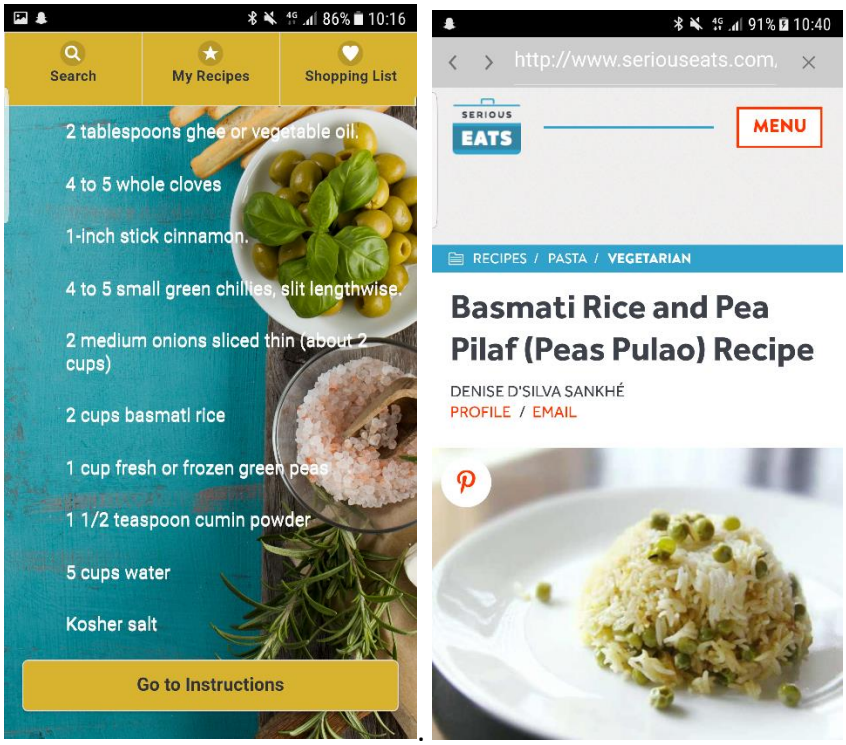
Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.



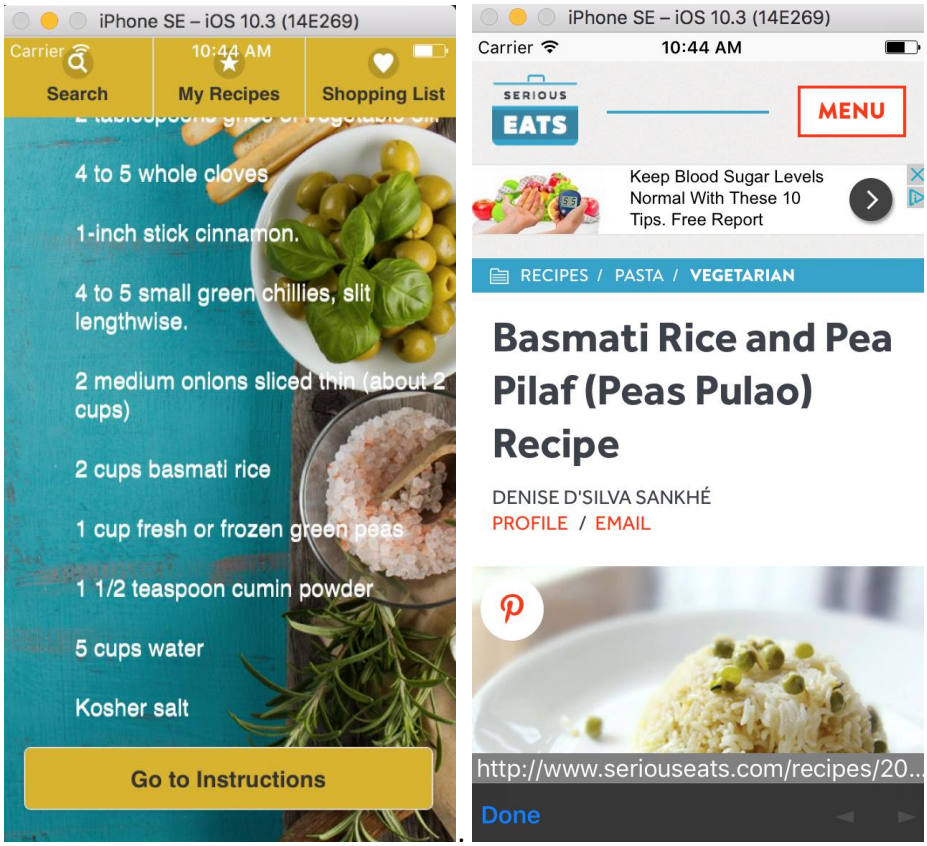
Samsung Tablet, Android 4.2.2, 7-inch display – 600x1024 pixels.



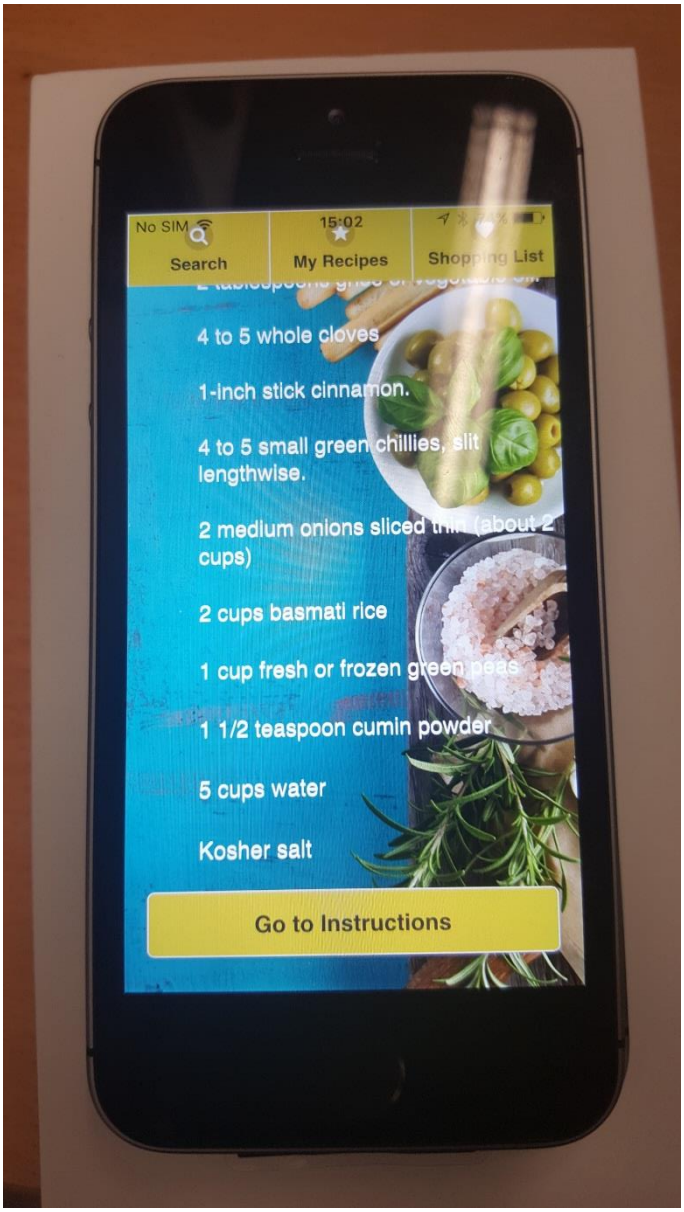
Samsung S6 Edge, Android version 7.0, 5.1-inch display – 1440x2560 pixels.



IOS SE, 10.3 (Simulator) , 4-inch retina display – 1136x640 pixels.

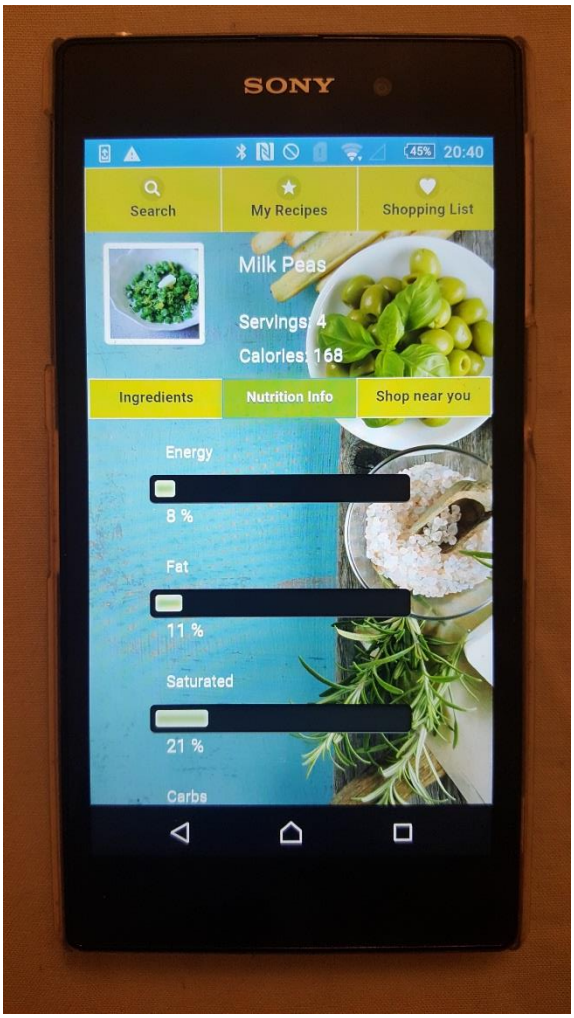


IOS IPhone SE, version 9, 4-inch retina display – 1136x640 pixels.

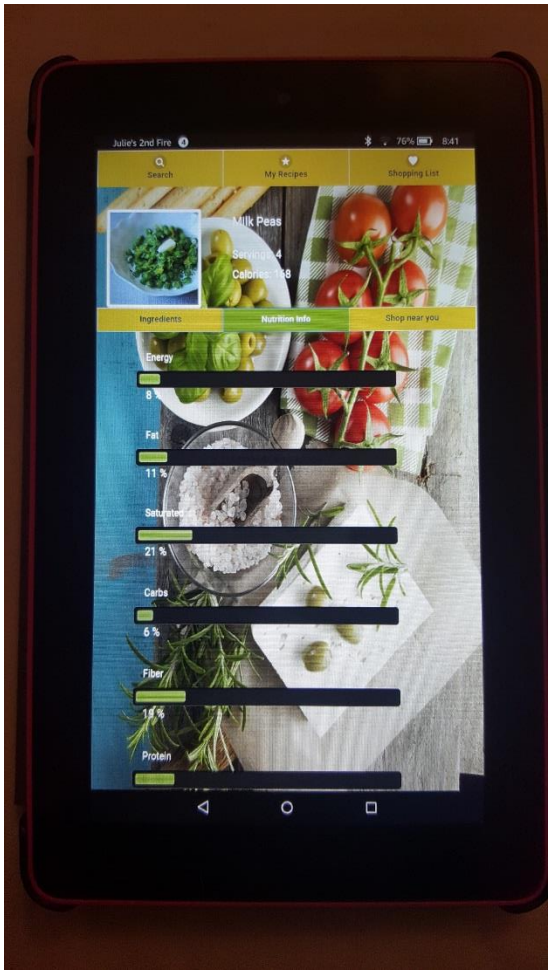


Appendix Nine

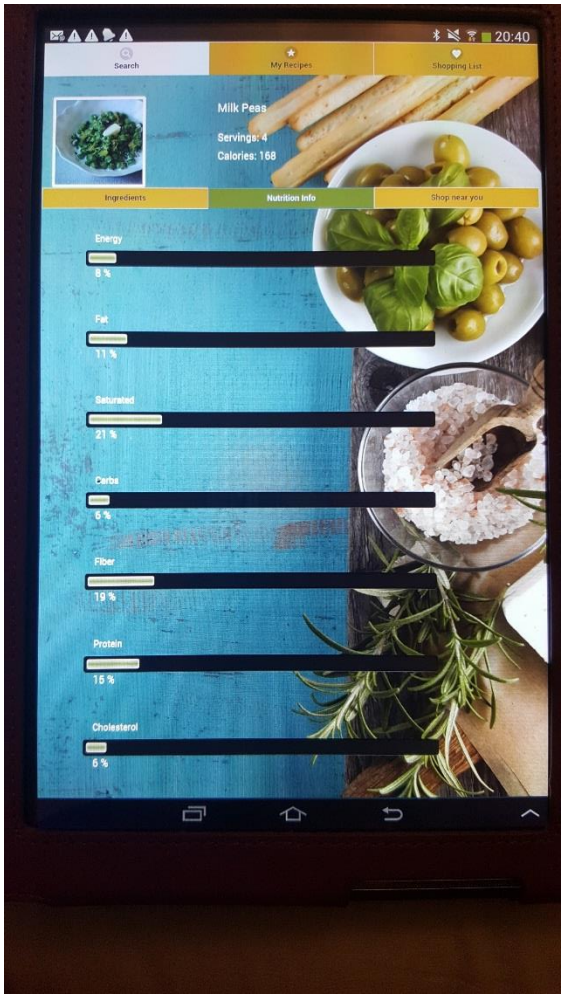
Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels



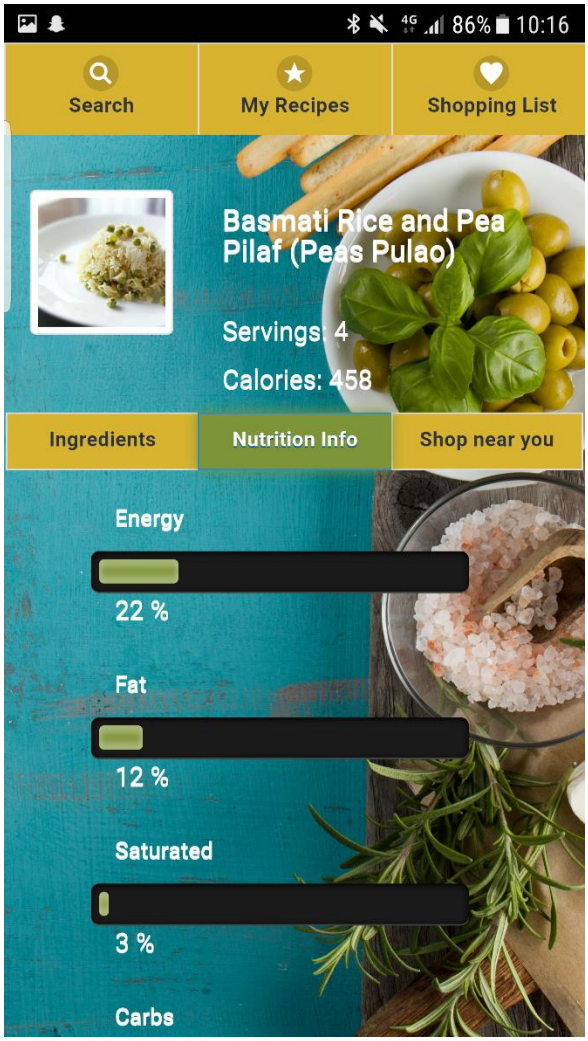
Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.



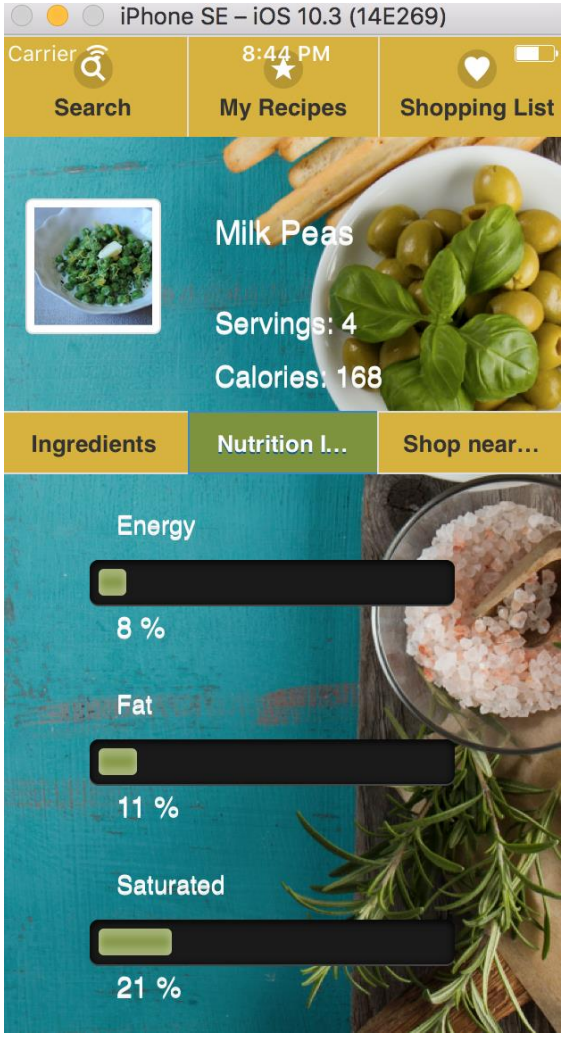
Samsung Tablet, Android 4.2.2, 7-inch display – 600x1024 pixels.



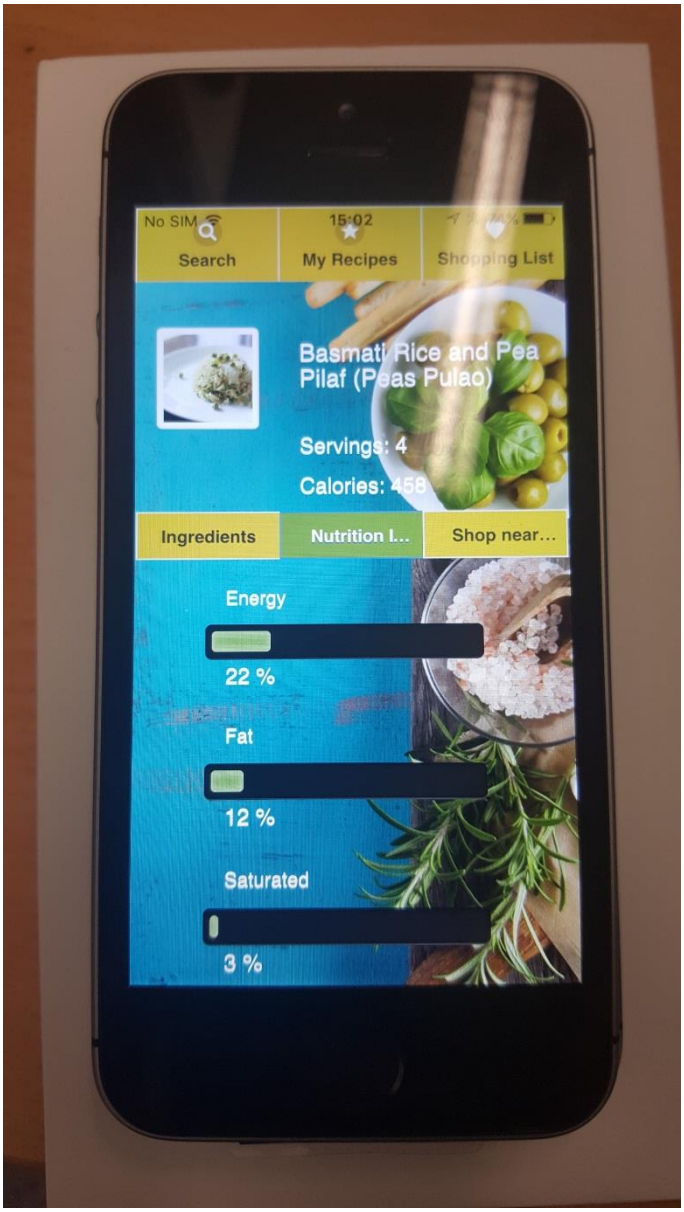
Samsung S6 Edge, Android version 7.0, 5.1-inch display – 1440x2560 pixels.



IOS SE, 10.3 (Simulator), 4-inch retina display – 1136x640 pixels.

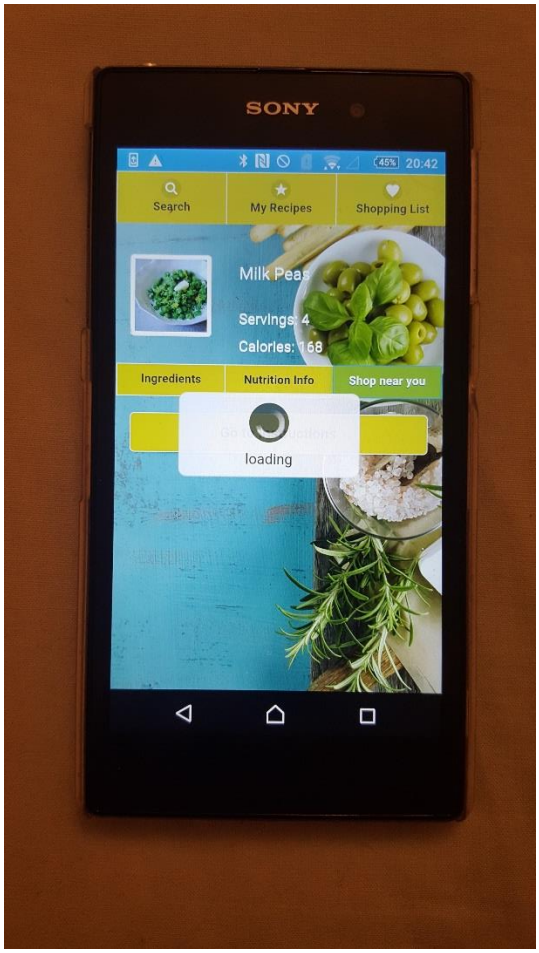


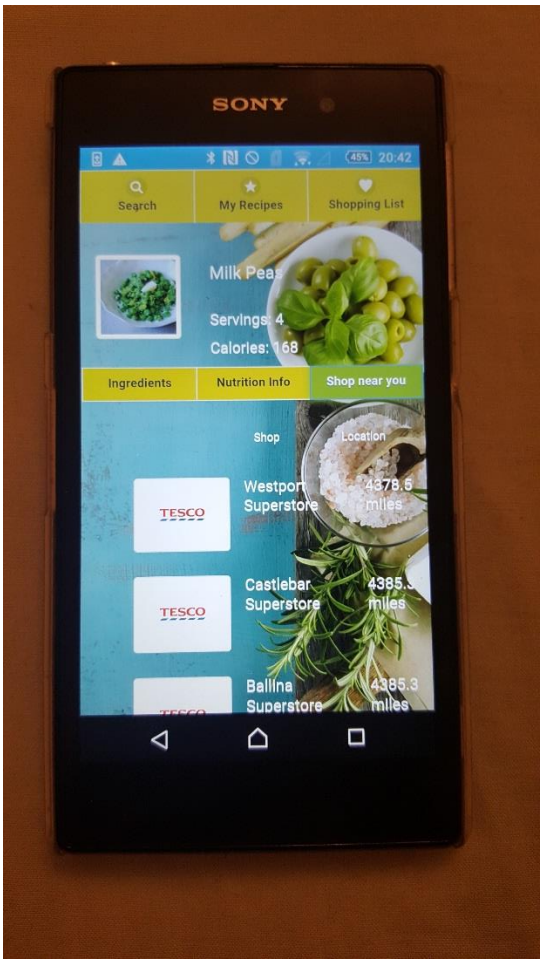
IOS IPhone SE, version 9, 4-inch retina display – 1136x640 pixels.



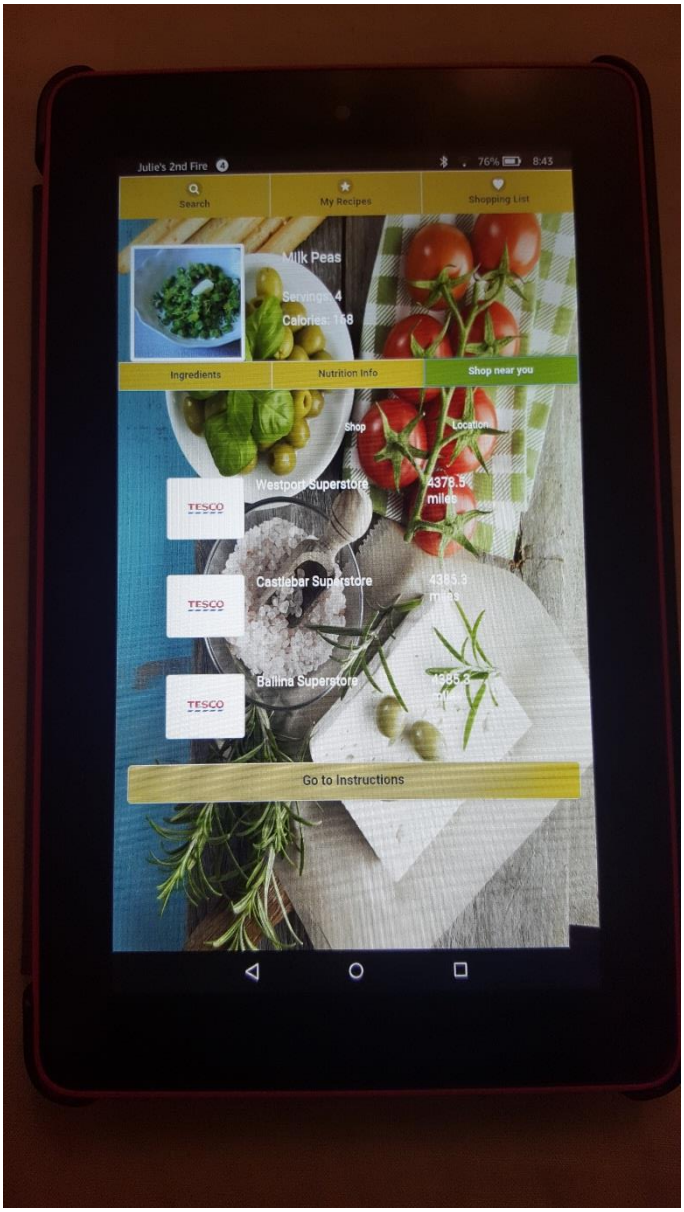
Appendix Ten

Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels

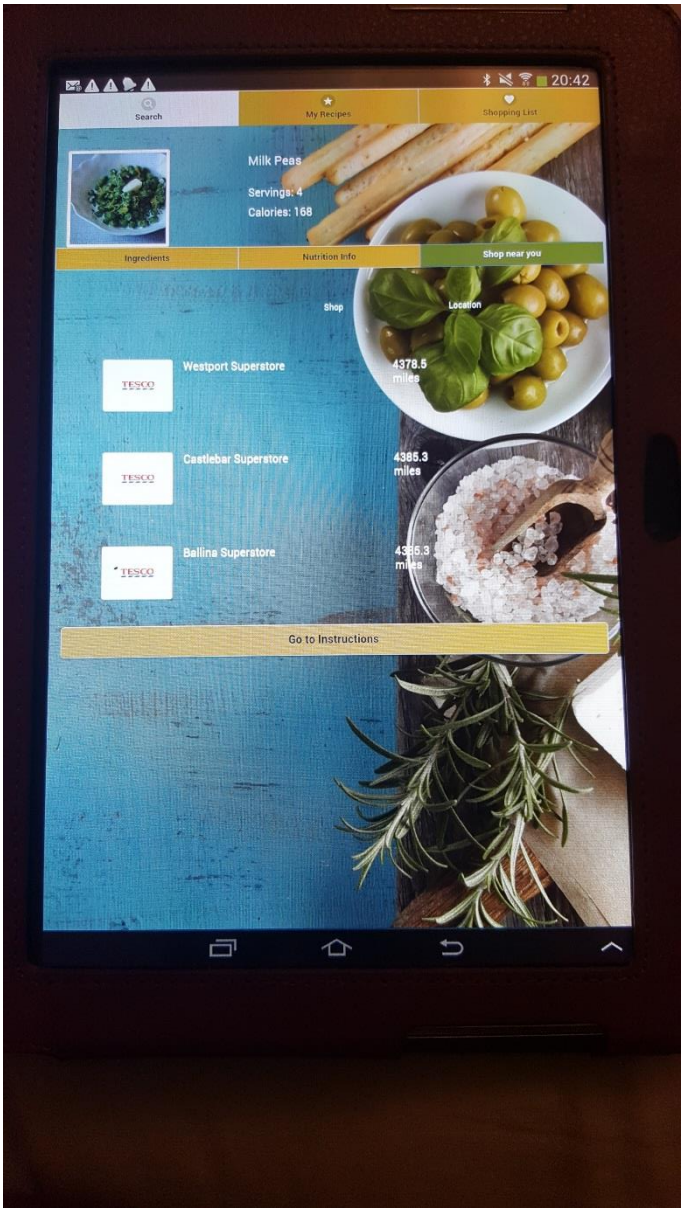




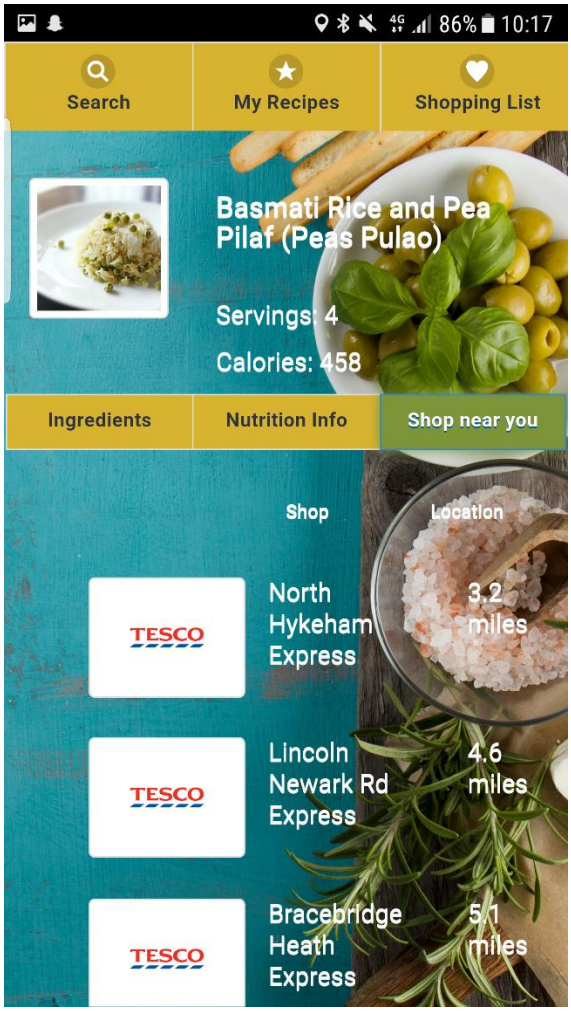
Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.



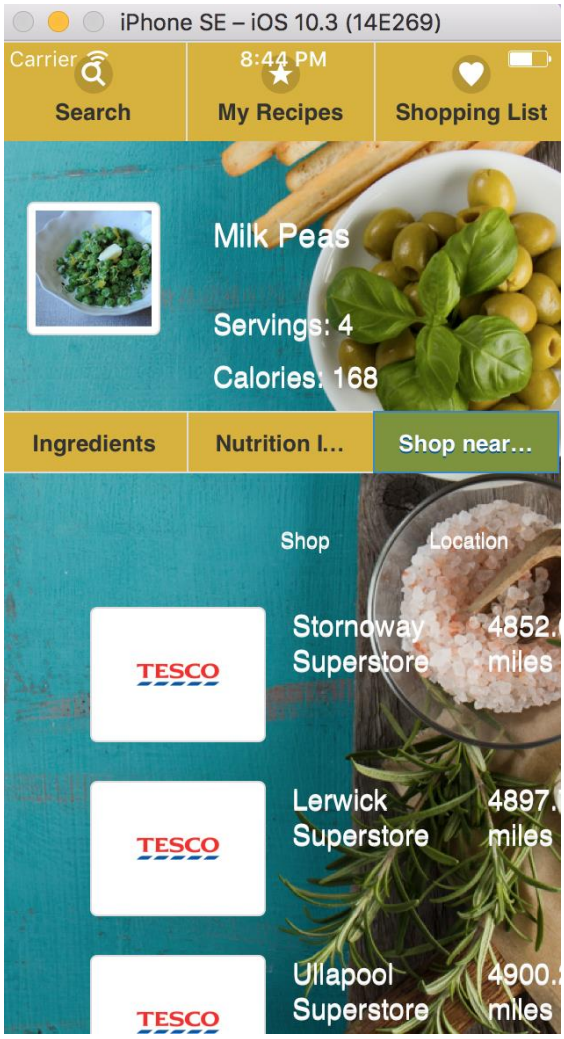
Samsung Tablet, Android 4.2.2, 7-inch display – 600x1024 pixels.



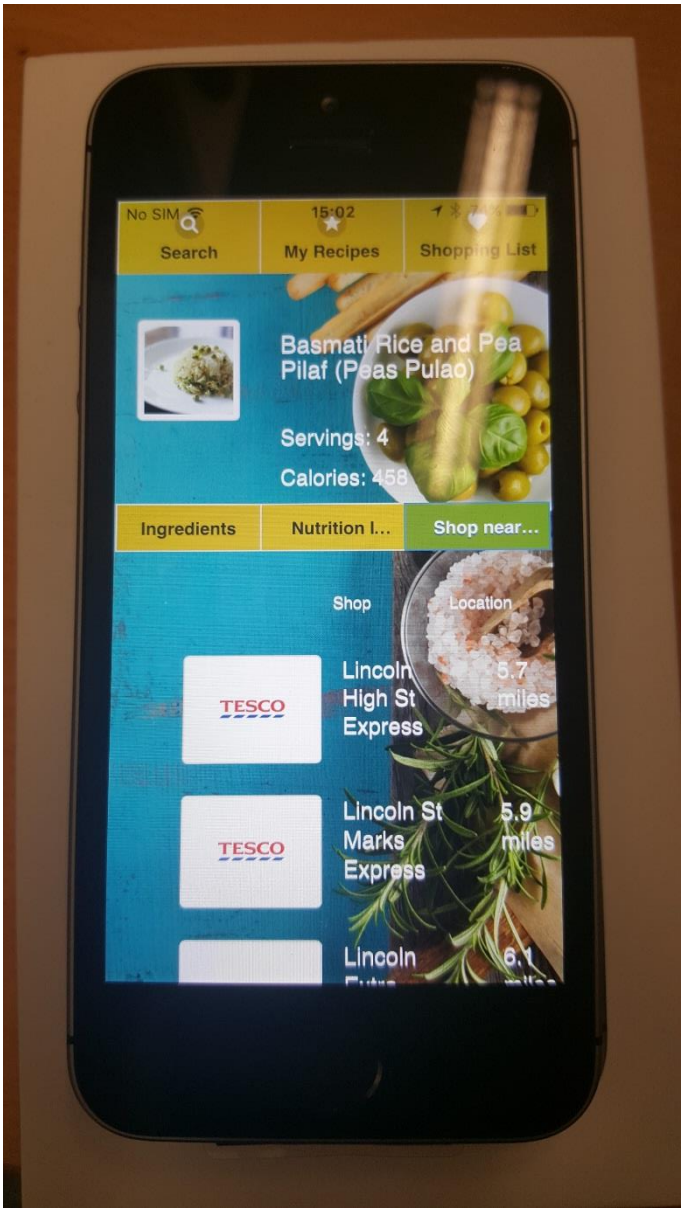
Samsung S6 Edge, Android version 7.0, 5.1-inch display – 1440x2560 pixels.



IOS SE, 10.3 (Simulator), 4-inch retina display – 1136x640 pixels.



IOS IPhone SE, version 9, 4-inch retina display – 1136x640 pixels.



Appendix Eleven

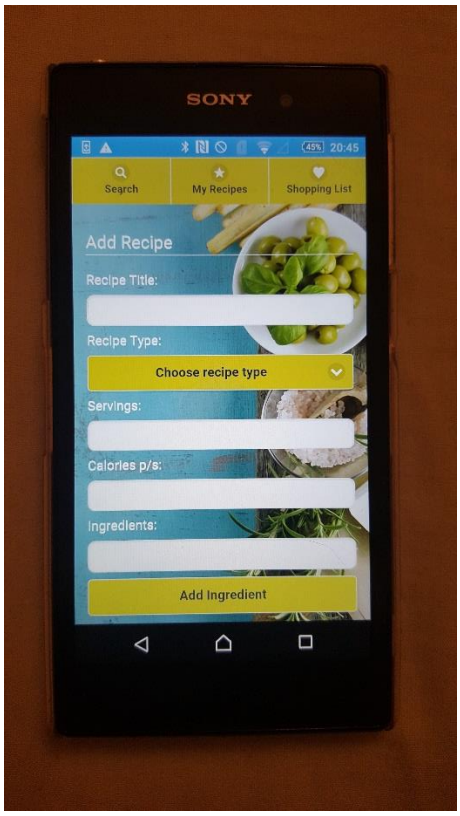
ID	Recipe Title	Image	Notes	Serving	Calories	Type
1	peas	asdfghjhgfds		2	123	Starter

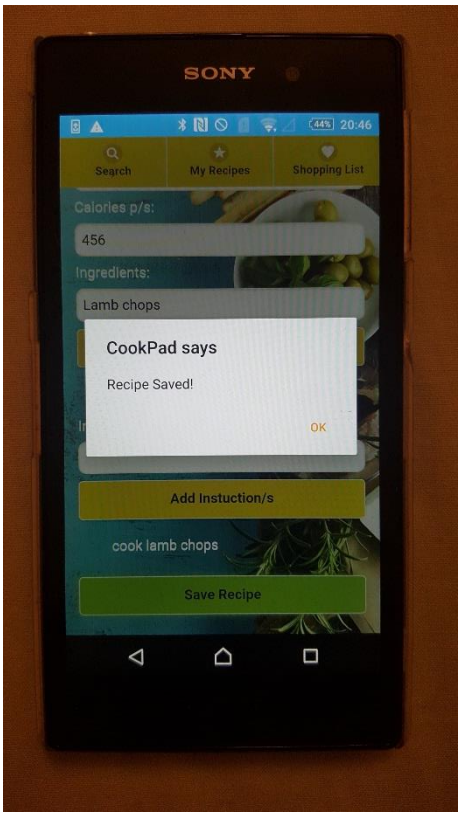
ID	Recipe Title	Ingredients
1	peas	peas
2	peas	water

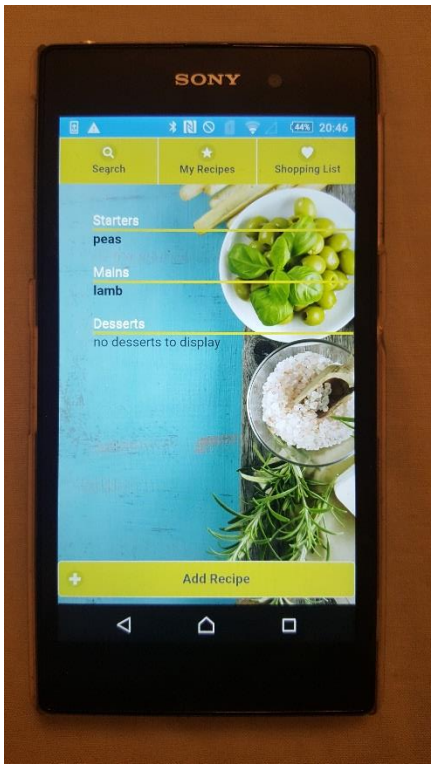
ID	Recipe Title	Method
1	peas	add peas to water
2	peas	boil peas in pan

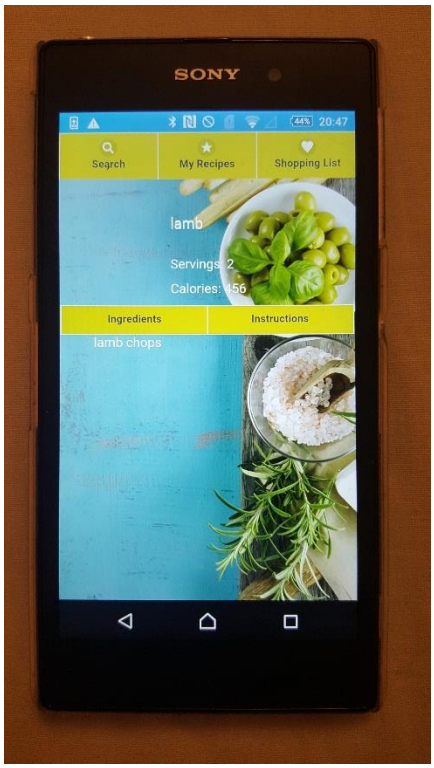
Appendix Twelve

Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels

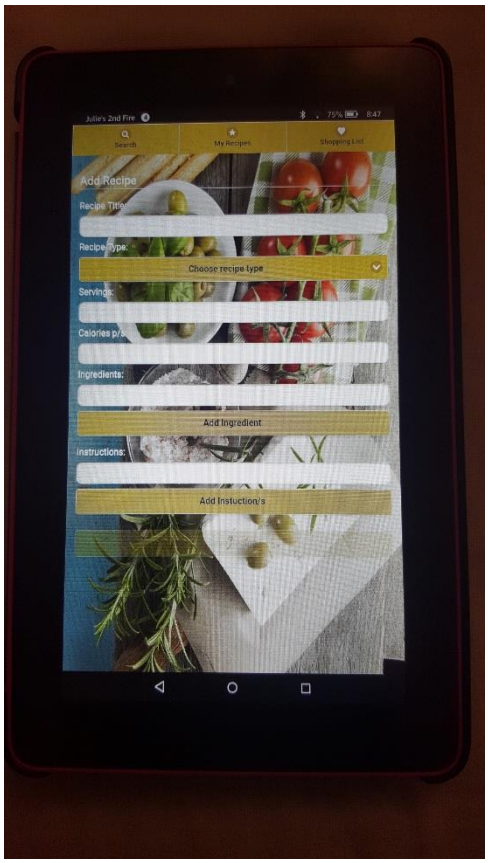


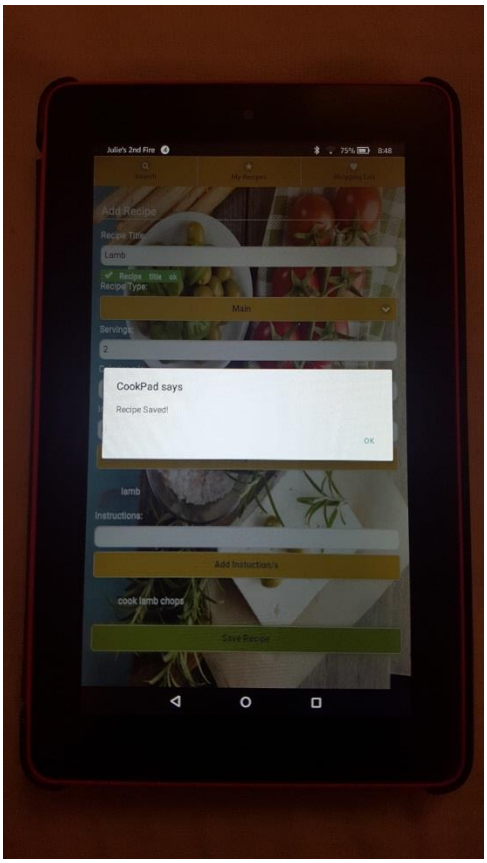


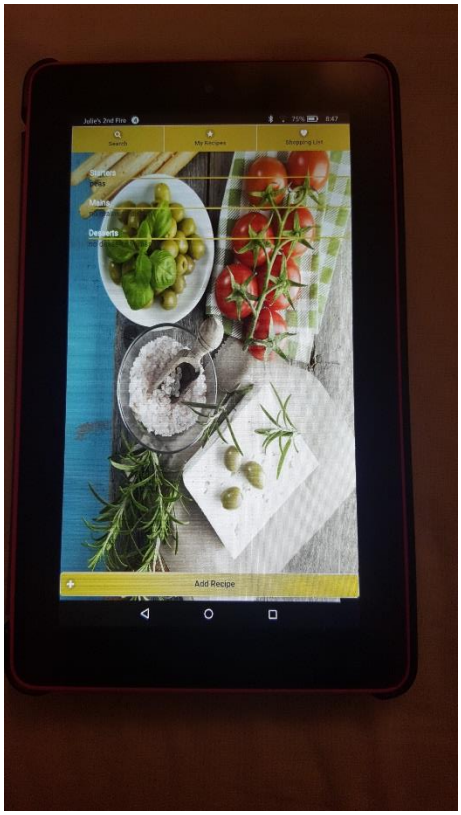


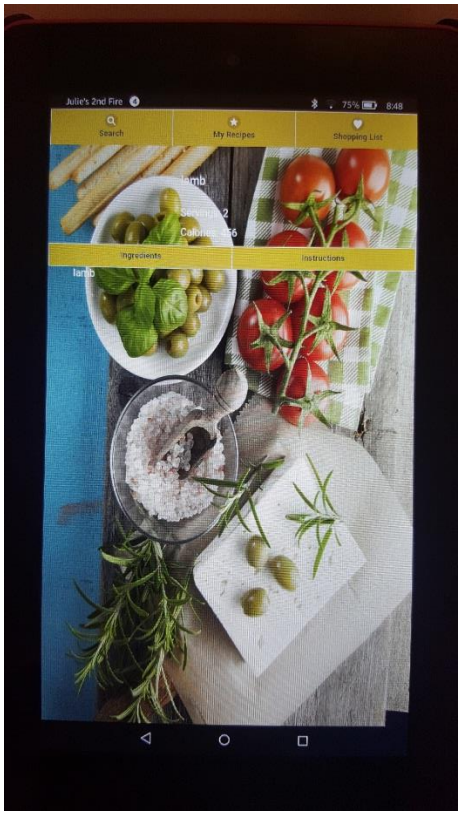


Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.

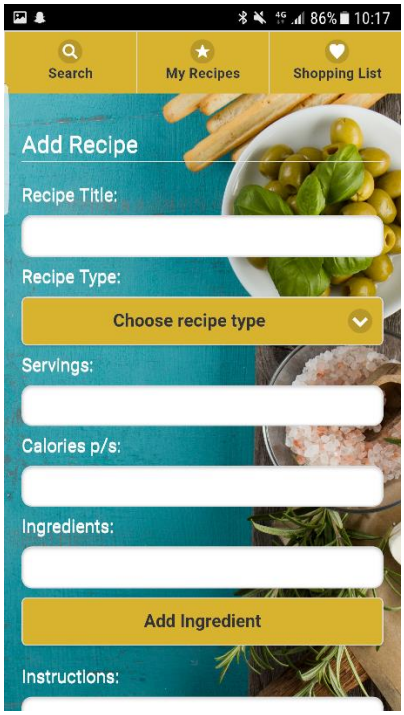




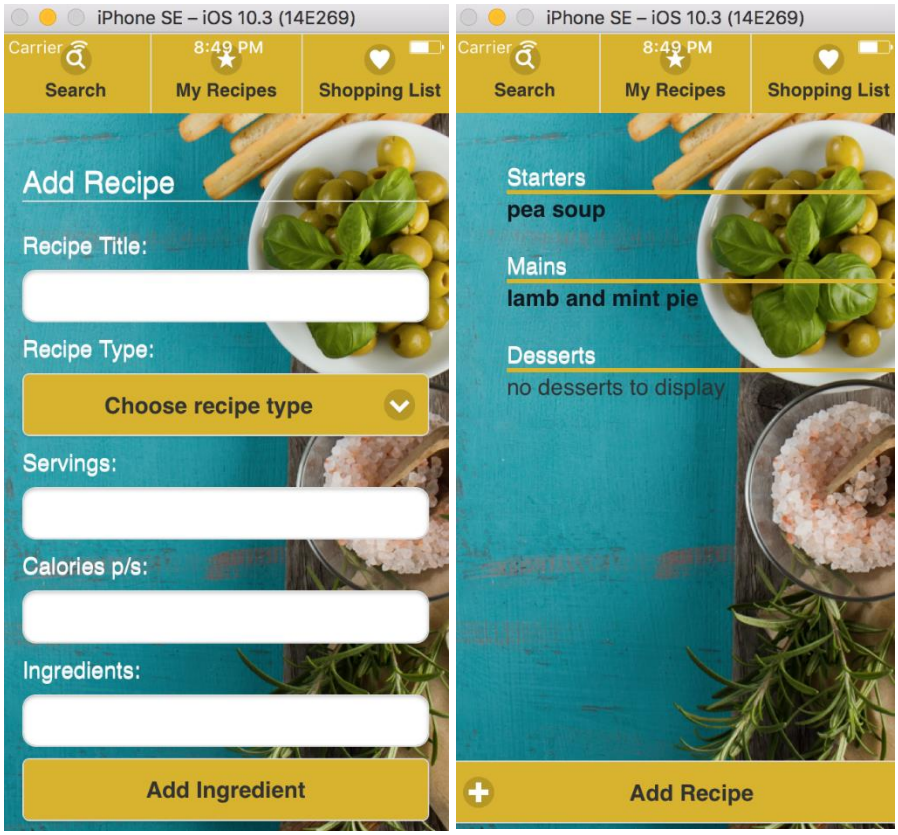


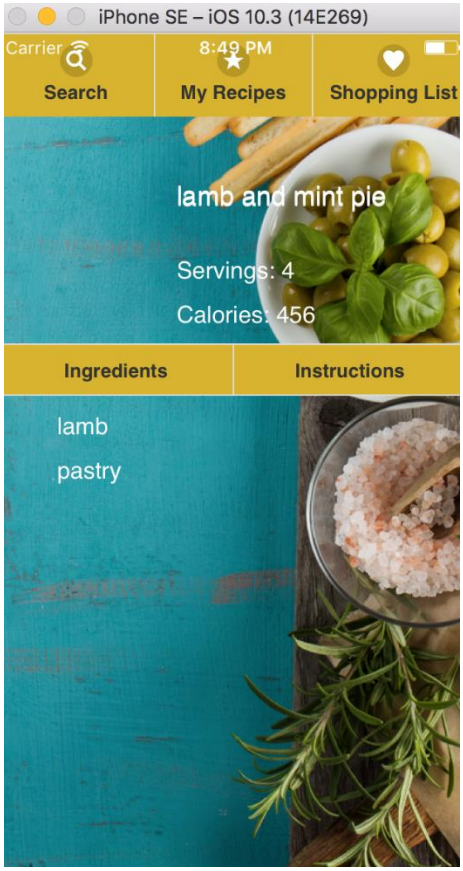


Samsung S6 Edge, Android version 7.0, 5.1-inch display – 1440x2560 pixels.

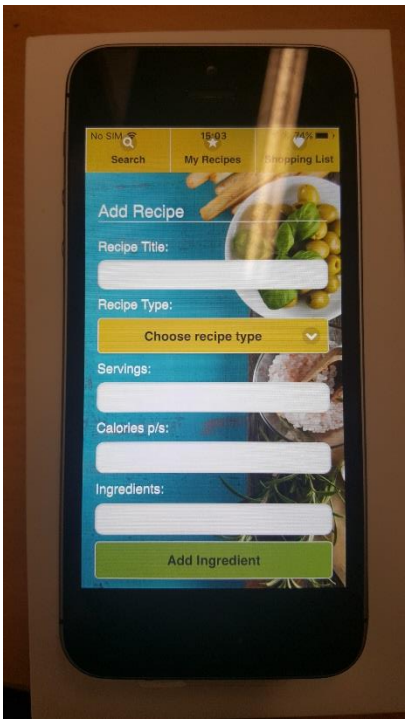


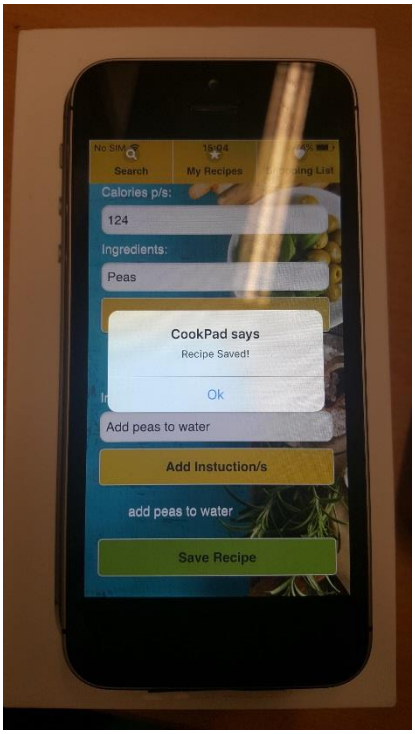
IOS SE, 10.3 (Simulator), 4-inch retina display – 1136x640 pixels.

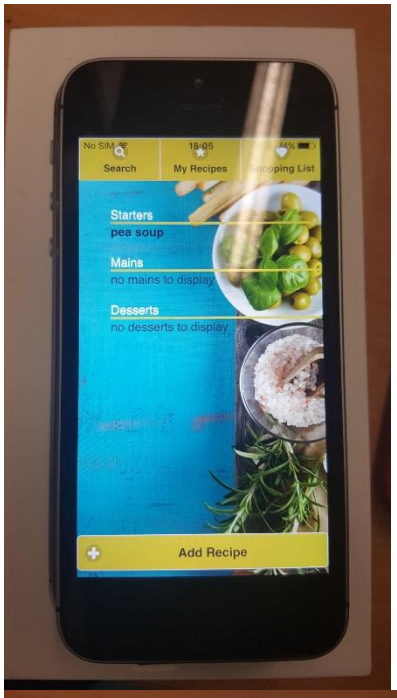


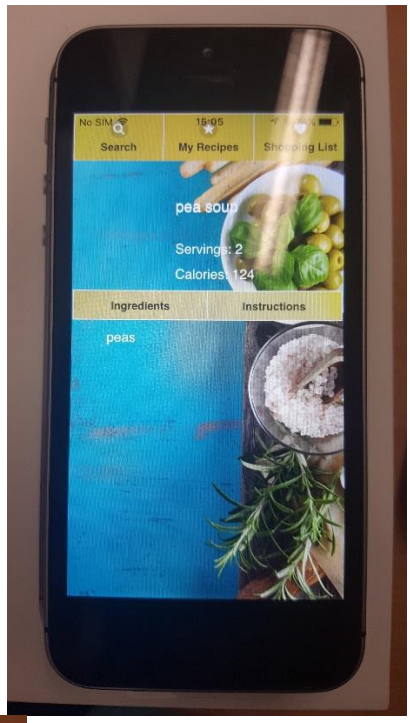


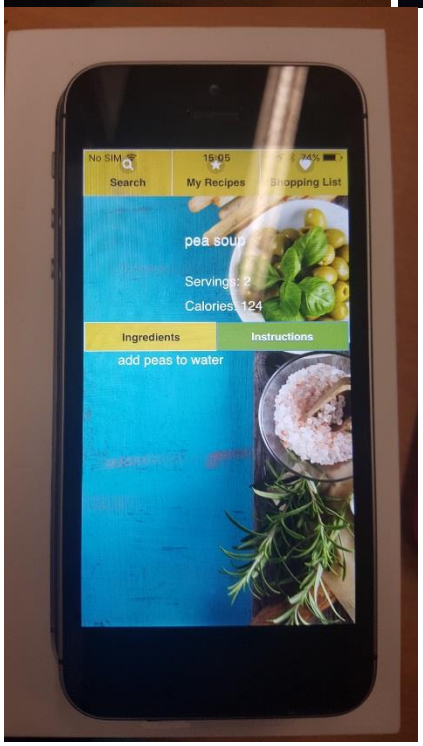
IOS IPhone SE, version 9, 4-inch retina display – 1136x640 pixels.





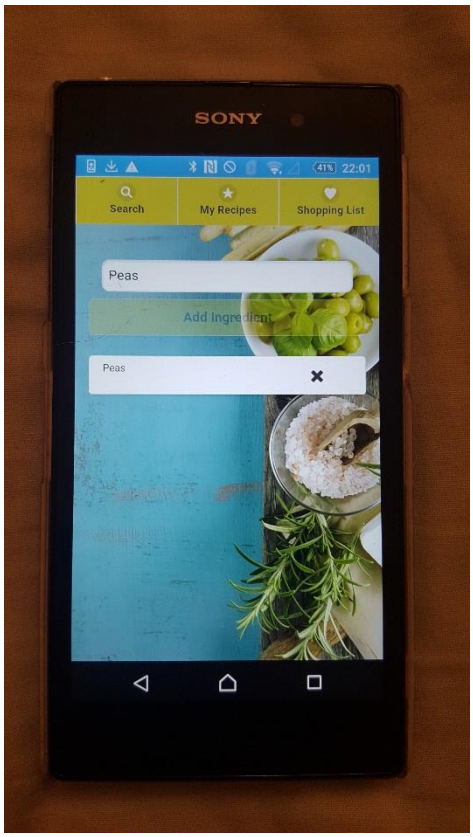

No SIM
15:05
Search
My Recipes
Shopping List
Starters
pea soup
Mains
no mains to display
Desserts
no desserts to display
Add Recipe


No SIM
15:05
Search
My Recipes
Shopping List
pea soup
Serving: 2
Calories: 124
Ingredients
Instructions
peas


No SIM
15:05
Search
My Recipes
Shopping List
pea soup
Serving: 2
Calories: 124
Ingredients
Instructions
add peas to water

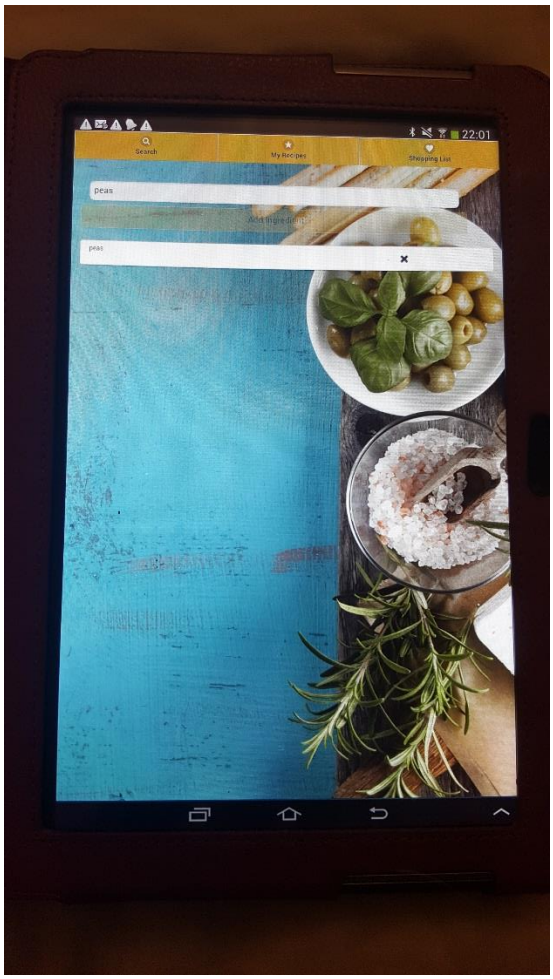
Appendix Thirteen

Sony Xperia Z1, Android Version 5.1.1, 5inch display - 1920x1080 pixels

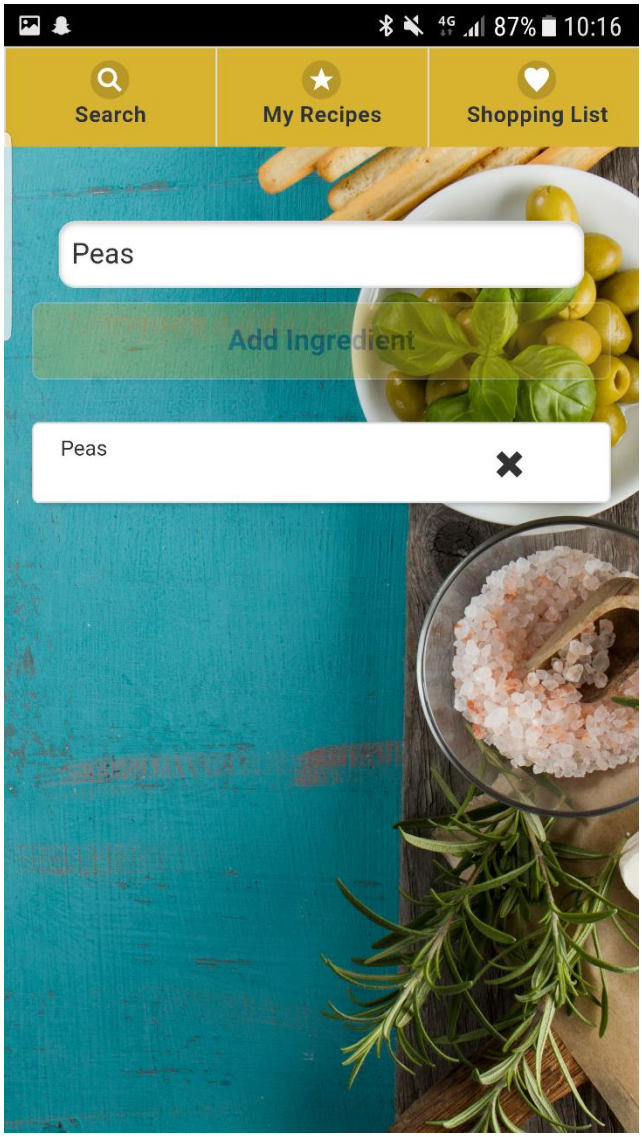


Kindle Fire, Fire OS 5.3.3.0, 7-inch display – 600x1024 pixels.

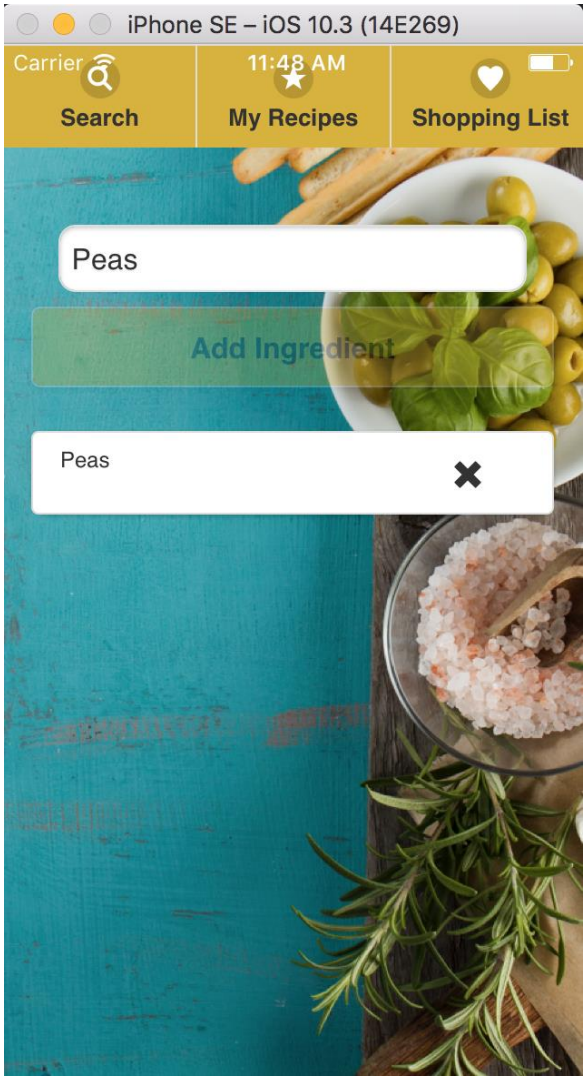
Samsung Tablet, Android 4.2.2, 7-inch display – 600x1024 pixels.



Samsung S6 Edge, Android version 7.0



IOS SE, 10.3 (Simulator), 4-inch retina display – 1136x640 pixels.



IOS IPhone SE, version 9, 4-inch retina display – 1136x640 pixels.