Chef Swipe
Final Project Report

TU858
BSc in Computer Science (International)

Stephen Halligan
C19332291

Paul Kelly

School of Computer Science
Technological University, Dublin

02/Mar/2023

Abstract

There is no doubt that as technology advances, things like books and newspapers are falling behind. [34] A large contributing factor to the downfall of print medium is the rise of eBooks and digital media. It's undeniable that technology is becoming a staple of day-to-day living, which leaves us with the question; what will happen to print media in another 10 years? With deforestation becoming a larger issue than ever before, [14] the world is looking at other solutions to this old style of sharing content, and in this case – recipes. Instead of continually printing new books which must then be disposed of once used, the globe is turning to technology to meet its' media-sharing needs. [8]

This project proposes a solution to the 'recipe books' or 'cookbooks' of old, whereby generally less than 100 recipes are stored in one large hardback book. While this method worked well for centuries, modern technology now allows us to store not just hundreds, but millions of recipes on a mobile phone or in the cloud. Hence, the idea behind this project is to replace bulky cookbooks with a single application that aims to meet its users' recipe sharing and finding needs.

This application will allow for users to interact with recipes, opening them up a new world of cooking they may not have experienced before. This gap in the market has been identified after researching many recipe applications currently available, and deciding that not many offer the level of interactivity valued in modern applications such as 'Tinder', [52] or 'TikTok'. [53]

As this application can retrieve data from the internet, users have access to a near infinite number of recipes at their fingertips. This would have been unfathomable as a mobile application 20 years ago, but with modern technology like cloud databases and APIs, this application will provide content on-demand to users looking for new and exciting meals to make.

Instead of sifting through pages upon pages of recipes and struggling to find the specific recipe you wish to make based on your individual needs and wants users of this application can swipe through a curated list of recipes filtered to cater to their allergies, preferences and requirements. Customisable filters and feed pages insure that users always find new recipes they will enjoy, in a fun and creative manner.

The community focused aspect of this application will allow users to share their own recipes, comment on recipes they have tried, and even follow their favourite creators. This social feature adds an engaging element to the application, alongside allowing users of different backgrounds and culinary preferences to connect and share their knowledge with other 'foodies'

With the world becoming more reliant on technology by the day, [64] we must adapt to the common challenges that come with these drastic changes. Hence, not only is this application a useful tool, but it is also an example of how we can embrace our reliance on technology, using it to enhance day-to-day living and promote sustainability. [65]

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

Stephen Halligan

04/Mar/2023

Acknowledgements

I, Stephen Halligan, would like to take this opportunity to give thanks to all those who have aided in the creation of this project.

I'd particularly like to thank Paul Kelly, my supervisor, for his invaluable assistance with all aspects of this project, from suggesting useful tools for UML diagrams to providing moral support while writing up the necessary logs and reports. Having someone who is passionate about and engaged in the development cycle of this project has been highly beneficial. After the countless meetings carried out either in person or remotely, Paul has been invaluable when it came to guidance and aid throughout this project's lifespan.

I would also like to thank Jonathan McCarthy, the Final Year Project coordinator, for his assistance in class and online in any way he can to help myself and other students. Having a coordinator like Jonathan who cares about and is enthusiastic about his students' work is very encouraging and inspiring to see.

Finally, I'd like to express my appreciation to those around me, including family and peers, who assisted in the application's testing process throughout the development cycle. This honest feedback on the applications many iterations on design decisions and features has been invaluable, and I firmly believe it has helped steer me in the right direction.

Table of Contents

1. Introduction

This section contains a summary of the planned project components; along with the process of documentation and development. The fundamentals of the application will be covered in this chapter, from the background of the project, up to proposed plans for development methods.

This project concept revolves around the belief that cookbooks/recipe books have not been brought into the 21st century. This project would create an interactive experience for users, allowing them to find new recipes whilst giving them the opportunity to share their own recipes.

The mobile application would offer a platform for users to share their passion for food in a unique way, based on the popular "swipe left or right" model seen in many dating applications; if a user wishes to view and save a recipe, they swipe left. Otherwise, they can swipe left to continue searching. Users would be presented with the opportunity to view ingredients and methods for saved recipes, along with sorting and filtering the database using specific dietary requirement filters.

The end-product would be somewhat of a social network for recipe sharing and discovery. There is very little in the area that solves the issue of locating recipes based on specific dietary filters, without having to compile a list from multiple different websites and databases from all across the internet manually. Hence, this application would be of great benefit to people with this specific use-case.

This idea would be successful and useful to users due to the fact that there are not many other alternatives in the area of social platforms for recipes, from researching the topic. It was inspired by the search for recipes for dinner parties; a significant amount of time was spent compiling different recipes from numerous different websites, such as www.bbcgoodfood.com [1].

It is particularly difficult to find recipes catering to guests with dietary requirements such as vegans and vegetarians. [29] This was when the gap in the market was noticed for an application which pre-compiles these recipe lists and presents them to users in an interactive and attractive manner.

Similar solutions to this problem are certainly few and far between, with exceptions such as an application the name of "SuperCook – Recipe Generator" [2]. Whilst this solution seems to be effective, with over 1 million downloads on the Google Play Store; the UI could be improved upon, and the way in which users interact with the application seems rather dated.

In this project, cloud solutions such as Google Firebase to host a database of recipes, along with user authentication allowing for logging in and out easily. Firebase is known for its ease of use and implementation [3], and the copious amounts of documentation and support for the product makes it an ideal solution for this project.

A project plan will be delivered first, outlining how different aspects of the application will function in detail, alongside multiple wireframes. This project plan alongside wireframes will allow me to focus on development of the primary deliverable; the complete application.

A weekly log of work carried out on the project will also be created, which will be supported by regular commits to a GitHub page. These constant updates will showcase the lifecycle of the project, from the initial wireframes and intentions to the completed product.

Due to the lack of research into this area, and the small amount of applications in the field of recipe compilation, we can confidently state that a gap in the market has been

identified; providing the service of bringing recipe books into the 21st century. The technologies outlined in this report will enable the development of the proposed application.

This application would provide plenty of opportunity for expansion, with the possibility of the addition of machine learning algorithms. There is also room for the addition of other features to aid in streamlining development, such as access to the numerous amounts of recipe APIs found online [30]

This application will have one main page, or 'feed', similar to applications such as Instagram [9]. This feed page will display new recipes to the user each time they open the app similar to how newsfeeds work. However, instead of scrolling down through a newsfeed as a user might on an app such as Facebook [31], users will be presented with the opportunity to swipe left or right.

Whilst this is a small change, it has been a highly successful method of navigation in other applications such as 'Tinder' [32], where users swipe left or right on a person based on their appearance, short description and overall first impressions. As the culinary world, like all things is developing a presence online, this approach of 'first impressions based on an image' can be applied to a dish.

For example, if a user is presented with a dish; "Homemade Coleslaw", they are given the option to ignore this recipe and move onto the next by swiping the image to the left. However, if they wish to 'like' this dish as is a feature present on many other popular social media platforms, they can swipe the image to the right.

Upon performing this action, the recipe details are pulled from the Firebase database, and presented to the user. Information such as ingredients, method and cooking time will be available. The set ID for the recipe will then be saved in Firebase under the user's account.

By saving recipes by ID under the user's account details, we will be able to retrieve information regarding recipes that have been saved for the user. These recipes that have been saved can then be compiled into a list, and presented to the user for future viewing. This will be a beneficial feature, as it reduces the requirement for user input to find recipes that have been liked in the past.

Utilizing firebase in order to store recipes will also provide another benefit in the form of allowing users to manually save their own recipes to the application. This will provide yet another level of engagement for users, as it provides a platform for them to share recipes and ideas with their friends and followers. As opposed to merely pulling recipes from an API, this feature will help to make users feel like they are a part of a wider community.

As an extension of user submitted recipes, the concept of 'friends / followers' is brought up. This feature would be a fantastic way to keep up to date with a user's favourite chef, baker or home-cook.

Similar to Twitter's follower system [33], we can prioritize recipes submitted by a user's friends to the top of their feed page. Doing so would create an opportunity for users to craft their feed based on dietary requirements, or food preferences. For example, if a user does not eat meat, they will be presented with the chance to follow creators that only post vegetarian dishes.

This provides users with a more 'personal' experience, and filters our data to recipes that may be of more interest to the specific user based on their eating habits. The benefits of content filtering do not end at personalisation however, as we can also use these filters in order to moderate content.

As the application will allow for user-submitted recipes, we must ensure that no inappropriate content is submitted. Whilst this may have to be moderated manually, [10] we can also use filters in order to block or remove certain content that gets caught within the set filters. [11] This technique will be very useful when it comes to moderating content, as the goal for user-submitted content is always to make the system run as automatically as possible, without need for human intervention.

Whilst this may cost slightly more in resources, it will cut down greatly on the requirements to hire employees to go through submissions one-by-one and flag the content which may be inappropriate.

This project will outline the steps taken in order to create a mobile application similar to the proposed; from setting up android studio, our IDE (integrated development environment) to linking Firebase, and coding the project. In our case, Firebase can be easily integrated into Android Studio for making use of their authentication & storage features in a quick and efficient manner [12].

This project will be developed using Java, the massively popular general-purpose object oriented programming language [13]. As will be outlined in this report, Java was chosen thanks to its support from a wide community of developers, and it's security. [14]

Alongside Java, markup languages such as XML will also be used within this project to style our application. XML was chosen thanks to its extensibility, and it's fantastic compatibility with Java. [15]

Documentation for this project will be made available through this report, alongside the author's GitHub page. [16] GitHub is a fantastic tool for documenting the progress of a

project, as it allows users to track changes made to code and project contents. These changes can be tracked back to the specific time and date of the 'commit' to the repository. This makes GitHub a great choice for keeping track of a project, and staying up to date with changes that are made over its lifespan. [17]

1.1. Project Background

It's no doubt that with the advancement of technology, things like books and newspapers are getting left behind. Whilst eBooks are booming in popularity, with over 191 million eBooks sold in 2020 alone [8], Cookbooks/Recipe books are a growing market, however they have not taken over standard paper cookbooks quite yet. [18] To aid in the digitalization of this medium, this application will provide an interactive social platform that serves as somewhat of an 'online cookbook'.

Though paper and hardback books continue to control the market, with roughly $5.5 billion in sales in 2019 alone [19], with sales of eBooks expected to grow massively in the next 10 years, projected to reach $32.19 billion by 2032 [20], it's safe to assume that this market has a massive potential to take off.

The benefits of creating one big 'eCookbook' are clear. Firstly, the production process of paper cookbooks is extremely harmful to the environment. Deforestation is a massive issue in today's world. Paper books contribute to up to 14% of deforestation currently. [21] This has a huge negative impact upon our environment, without even taking into account the rest of the production process for these books which involves a large amount of chemicals, energy, ink and water.

Whilst eReaders and mobile phones do take a significant amount more carbon to produce, these devices can be used again and again to read a near unlimited amount of books. [22] As a result, it can be safely assumed that this app will benefit the environment by reducing the need for the resource-intensive production of paper- books.

In addition to aiding the environment, storing recipes online and displaying them via this application. Whilst regular cookbooks may hold around 100 recipes, with the storage functionality of Google Firebase, this application has the potential to store thousands of recipes, with the added benefit of being lightweight; merely taking up storage space on a mobile device instead of being required to keep a physical copy of a cookbook.

As an extension of this advantage, this application would also provide the benefit of being accessible from anywhere in the world, on the go or at home. Instead of being required to carry heavy and often expensive books; users of this application would be capable of gaining access to these thousands of recipes on demand.

The benefits of this are clear, as we can easily share recipes and ideas virtually without the need for the dated cookbook. Also, needless to say, a dynamic application gives us the opportunity to provide constant free updates, instead of requiring users to purchase a new physical recipe book every few weeks/months.

Technology is no doubt a crucial part of our future; physical books, along with newspapers will almost certainly be a thing of the past in 10-20 years' time, due to the fact that they are 'unable to update like apps' are. [34] Hence, this application would not only be fun and handy, but also yet another stepping stone towards a more digital world.

1.2. Project Description

As stated, this project will be developed as an android application to be deployed at first on android phones, with the hopes to later expand into the iOS market. The application will be a social network for strictly sharing recipes and images of recipes, unlike other platforms such as Instagram [9] which are not tailor-made as a 'cookbook' of sorts.

The application will make use of a recipe API in order to compile a list of recipes, uploading them to a Firebase database so that users have a near endless supply of recipes to swipe through to their hearts content. Access to a recipe API such as mycookbook.io [23] allows the application to stay up-to-date with the latest trends; and as a result, drive up engagement.

Using a database service such as Firebase will also allow us to provide a 'user uploaded recipe' service to users, allowing them to submit their own recipes to be displayed on the app to friends, or people that may be interested in their content. As mentioned in the project introduction, the process of user-uploaded content would provide a deeper level of interactivity with the application. Ultimately, interactivity leads to engagement, which is the end-goal of any application.

Making an application engaging is very important for its survival. To make an application that is fun to use and stands out from the crowd, a gap in the market must be found, which in this case; could involve making design choices which are not often in the world of recipe applications. In this project, it was decided to employ the tried-and- tested model of 'swiping' on a stack of cards in the application.

This model is very simple, would provide a way for users to like or favourite recipes they enjoy. The model would involve a recipe image, with a description underneath as a 'card', which the user can swipe left on to skip; or swipe right on to 'like' the post, and be provided with further information on the recipe.

This design model is often utilized in dating apps, however there is no reason that this approach could not also be very successful in the world of cookbook apps. Due to its interactive nature, the goal is that not only adults could enjoy this application, but would help the whole family to get involved; enabling parents to allow their kids to pick what to have for dinner (with realistic filters to make sure they're getting their nutrients)

Users will also be able to 'follow' their favourite creators on the app, adding them to their friends list. This 'follower/friend' option will be implemented so that users can keep track of new recipes as they are posted, allowing them to personalize their feed page. Instead of the user seeing a default stack of recipes, recipes shared by their friends will be prioritized when they access their feed page. As has been made apparent by applications such as Instagram [9], enabling users to comment and like on submitted recipes will provide a community-driven 'rating' system.

1.3. Project Aims and Objectives

- The aim with this project is to create an application that brings cookbooks into the 21st century.
- A goal of this application as an extension is to create a fun platform for users to interact with one another.
- It is an objective of this project to ensure a safe environment for users of all ages, as this is a social platform.
- User submissions will be stored on the cloud, reducing the need for locally saved items clogging up storage. This will be of direct benefit to users.
- It is important for this project's lifecycle to ensure readable documentation is kept at all times, outlining processes and methods employed.
- This project aims to utilize a live database, which can be continually updated with new information.

1.4. Project Scope

This project's scope is somewhat of a social networking platform, which allows users to communicate by sharing recipes; essentially an online cookbook, it is not strictly an image sharing platform, as user submissions must be of benefit to the app, i.e. when posting, users must contribute instruction or feedback regarding their post.

Users are not required to post, and as an extensions are not required to follow users in order to find recipes. The basic idea is a platform whereby users can be exposed different recipes, with the added benefit of being able to follow their favourite creators, keeping up to date with their new recipes and content.

As the application would be primarily for mobile-use, it would not be hardware- intensive. For development, all that would be necessary would be a laptop that supports android emulation, also allowing for possibility of deployment to iOS devices. Of course, the Android Studio IDE would be a necessity for development, as it allows for app development in a simple and well-documented manner.

Finally, the aforementioned cloud solutions would be a necessity for this project. Google Firebase will be utilized in this case, due to it fitting with the needs and requirements of this application. The benefit of using Firebase is the excellent support online, and of course; as it has a free version the application would have minimal running costs.

The lack of running costs posed by this application would result for an experience which could be made available to any user with no upfront cost on their behalf, which would directly compete with other options flooded with advertisements and/or a significant amount of paid features [79]

1.5. Thesis Roadmap

- Chapter 1

  Introduction and background of project, outlining the basics of the application. This chapter gives a brief, not too in-depth overview of the proposed project components. The introduction and background of this project possesses the plans for the development and documentation of this project.

- Chapter 2

  Literature review into existing and similar applications to the proposed project. This chapter outlines the research carried out for this project. Applications related to this project are assessed in this chapter, with research done into any other similar solution which is currently in use.

- Chapter 3

  Experiment Design is the layout of the systems employed in this application. An outline of the project's design development process is provided in this chapter. System Design also covers UML diagrams of the project, along with a showcase of the different iterations of design testing and evaluation carried out to improve upon the application UI.

- Chapter 4

  Experiment development is an important phase, whereby the project development cycle commences. This chapter provides an overview of this cycle and the procedures involved in the project's development phase. The program's design components are implemented during this phase, alongside software-development methodologies utilised.

- Chapter 5

  Testing and evaluation is a phase of development outlines the evidence of the applications functionality, and ensuring that the application satisfies the demands and expectations of its users. This chapter outlines the testing model applied, alongside the information gathered throughout the rounds of testing.

- Chapter 6

  Conclusion and future work provides information regarding the future lifecycle of the project, including the plans for continued development of the application. This chapter also provides a conclusion to the current state of the application;

2. Literature Review

2.1. Introduction

This chapter provides an overview of the research conducted on existing and similar applications to the proposed concept. This includes looking into any other similar solutions that are currently in use, as well as researching technology that will be employed in this project.

Outlined in this chapter are numerous solutions to the problem I have presented; whether it be in print format, or a digital application similar to this project, 'ChefSwipe'.

This chapter will assess these solutions, and review them based on positive and negative aspects they possess. If any inspiration is drawn from these existing solutions, it will be referenced here and considered in Chapter 3, System Design.

Final Year Projects from past students will also be studied and research in this chapter, again, reviewing them and evaluating them based on their value, pros, and cons. This process is very helpful for the development of the project, as it aids in highlighting where time should be spent to improve upon downfalls of past projects.

2.2. Alternative Existing Solutions

2.2.1 Standard Print Cookbooks

An obvious existing solution to the problem is of course recipe books and cookbooks. With the oldest cookbook said to have dated back to 1700BC [24], it is difficult to argue that this medium will cease to exist any time soon. However, as is the case with technology, the way that the contents of these books are delivered can be made more efficient and convenient.

It is no surprise that print books still dominate a massive chunk of the market [25], and are only growing in popularity. 2021 saw a massive 42% increase in cookbook sales when compared to 2020's sales numbers. [26] With sales for print cookbooks at an all- time high, this solution is very old and will be used for many years to come.

An advantage of this existing medium is the fact that, as previously mentioned, the general population tend to prefer physical copies of books and as an extension, cookbooks, as opposed to digital alternatives. This is evident in many articles, including CNBC's research into the area of physical versus digital books. [37]

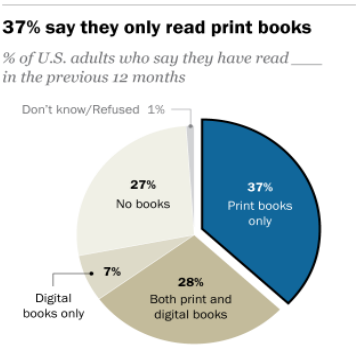
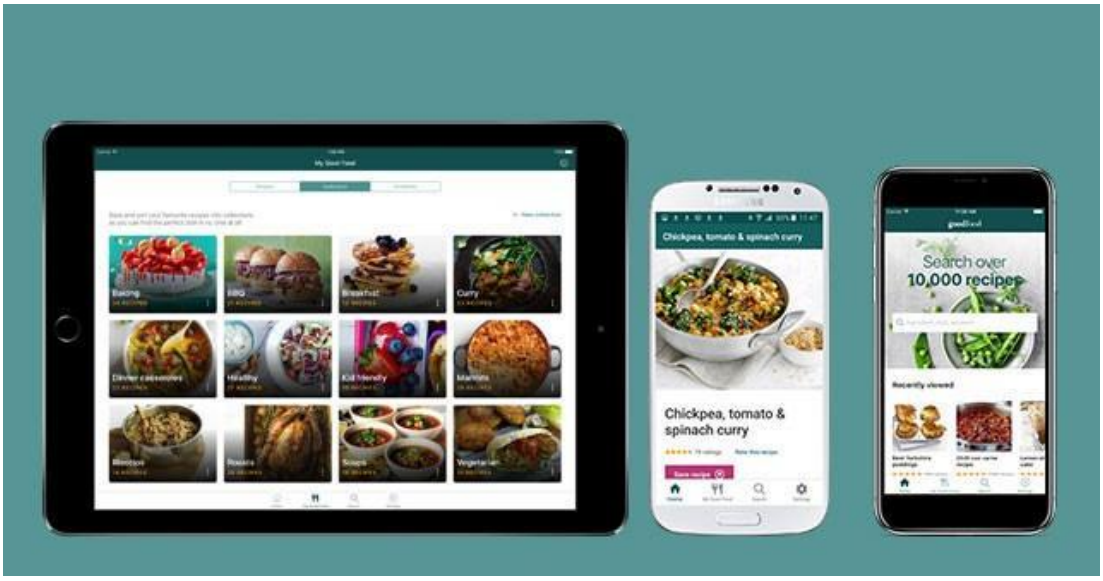


Image courtesy of Pew Research Centre [38], which showcases the preference for physical books as opposed to eBooks.

2.2.2 BBC Good Food

Another popular solution is the BBC Good Food app. [39] This application has a huge library of recipes, with currently over 10,000 recipes available to users of their app. These recipes have been curated by not only famous chefs, but also by users, which does not seem to be particularly common in the world of recipe applications.

Thanks to this applications massive popularity and funding, it seems to be performing very well on the Google Play store, [40] with over 500,000 downloads at the time of writing. Averaging 3.8 stars, this application also seems to perform well in real-world testing. However, there does seem to be some recent poor reviews with regards the 'save' feature on the application being poorly implemented. As this is clearly a feature that is very important to users of the BBC Good Food application, it is critical that it is well implemented in this project.



In its current form, this application allows users to filter recipes by 'collection', or dietary requirement. For example, there is one section dedicated to Vegetarian foods, another dedicated to BBQ and so on. This gives users a lot of choice when it comes to selecting their requirements and preferences. However, the amount of options presented on the page is rather overwhelming as seen in another customer review. This project aims to significantly reduce the overwhelming design seen in this application, by presenting users with 1 recipe at a time; as opposed to numerous collections with cluttered UX, scroll bars and text boxes.

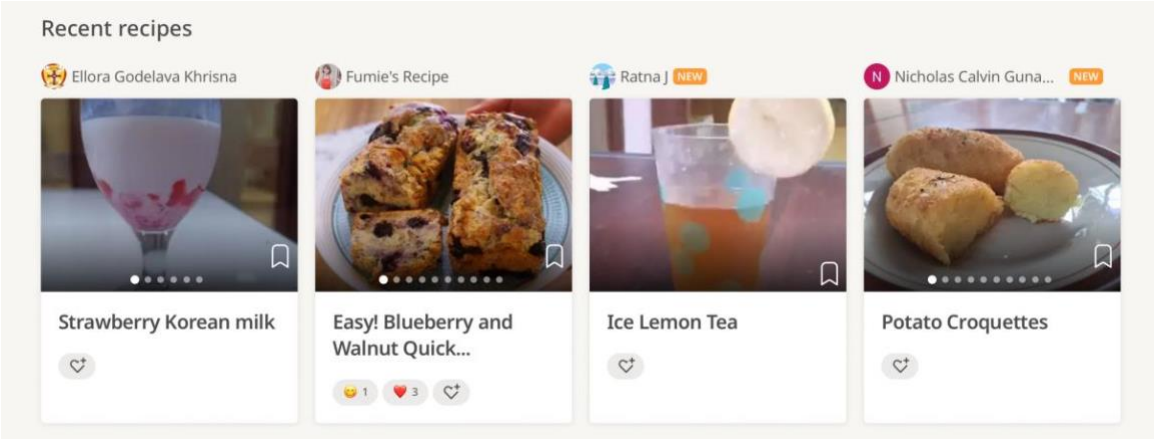
This application works on subscription model, whereby the application is free to download, however using the application to its full capabilities requires a subscription. Upon reading reviews and utilizing the application, it was clear to see why customers were becoming frustrated with this selected business model. Whilst the application does offer a free trial, customers can become quickly frustrated by features locked behind paywalls within the application. Whilst monetization is very important, especially in mobile development; [41] in the long-run it is not worth losing the customer-base in favour of earning extra short-term capital.

A significant advantage of using BBC Good Food is its precedence as a supported and widely popular service. This ensures support for years to come, along with ensuring a big community behind the service. Unlike smaller applications and websites, BBC Good food is almost guaranteed to stay up-to-date with the latest trends and changes in the culinary arts space.

With this in mind, and putting the outlined disadvantages aside, this application does achieve the goal of bringing cookbooks into the 21st century. Overall, this app seems to be a big success; and based on the numerous 5-star reviews, the app appears to be performing very well.

2.2.3 Cookpad

Cookpad is a website, and mobile application which performs similarly to BBC Good Food. Upon opening Cookpad, users are presented with a page full of recipes which have been recently submitted by users. [42] These recipes can be reacted to with emoji, saved for future viewing, and clicked into in order to view ingredients et cetera. An example of the home page can be seen in the screenshot below.



The website also includes 'tip sections, where users can submit cooking tips, and a 'Cooksnap' section where users can publish pictures of food they have created, thanks to a recipe from the site.

A significant advantage of using this application and website is that there are not many other options out there, for a social media on which people share recipes and general culinary knowledge. Websites like this have somewhat of a monopoly over the space, as there are very few options out there which provide the service of Cookpad, or ChefSwipe.

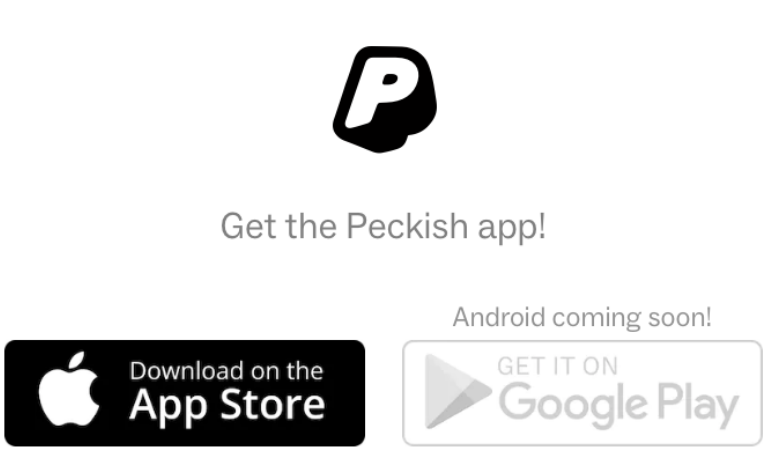
Reviews seem to be generally positive for this application, with an average of 4.1 stars on the Google Play Store. [43] Whilst views seem to be rather good for the mobile version of this application, users often complain about the 'tag' system; for example, one user searched for 'fish' dishes, but 'chicken' was returned instead. This is something that will be kept in mind during the development cycle of this project, as accurate search results are of the upmost importance. Especially when dealing with food, it is crucial to consider things like allergies. For example, if a user searches for recipes without nuts, the search must not under any circumstances return recipes containing nuts. Allergic reactions can be very serious, and are often lethal. Hence, tags and filters are something to be taken very seriously when developing this application. This is a significant disadvantage of using this website and

application. If someone wishes to search for a specific ingredient or filter by avoiding a specific allergen; only recipes meeting those filter requirements should be returned; not any others.

2.2.4 Peckish - Mob Kitchen

This page [80] is an existing solution to this project; ChefSwipe, as it is essentially a social platform for viewing and sharing recipes. Mob Kitchen, a media outlet created the application, 'Peckish', which was released on 1st Febuary 2023. This application can be used in a similar way to ChefSwipe, and is a self-proclaimed "online hangout for food lovers and cooking enthusiasts".

Though this application does not use similar technologies or design types as ChefSwipe, both applications have similar goals: to create somewhat of an online community based on users passion for food and sharing their much loved recipes and culinary skills. This application has significant up-sides compared to other applications in the space, such as daily competitions. This is also an advantage over ChefSwipe itself. [81] As 'Mob Kitchen' had a large online following prior to the launch of their 'Peckish' application, [82] the user-base for Peckish was almost certain to have a large boost at launch. This would be difficult to achieve for an application like 'ChefSwipe' which would have no social media presence at the time of it's launch. Because Peckish's userbase has been so large since its inception, the number of recipes available on the app would be significantly greater than that of ChefSwipe, as hundreds of users would rush to publish their recipes on Peckish, whereas ChefSwipe would require a much larger following to achieve that level of recipe submission.



Though not many disadvantages were found on this application, there certainly are downsides to an application so large, which is of course the running costs associated with it, along with the need for manual moderation at a much larger scale than ChefSwipe. Running large databases and housing the amount of user data required for an application like Peckish would be quite expensive, as will be discussed in chapter 4, section 5; development issues. Another downside to this application is the fact that it is currently not available on Android devices. [82] This cuts down on a significant potential market of users. As found in a Digital Information World study [83], in 2022, 71.7% of the mobile operating system market was dominated by Android whilst only 27.57% of the market were iOS devices. This increases the confidence that developing ChefSwipe for android devices was a good decision, whilst also informing us that developing for iOS may not have been as successful of a decision due to the fewer amount of devices sporting this operating system.

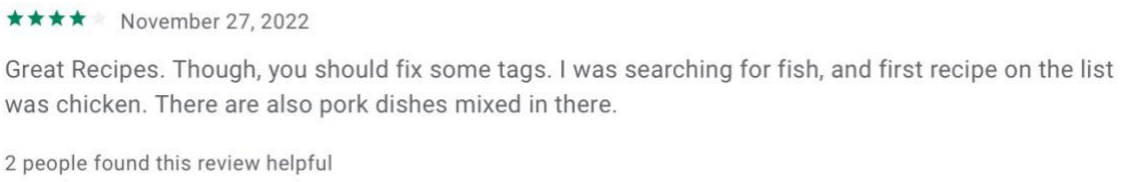
With this advantages and disadvantages taken into consideration, this application is still highly successful, which proves that there is certainly a market for a social network for foodies to share their dishes.

2.3. Technologies Researched

The research phase of this project is critical, as it allows me to see if the idea would be unique enough to succeed in the current market of recipe applications. From investigation into the field so far, the proposed interactive design, which allows users to search for recipes, is viable and distinctive enough to thrive. [28]

Java in Android Studio will be used due to the documentation available for this platform and language online. This would somewhat limit what platforms the application could be deployed to however . This would limit the userbase, with over 1 billion iPhones currently in use across the globe. Due to this, Kotlin could be used in place of Java, as it would allow for deployment of application more easily to other platforms, not just Android phones et cetera. As Java is an object oriented language, we are provided with many benefits such as flexibility, modularity and reusability. [44] Many of these features provide us with the massive benefit of more efficient and concise code. Needless to say, concise and efficient code helps down the line with future development and debugging.

Thanks to Android Studio's support for emulation, It would be a suitable fit to developing the project, especially in the early phases where constant testing is absolutely critical. [27] This allows for easy trials of the application, to ensure features work without having to constantly deploy the app to a mobile device for testing; eating up precious time. [6]



As previously mentioned, this particular project would involve development using Android Studio. However, before even beginning on development of the application, wireframes will be created to map out the layout of the application. These wireframes would certainly be subject to change, but are often very helpful during development, as they aid in staying focused on the necessary aspects of the app. [35]

The plan was to make use of cloud solutions such as Firebase to store and work with data, maintenance on the application would be rather straightforward. Recipes and user data can be edited without having to push updates to every user's mobile device. This cuts down on storage space and keeps users happy; no one likes an application that requires an update each and every time you open it, it's frustrating and may end up losing the app users in the long run. Google Firebase is a powerful tool for mobile app developers. It's free to use and easy to set up, which makes it a great option for anyone looking to create a mobile app.

Databases are an essential component of many applications because they allow for the storage and management of large amounts of data. This data may then be used by the program to provide a variety of features and capabilities. A database, for example, can be used to hold information on users, items, orders, and other data important to the application. This data may be accessible by the program as needed, allowing it to deliver tailored experiences and other features that rely on up-to-date information. Databases may also help to improve an application's performance and scalability by offering efficient ways to store

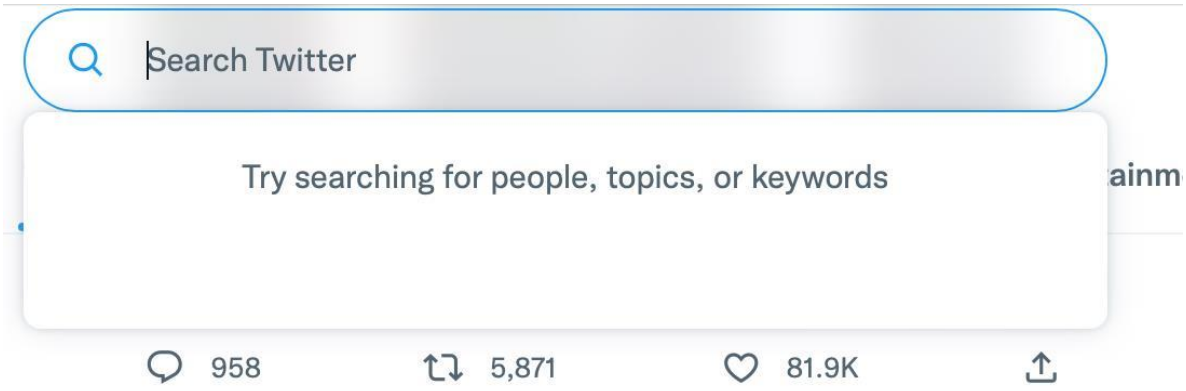
and retrieve data. Overall, databases play an important role in enabling apps to give rich and dynamic experiences to users.

Firebase has a whole host of different features but in this article I'm going to be focusing on the authentication service as well as the database service. The authentication service allows you to log in your users using either Google or Facebook accounts, while the database service allows you to store data.

In addition to using Firebase, there are several APIs online which would help with finding recipes online to present to the application's users. Android Studio documentation will be used [4] to aid in the creation of the project, due to the fact that there are a number of write-ups which would aid in implementation of features like user-to-user chatting or user-submitted content/recipes. [36]

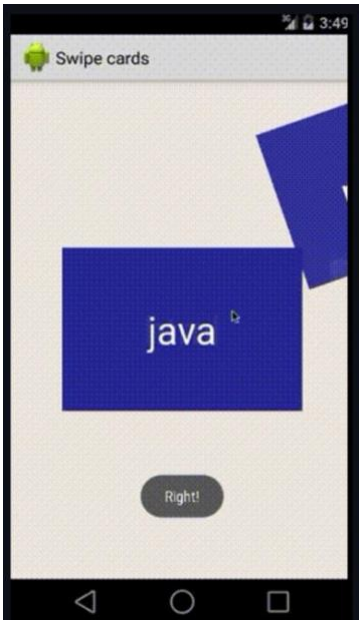
2.4. Other Relevant Research

Other research carried out includes delving into the features present in applications such as Instagram [9] and Facebook [31]. This research contributed in the development of this project by inspiring features such as the 'tags' and 'search' functions.



In addition to researching other social medial platforms, research was also carried out into the area API and API access. As previously stated, there are numerous APIs hosting recipes, but one in particular that would be beneficial to this project; mycookbook.io [23]. With the addition of an API, there would be a massive workload reduction in terms of adding recipes. Less of a focus would have to be put on manually entering recipe details, as a script would allow for this information to be automatically added to the Firebase storage. This would provide users with a much larger amount of recipes, as opposed to the limited supply when not using an API, and manually entering recipes. Access to an API such as the mentioned [23], would also enable the application to automatically keep up to date, which again is of massive benefit to both developers and users alike.

Research was also carried out into the eBook market, as this application would be in direct competition with 'eCookbooks', [20]. eCookbooks and paper books have advantages and disadvantages. eCookbooks are electronic copies of cookbooks that can be read on devices like smartphones, tablets, and e-readers. They offer several advantages over physical cookbooks, including being less expensive, more portable, and simpler to keep. They also allow you to read more books without having to go to a bookshop or library. Some people, however, prefer physical books since they give a more conventional reading experience and do not necessitate the use of technological gadgets. [20] Physical books have the added benefit of being able to be shared with others and resold or given after you are through with them.



As the eBook market is a growing market, offering an alternative solutions to using an eReader for the issue of finding new recipes will add a new form of competition to the market. This will allow those without the use of eReaders to avail of this new form of recipe book.

From researching infrastructure that could be used within this project, a library named 'Swipecards' was discovered. This library implements functionality to 'swipe' an item in a FrameLayout. According to the author of this library, this library "creates a similar effect to Tinder's swipable cards with Fling animation."

The functions provided by this library allow for code to be called when the item is swiped to either the left or right.

In addition, functions supplied handle events such as the FrameView being clicked on, and updating the 'cards' array adapter automatically by calling the 'updateArrayAdapter' function.

2.4.1 Legal Research; Image Copyright

As this project makes use of images to display when presenting a recipe to the user, image URLs and files must be saved to the project database. However, when dealing with images and photography, copyright must always be taken into account.

The legal protection afforded to the author of a picture to prevent others from using it without permission is known as image copyright. This means that if you develop a picture, you own the copyright to it and have the ability to regulate how it is used and distributed. If someone else wants to use your image, they must first get your permission, unless the use comes under one of the copyright exceptions, such as fair use. [61]

As a general guideline, images should only be used when permission is given by the copyright owner. This means that one should avoid using photos found online or in other sources unless it is confirmed that the use of the image is permitted under the terms of the copyright holder. For example, some copyright holders express that permission is granted to use in commercial projects. Images with this permission granted are often referred to as

'royalty free'. In some situations, it may be possible able to use an image under 'fair use', which allows for the restricted use of copyrighted material without the permission of the copyright owner for purposes such as criticism, commentary, news reporting, teaching, scholarship, or research. However, commercial projects are often not covered under the 'fair use' agreement. Hence, images will not be covered by fair use in this project, as it has the potential to generate revenue. [62]

If copyright laws are not followed, legal penalties may result. This could involve getting sued for infringement of intellectual property, which can result in significant damages and legal fees. [63] Aside from the potential financial costs, utilizing copyrighted content without permission can harm one's reputation and diminish trustworthiness. Hence, it is absolutely critical to respect copyright holders' rights and to use photos and other copyrighted material only in line with the law.

2.4.2 Research Paper

Food Allergies and Other Food Sensitivities

Steve L. Taylor, Ph.D. and Susan L. Hefle, Ph.D.
[Accessed 01/Dec/2022]
https://www.functionalmedicineuniversity.com/AllergiesandSensitivities.pdf

This paper outlines the importance of catering to food allergies, highlighting the various forms of allergies to certain foods, alongside the worries faced by those with food allergies. As the paper states, consumption of a food item one is allergic to can cause often serious reactions, such as anaphylaxis. This reaction can be very dangerous, and is sometimes lethal. Hence, the authors speak of the way of life that those with allergies must resort to; they must often pay full attention of ingredient lists, rigorously searching through the contents of an item to ensure that their allergen is not present.

As defined by Dr. Taylor, food allergies are "*individualistic adverse re-actions to foods*" This is because of the fact that not all people are affected by these reactions to the same extent. As these reactions are individualistic, the extent of the reaction can often not be predicted in a given person with an allergy. As a result, those with specific allergies are advised to avoid allergens wherever possible.

As stated in this research, not all reactions are immediate. Some may take 24 or more hours to present themselves. Currently, the reason for delayed reactions is somewhat of a mystery, according to Dr. Lemke and Taylor.

Histamine, a chemical found in cells, is present in many foods. Sensitivities to histamine often result in histamine poisoning, upon ingesting certain foods (particularly fermented cheeses, citrus fruit, legumes etc.) However, unlike specific allergies, histamine poisoning is not completely avoidable, as these poisonings can occur in anyone; not just those with existing allergies. Common causes for histamine poisoning are from consuming spoiled fish, such as tuna.

Food intolerances are reactions to foods which are not considered normal in the majority of the population. Lactose intolerance is a very common example of a food intolerance. These intolerances are often less severe than allergic reactions, and are considered defect present in the metabolism, as opposed to in the immune system. Symptoms of these intolerances can present as many things, but likely result in tummy pain, bloating or skin rashes.

Allergies to food play a highly important role in food production, whether that be in a factory, or in a restaurant. Due to possible severe reactions to certain foods in individuals with allergies, it is of the upmost importance that ingredients are listed on food products where possible.

In this application, it is the goal to highlight any major allergen presented in the ingredients of a recipe; in order to help nullify the risk of any adverse reaction to dishes listed on the application. Whilst allergen identification cannot be guaranteed, filters put in place to aid in identification could prevent possible adverse reactions.

References:

- The Importance of Metabolism for Immune Homeostasis in Allergic Diseases https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8355700/ [Cited: 01/Dec/2022] Provided by Frontiers Media SA
- Foods High in Histamine https://www.webmd.com/diet/foods-high-in-histamine [Cited 01/Dec/2022] Reviewed by Dan Brennan, MD
- Food Intolerances https://www2.hse.ie/conditions/food-intolerance/ [Cited 01/Dec/2022] Supplied by the NHS

2.5. Existing Final Year Projects

Project A)

Title: Lisgrey House Restaurant Web Application
Student: Matthew Magee

Description (brief): This project was developed as a web application to allow clients to order from a restaurant post Covid-19, as from the student's research, the restaurant's ordering system was somewhat stuck in the 20th century, with merely a phone-in ordering system.

What is complex in this project: The complexities of this project came from, according to the student, building a custom system for the particular restaurant. Instead of a generic system, the student built a fully customized platform for the use of the restaurant.

What technical architecture was used: The project was primarily coded using python, using GitHub for version control. This project is a web application, hence it is interactive and fully dynamic.

A key strength of this project is that fact that it was fully custom to cater to the needs of the restaurant. Instead of developing a system that was just generic to any restaurant, the student decided to focus on one specific area in order to perfect it. For the restaurant's needs, this is absolutely a strength, as it is a 'plug & play' system; no further customization is needed. However, this could also be seen as a weakness. It would be much more difficult for a different restaurant to pick up this system and adapt it to cater for their needs. For a more universal purpose, it may be more beneficial in the long-run to develop a system which supports customisable table layouts, and editable menus. This would allow for the web application to be utilised by other restaurants as opposed to being custom-built for the 'Lisgrey House Restaurant'.

In addition, mobile applications are very convenient in the restaurant industry these days, whilst it is not necessarily a weakness of the project, a mobile application port of the project would be very beneficial, due to the fact that it may be easier to use and more convenient to connect to.

Project B)

Title: Development of a iOS application to query and display nutritional data to a User
Student: Rachel Callan

Description (brief): This project was made in order to allow users to 'search, view and store nutritional data' provided by different types of food.

What is complex in this project: The use of databases to pull nutritional data for specific foods, along with finding a price to provide some sort of nutrition-to-price ratio was a complex aspect of this project, as it would require multiple APIs to complete.

This application was developed for iOS devices, using Objective-C and Xcode. This application used a GUI to communicate with the client application, a web server, and finally a database. MySQL was used to store user data on a database. It also possesses a barcode scanner, which users can make use of to find nutritional information easily, simply by scanning a barcode on food products. In addition to these aspects, prices will be provided for the scanned item, which can be expanded upon to provide pricing information for items from different stores.

A disadvantage of this project in its current state is that it was not exported to Android. Though iOS users account for over 1 billion mobile devices, without exporting the application to android, the application is not hitting a large number of smartphone users. This will overall decrease the potential userbase. As mentioned in chapter 2, section 2.4, not exporting an application to android misses out on their dominance of the mobile operating system market, dominating at 71.7% [83]

A strength of this project is the barcode implementation, and its overall uniqueness. Hence, we can confidently state that the author has identified a gap in the market. This may improve the userbase significantly, as whilst there are other applications in this area, it's not always possible to find an application which works for your chosen store or for all food and barcode types.

2.6. Conclusions

From this research, we can tell that there would be a market for this application based on the presented research into the growing eBook market. The application's viability and use-cases are also discussed, as it is clear from the study presented that it would provide a helpful and beneficial service to users. This is, without a doubt, critical, as providing such as service will ensure user retention. Based on existing project research, the area of food nutrition, meal planning, and so on, it is apparent that this topic is at the forefront.

The research into different techniques for development has also been outlined, and from this research, we can conclude that the best techniques for this project would be to utilize Java, alongside Android Studio. It is acknowledged that languages such as Kotlin could provide the benefit of being deployable to more platforms. However, from the research carried out as a part of this project, Java has a huge amount of community support when compared to almost any language [48]. This provides a large benefit to the development process within this project, as there is a large number of articles, projects and libraries online that provide support for Java.

3. Experiment Design

This chapter offers an explanation of the experiment design process that was used to create this project, outlining the many, constant design testing and evaluation which will be further developed upon in chapter 5. Also, a summary of the project's system design, including the diagrams created using the Unified Modelling Language will be provided. This chapter also describes some design enhancements added in response to user comments and feedback.

3.1 Introduction

This chapter provides an overview of the project's development process. The System Design chapter also includes project UML diagrams, alongside further information on the research and development of this program.

Good system design can provide a various advantages, including enhanced performance, scalability, and maintainability. [60] A well-designed system is usually easier to understand and use, making it more efficient and effective. This can lead to better performance since the application can run more efficiently. A well-designed system can also be more scalable, since it can easily accommodate larger volumes of data or users without significantly decreasing performance.

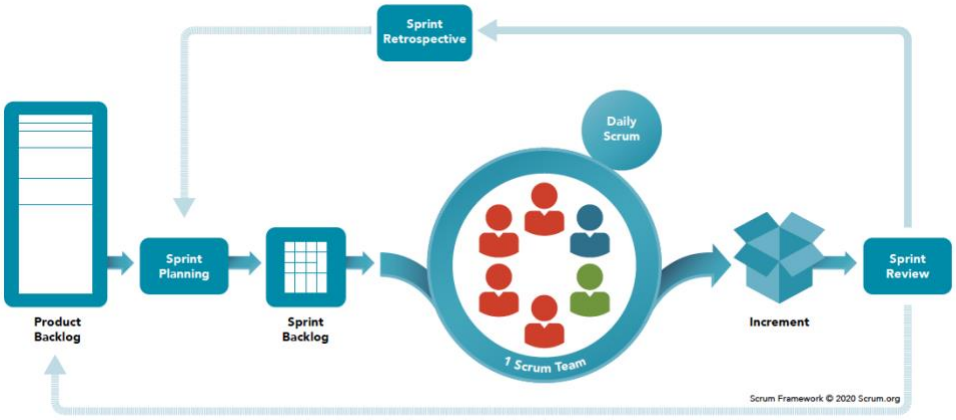
From IDE to prototype, this chapter will outline the development process so far, with insights into software methodology and design choices that have been made regarding application styling so far.

In this chapter, key system design principals will be discussed ranging from different types of software methodology, to the importance of good maintainability within your code.

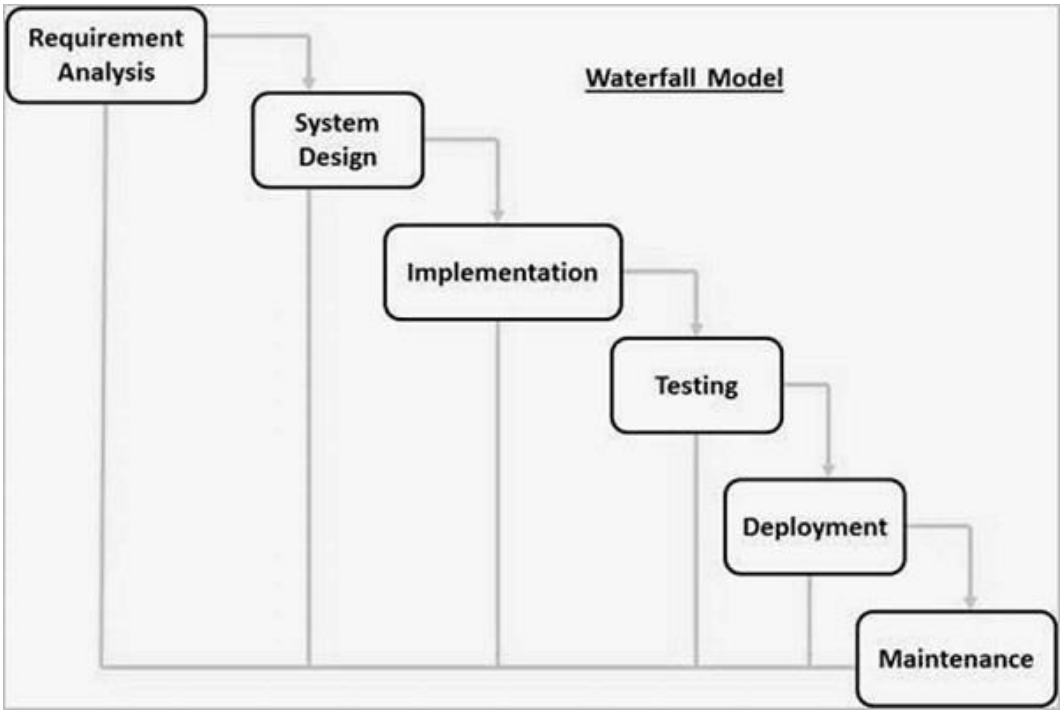
3.2. Software Methodology

There are many methodologies which may be used in development of a project. Like with any significant choice, each comes with its own set of advantages and disadvantages. In software development, these 'pros' and 'cons' so-to-speak must be weighed against the specific requirements of the application in order to determine which will provide the biggest benefit to us as developers.

One very popular mythology, named 'Scrum' is often seen made use of in the world of software development. 'Scrum' is based on the idea of having multiple development cycles. These cycles are organised into sprints; 2-4 week blocks of time which focus on outputting a specific group of features or functionality. Overall progress on the application is evaluated by the efficiency of each sprint, or in other words; how often a Sprint is completed to a high level by the pre-determined timeframe. The benefits of scrum are evident, as it provides a high level of adaptation to project requirements which may change over time. However, it was not particularly suitable for this project as available time to work on the development aspect of this project may fluctuate from month-to-month, depending on uncontrollable factors such as external submission deadlines and workloads. [67] An example of this is displayed below, courtesy of scrum.org.



Yet another methodology which was considered for this project was the 'Waterfall' methodology. This system provides a focus on completing specific tasks set out at the start of the development process, which must be completed before moving onto the next step of the 'waterfall'. This system is beneficial as it can help developers stay on one task before starting on the next; eliminating the risk of them 'jumping' from task to task, and never seeing one feature through to completion. However, as this applications requirements are likely to change over time, this waterfall methodology would not be suitable. The changing of final product requirements may cause the waterfall to be restarted, which can cause unnecessary delays. [68] An example of the waterfall model is displayed below, courtesy of tutorialspoint.com.



For this project, the software methodology used most regularly is Agile project management. This method is designed to be more flexible and dynamic than the older waterfall method [54]. The four main values of Agile development, according to the Agile manifesto [55] are as follows:

1. 1) Individuals and interactions over processes and tools
2. 2) Working software over comprehensive documentation
3. 3) Customer collaboration over contract negotiation
4. 4) Responding to change over following a plan

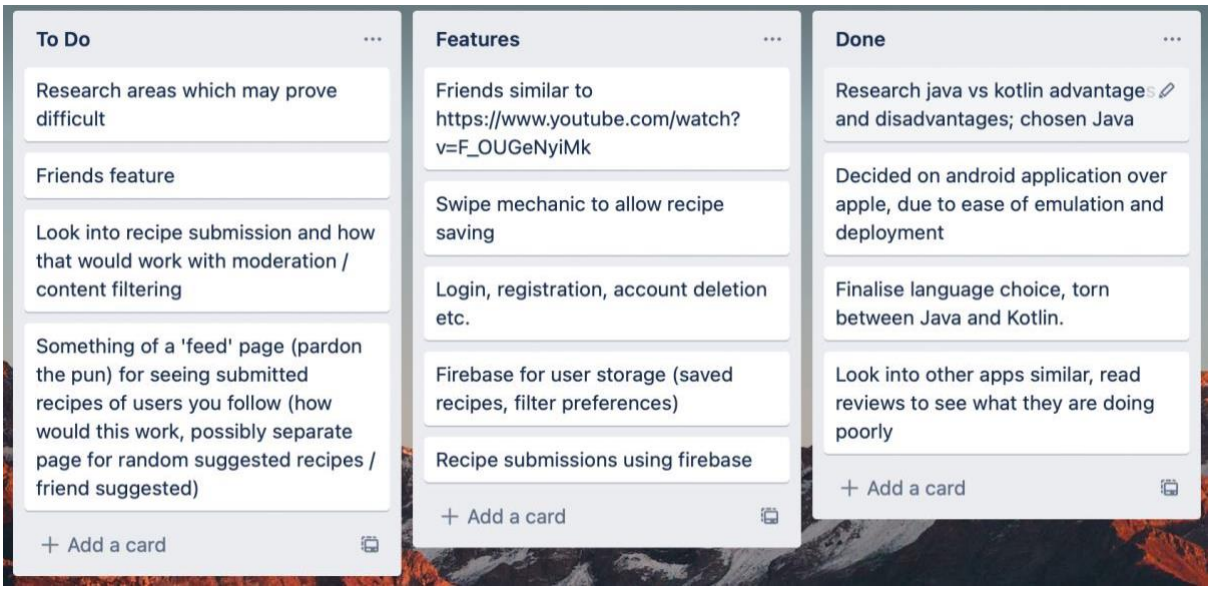
Agile Kanban management is used in this project's case, as it is has high flexibility, and categorises features into set phases; 'to-do', 'work in progress', and 'done'.

This system for categorising feature progress is highly beneficial in this project's case as it provides constant reminders of the project scope and context. This is of great help, as it avoids distractions, and reduces the risks of getting side-tracked during the development process.

Also, as this application is built for clients, or 'users', it is important to take up their perspective when making design choices. Hence, the Agile development methodology has been chosen as this project's software methodology to see through the eyes of a client, and to help understand their needs and wants. Unlike in scrum, where the project manager is viewed as the end-user, this approach, as mentioned, allows collaboration with clients.

Following on from the values of Agile development, when working without a development team, responding to change is of the upmost importance. For example, as tasks cannot be delegated, it is difficult to come up with a time-frame from for the development of a given feature. Hence, it is important to respond and adapt to changes in a development plan as they occur, instead of sticking to a rigid plan, and faltering if said plan cannot be followed.

Furthermore, a ‘Trello’ board was set up to aid with staying on track with development, with tasks that must be completed kept in the ‘to do’ section, which get moved over to ‘done’ once they are complete.



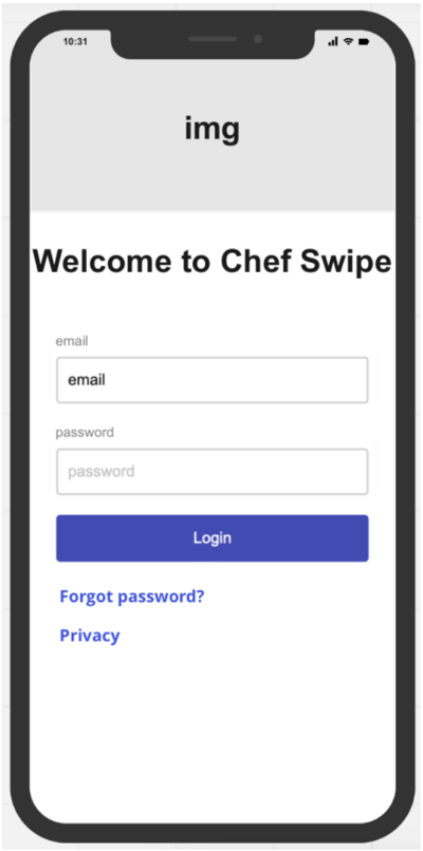
Trello boards have been shown to aid in the completion of tasks by keeping users on track of what needs to be done. [51] As a result, this will undoubtedly help the project's development process by keeping track of work that needs to be completed and work that has already been completed.

3.3. Overview of System

Currently, the application is running locally on an emulated Android Device. The device being emulated is a Pixel 3A, via Android Studio’s built-in emulation system. [27]

As outlined previously, the application is being developed using the object-oriented programming language, Java. The modularity afforded by Java enables for quicker troubleshooting, which will substantially benefit in the testing process. In addition to modularity, we can reuse code using inheritance. This will be especially beneficial in the software design portion of this project, as objects will frequently need to be called from various classes at different times. Code does not have to be typed several times when using inheritance (and functions). This improves efficiency and overall conciseness of code.

3.3.1 Login & registration page



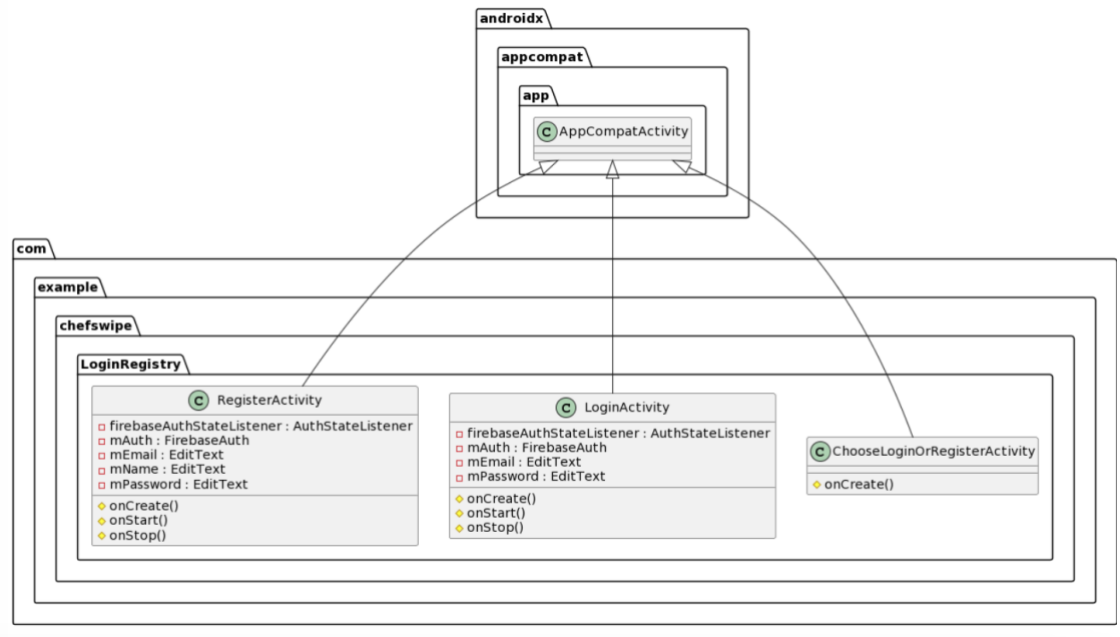
Upon launching the application, users will be asked to log into an account on the 'login' page. A sample wireframe of this page can be seen in the image to the left.

In the case of new users, an option will be presented by which the user can register. Registration requires a valid email address, a password and username. This information is passed through Firebase, where the email address is authenticated.

If the registration process is successful, the Registration page utilizes Firebase's ID system to generate a random user ID, and assign this string to the account that has been created.

The account details are then sent to the online real time database using a DatabaseReference, as is seen in the below screenshot of the 'Register' class within the project. This screenshot showcases the system present to store user details under the heading of the user ID.

This process can be seen in detail in the below UML diagram of the 'Login and 'Register' activities. This diagram gives an overview of the functions in place within the application. As previously stated, upon booting up the application, the user is immediately brought to the Login activity, and is given the option to load the registration activity. These activities both extend the AndroidX AppCompat activity .



For users that have already been authenticated by firebase, and are 'logged in' to the application; there is no need to request them to login to the application again. Whilst this extra step of authentication can be very helpful in locking down user accounts, it can also be very frustrating for users to deal with. [46]

To avoid having the user input their login details each time they boot up the application, we use an 'if statement' within the Login page to check if the user is authenticated.

Upon booting up the application; if the user is already logged in, we can skip the login page entirely and boot up the main activity, or 'feed' page of the application. However if the user is not detected to be 'logged in', we continue with the process of authentication.

3.3.2 Home page

Once the user has been logged into their account, the 'feed' page is loaded. This page is the home page of the application, where the user will utilize the 'swipe' functionality to like recipes. A wireframe of this page is featured below, which showcases the planned design for this page.

As we can see from this wireframe, the top of the page features a search bar. This search feature will be used to search for recipes, or search for user accounts. From here, the opportunity will be presented to follow certain tags or creators. This feature will have similar functionality to Twitter's 'search' function [47].

Continuing on from this, we see the 'Recipe Name', which would retrieve the name data from our Firebase database and present itself to the user. This will be accompanied by the matching image, and recipe tags.

These tags will inform the user of the main components/allergens of the dish, such as 'Nuts', and will also let the user know if the recipe is Vegan, Vegetarian, et cetera.

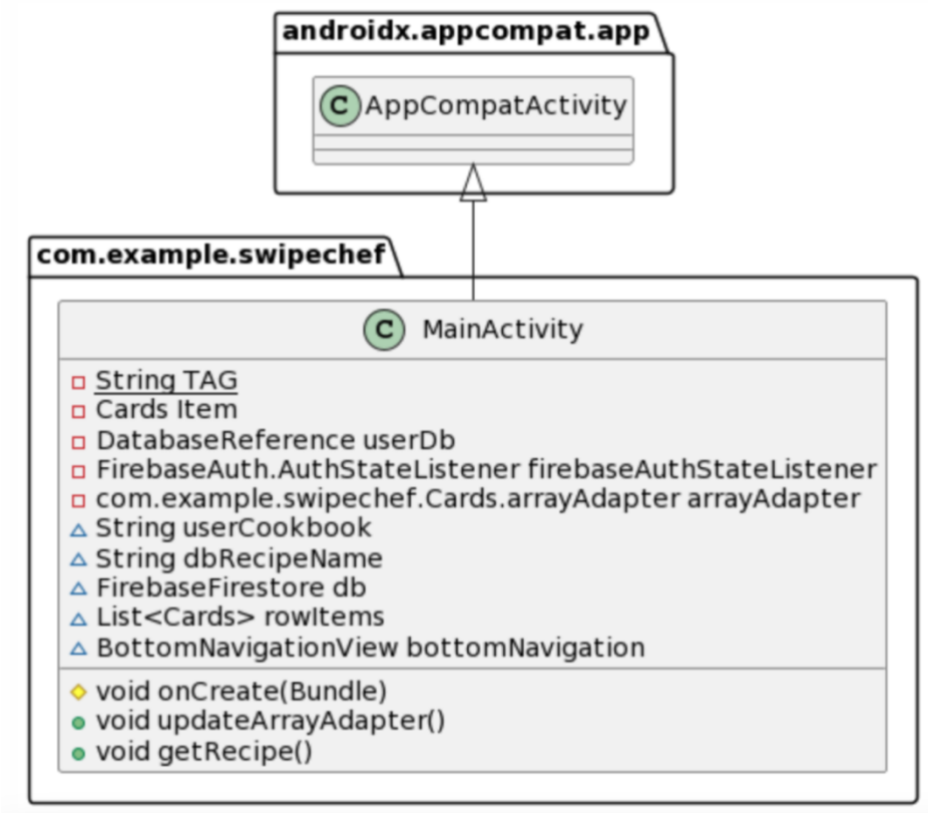
A description of the recipe, along with an author name will also be provided on this page. The description will give a run-down of the recipe, regarding what it is, while the 'author' field will be the username of the recipe creator.

As likes are being tracked in order to 'save' recipes, these likes can also be displayed on this page, to let application users know how many other people 'liked' this specific recipe.

As will be showcased within this document, this 'home page' will go through many different design iterations, with constant changes to the UI being applied through the application's

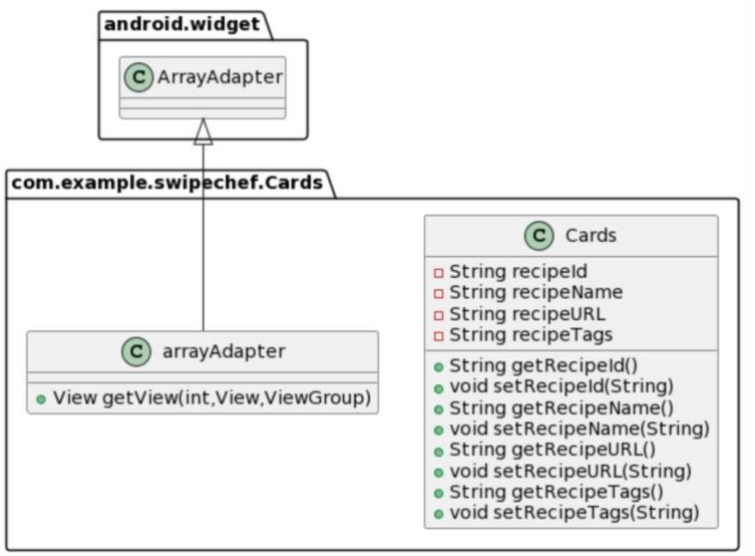
lifespan so far. These changes have been made based on user feedback as outlined further in this chapter.

The above image is a UML diagram for this Main activity is included below.



As seen, this activity possesses some basic functions such as 'getRecipe', which retrieves data from the Firebase database. It also contains links to another class, 'Cards', which is used to store the data from this 'getRecipe' function. This 'cards' class has getters and setters, which retrieves the data from our 'getRecipe' function and allows us to add the data to a list named rowItems. This list is then used as our 'stack' of recipes.

The UML diagram for the 'cards' class can be seen below.



At the bottom of this page is a navigation bar. This bar will be used to switch through pages on the application; for example, bringing the user from the home page over to their profile page.

3.3.3 Profile page

This profile page will be available to the user for purposes of uploading a profile picture, or adding a bio to their account; making them more recognisable to their friends and followers. This page will also contain a list of the user's published recipes. Keeping this list on the user's profile will allow other users to quickly view what recipes have been submitted by this user in the past, giving them insight into the nature of the user's recipes.

Also present on the navigation bar will be a 'saved recipes' page, where all of the user's saved recipes will be saved. This page will include a small image of the recipe, along with the recipe title, author and tags. The recipes will be saved in a list which the user can swipe through to find what they wish. This page will also give the user an opportunity to 'remove' a saved recipe, 'unliking' it and deleting it from this list. A sample wireframe of this page can be seen below, giving insight into the concept behind this feature.

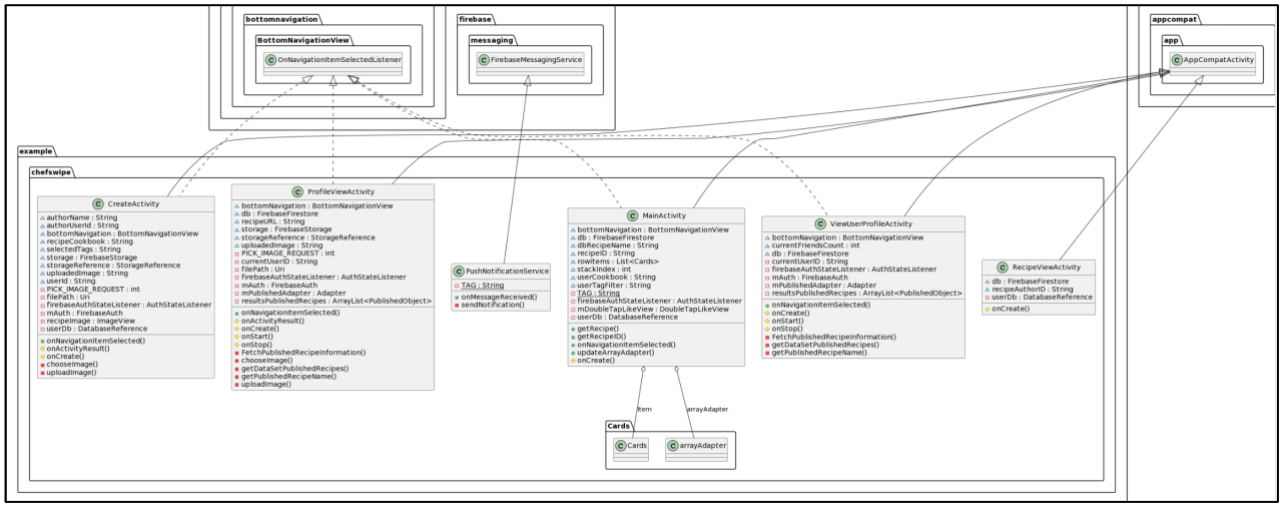
A major risk with regards the design of the project system relates to the concept of 'user-submitted' recipes. As this platform is designed to be somewhat of a social network, it would be preferable if users could submit their own content for other users to view. However, this could lead to inappropriate content being submitted by users. Needless to say, such content would not be accepted on the proposed platform. This is a large risk in terms of local laws for this project. Also, the project must strictly adhere to Firebase's terms of service (TOS) policy. [50] To mitigate this risk, it is suggested that the content be filtered before being uploaded to the Firebase database. This reduces the possibility of inappropriate content being published.

By performing manual checks on the content being submitted, it is possible to increase the security of these measures and thus reduce the risk. It is proposed that before content is published for public viewing, an application moderator / administrator be given the option to approve or deny said content. This would almost eliminate the possibility of any objectionable content being published to the application.

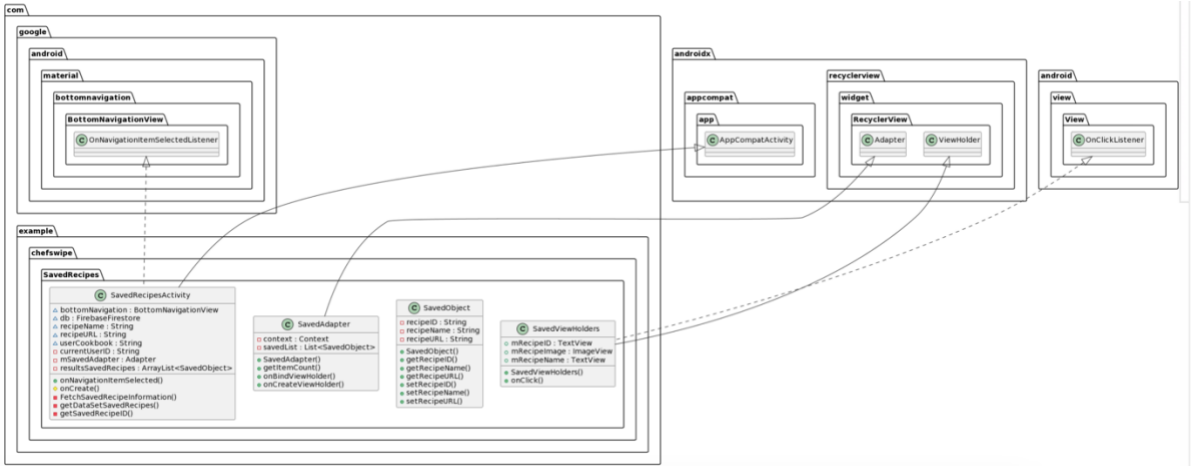
This design was implemented in the form of a simple Boolean value, which checks if the user-submitted recipe has been approved by an administrator. When a recipe is created, this value is set to false and stored alongside other data on the recipe. However, once an application administrator has checked that the content in that recipe is appropriate, they may set this value to 'true', and the recipe will be considered approved by the application. This is a simple solution, but is certainly effective, as it ensures that only recipes which have been personally approved by administrators will be displayed to other users.

A screenshot of this is provided below, where an example of a pre-approved recipe is given. As the ‘approved’ Boolean has been set to ‘true’, we can assume that this recipe has been checked over, and is safe to display on the application home page.

3.3.4 Class diagram



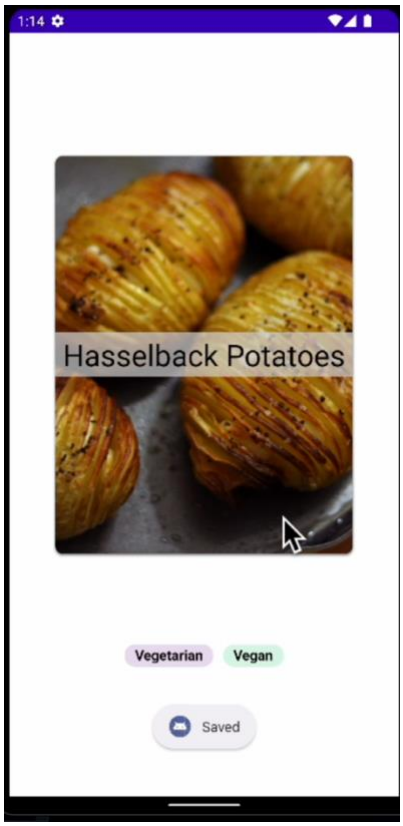
3.3.5 Saved recipes diagram



3.4. Design evaluation

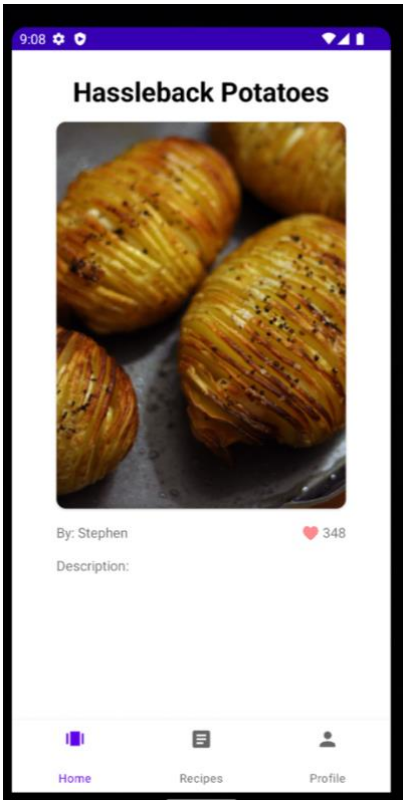
Well designed and intuitive UI was a large concern during the development process of this application. As this application is somewhat of a 'social media' platform, it would be in competition with other food-based platforms with large teams of developers working on building the perfect user experience. Things like user engagement, user retention and brand image are all directly affected by the design choices made within an application. Hence, it is of the upmost importance to get these things right during the development phase, before the application is published and live. [66] This task can be difficult to carry out when working on a small team, so user testing was employed in order to iron out issues with the applications UI. The screenshots below outline the various changes that the home page went through during this design evaluation phase. Live feedback/development sessions were carried out, whereby changes would be made to the UI, and testers would rate the changes. The design would then change again, and testers were asked to compare the changes, weighing up advantages and disadvantages of each design choice.

3.4.1 Homepage testing & evaluation



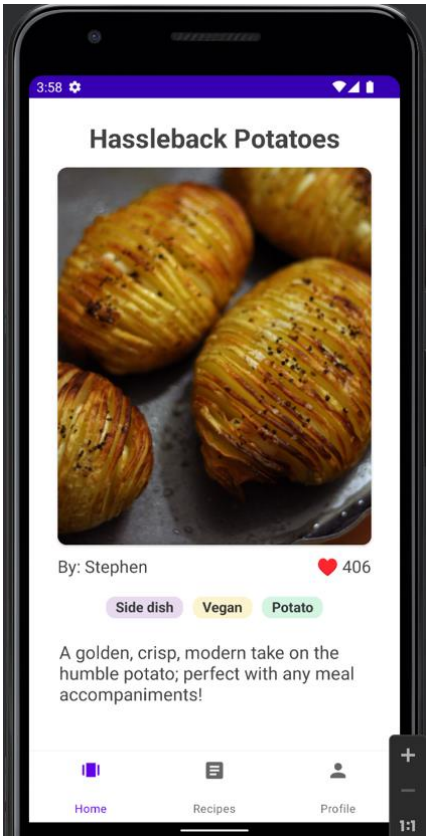
This task can be difficult to carry out when working on a small team, so user testing was employed in order to iron out issues with the applications UI. The screenshots outline the various changes that the home page went through during this design evaluation phase. Live feedback/development sessions were carried out, whereby changes would be made to the UI, and testers would rate the changes. The design would then change again, and testers were asked to compare the changes, weighing up advantages and disadvantages of each design choice.

This screenshot is the first iteration of the home page. This displays the essential information about the recipe, such as the name, image and tags; but nothing else. Users found this to be too 'plain' and 'simple', which is evident from the lack of information presented to the user.



The next screenshot showcases significant changes to the home page, with the addition of placeholder text for things such as a 'like counter', and descriptions. The 'like counter' was an unplanned addition to the user interface, however users felt as though it made you feel more "connected" with other users, and that your actions would actually "impact on other peoples' decisions on recipes". The recipe name was also moved above the image, as one tester in particular felt as though too much of the image was being covered by the recipe name, and it looked "messy" as a result.

During this phase, a navigation bar was also added to the application, to allow users to switch between the different pages available to them.



The next batch of changes to the home page was not as significant from a design point-of-view, however the linking of the placeholder text to the database information allowed for further text customisation, with styling choices being made with regards font size and colours; with the hopes of increasing readability.

These changes, visible in the screenshot to the left made the app feel much more professional, according to a number of testers. From here, the design did not change much further with the exception of some features mentioned below, such as the title bar, as seen below

In addition to these minor changes; the 'tags' feature was added to the home page, which allows users to see a recipes key tags, such as dietary requirements or allergens. This aids by letting users glance at a recipe, and quickly check if it would be suitable to make, if for example; they are vegan, allergic to egg, and so on.
This feature was very successful with testers, and positively added to the style of the application to a great extent.

Now that the testers had agreed upon a design which they enjoyed utilising, work began on the addition of and styling of the home pages title bar. This title bar would house filters; buttons which can be used to narrow down recipe searches, and look for recipes with specific dietary properties, like 'vegan' et cetera. This also gave the users the opportunity to search for recipes of a certain type, such as deserts or side dishes.

3.5. Conclusions

Software design, along with the correct software design methodology are an essential part of any project. It is often seen as a roadmap, which outlines the key points within the development cycle of a project.

Software design can be related to the 'foundation' of a project's code [58]. Once we have the 'foundation' of software design in place, other aspects are considered, such as modularity. Modularity is a simple but important concept, whereby code is broken down into smaller sections and reused as necessary, as opposed to one large incomprehensible block. Some other important aspects within software design include:

1) Maintainability
2) Flow and functionality
3) Portability

This project has already outlined the portability aspect of software design, in the decision to build for the Android platform. Whilst this decision limits the possibilities for deployment, removing the opportunity for the application to be ported to the Apple App Store, android devices account for over 1 billion devices in the world today (over 70% of the market) [59] from this, we can see that portability has been taken into consideration, however it was decided to build the application for 70% of the market as opposed to building for the minority of the market; the >30% of active iOS devices.

Good software design practices are of great significance when building a large-scale application or project, as they ensure that code is easier to write and understand thanks to good maintainability; which in turn aids in reducing the number of bugs introduced. Well-documented and maintained code is generally less error-prone, [60] which is of interest to developers in the long-run.

4. Experiment Development

This chapter, concentrating on experiment creation, is an important phase, offering an overview of the application's development cycle and how the aforementioned program design components are implemented. This chapter includes everything involved in the development cycle of this project, from software-development approaches to explanations of development choices.

4.1. Introduction

This chapter provides an overview of the project's development process. Outlining project UML diagrams, alongside further information on the research and development of this program.

Good system design can provide a various advantages, including enhanced performance, scalability, and maintainability. [60] A well-designed system is usually easier to understand and use, making it more efficient and effective. This can lead to better performance since the application can run more efficiently. A well-designed system can also be more scalable, since it can easily accommodate larger volumes of data or users without significantly decreasing performance.

From IDE to final product, this chapter will outline the development process of the application, with insights into software methodology and design choices that have been made regarding application styling.

In this chapter, key system design principals will be discussed ranging from different types of software methodology, to the importance of good maintainability within your code.

An organized strategy is required for effective software development. This is often achieved by the employment of numerous software development approaches; which provide somewhat of a template for the development process.

These methodologies are of great assistance to developers, as they require the initial gathering of user requirements; helping them stay on track. They also aid by guiding development through to final user tests and deployment. Hence, it is of the upmost importance that an efficient software development approach is considered before the project enters its initial stages of creation.

UML diagrams are used throughout the experiment development process in order to clearly convey the current behaviour of a system in the application. These diagrams, present in chapter 3, are a great tool when it comes to developing the system further, as it allows developers to see connections between classes and functions. They aid us as developers by confirming that the system is well laid-out, and that all criteria set out in the initial software development stages are satisfied.

4.2. Software Development

Development of this project began with the Firebase login system, which would be used to securely authenticate users, and save their corresponding data. [3] Firebase is a useful tool, used consistently throughout this project in order to house all forms of data, from user IDs to recipe information.

4.2.1 Login and Registration Development

The login page of the application gives the user the option to either log into the app, or register as a new user. Choosing the 'register' prompt will allow the application to write new data to the database, recording the users information as a new entry in our realtime database.

This login system features a number of quirks, one of which allows users to login automatically once the application is opened. This additional system, though not necessary, reduces the frustration related to requiring users to enter their personal details each and every time they open up the application. To implement this system, an 'if' statement was added in order to check if the current user has been previously authenticated by the application.

```
mAuth = FirebaseAuth.getInstance();
firebaseAuthStateListener = firebaseAuth -> {
    final FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
    //Check if user is logged in
    if (user != null) {
        Intent intent = new Intent( packageContext: LoginActivity.this, MainActivity.class);
        startActivity(intent);
        overridePendingTransition(R.anim.fade_in, R.anim.fade_out);
        finish();
    }
};
```

The above screenshot is a snippet of code that performs this process. Firstly, we create a listener, which essentially 'listens' to the current authentication status of the Firebase session. This status is assigned to the 'user' variable. If Firebase does not detect an authenticated, logged in user, this variable defaults to null. Once this variable has been set, we can check if the user is 'null', or 'not logged in' using an 'if' statement.

If the user is logged in, we can skip the login page entirely and boot up the main activity, or 'feed' page of the application. However if the user is not detected to be 'logged in', we continue with the process of authentication.

Whilst this may not be the most secure way of validating users, for this specific application, it was considered that no particularly sensitive information will be stored. Hence, the ease-of-use of the app was considered more important to superior levels of account security.

4.2.2 Homepage Development

Following development of the authentication system, came the process of implementing possibly the most fundamental aspects of the application; the home page. This page is where the user will interact with recipes by 'swiping' on them. As this page will be a major aspect of the application, it was important that it was developed to a very high standard, with efficient methodologies and attractive design. The basis of this page was developed using the 'Swipecards' library. This library allowed for items to be added to a list, which is then displayed on an interative Framelayout. This library is explained further in chapter 2.4.

The benefit of using this library is the easy implementation of some included functions, which allow for editing the stack of recipes, along with detecting actions upon the stack, such as clicks or swipes to the left or right of our screen. [49]

An example of this functionality is included in the below screenshot, which outlines the 'swiping' functionality. These functions detect if the card has been swiped to the left, or 'discarded', and if the card has been swiped to the right, or 'saved' respectively.

```
//Called when card is swiped to left
@Override
public void onLeftCardExit(Object dataObject) {
    Cards obj = (Cards) dataObject;
    dbRecipeName = obj.getRecipeName();
    userDb.child("Saved Recipes").child(dbRecipeName).removeValue();
    updateArrayAdapter();
}

//Called when card is swiped to right
@Override
public void onRightCardExit(Object dataObject) {
    Cards obj = (Cards) dataObject;
    dbRecipeName = obj.getRecipeName();

    Integer recipeLikes = obj.getRecipeLikes();
    recipeLikes += 1;

    DocumentReference docRef = db.collection( collectionPath: "Cookbook").document(obj.getRecipeId());
    docRef.update( field: "Likes", recipeLikes);
    userDb.child("Saved Recipes").child(obj.getRecipeId()).setValue(true);
    updateArrayAdapter();
}
```

As seen in the 'onLeftCardExit' function, the program simply removes the current object from the user's saved recipes, and updates the stack of recipes; ensuring that the recipe has been removed and is no longer visible on the application.

However, in the 'onRightCardExit' function, we can see the 'save' functionality of the application. Once the user has 'saved' a recipe, we update that specific recipe's like counter, adding 1 to its current total. This simple but effective design choice allows for printing the current total likes of that recipe, so that other users can see how many times it has been saved! This system was recommended by a tester, who suggested that it would be nice to know how popular a recipe is with others at a glance.

Once the 'like counter' has been updated and saved to the database, we add the ID of the liked recipe to the current user's personal saved recipes list, and update the array adapter to showcase the next recipe.

The getRecipe function, a portion of which is seen below, was added to retrieve document information. Recipes are stored as its own document, which is cycled through using a for loop, and added to our Swipecards recipe stack. The below function is utilised to get information on the requested recipe, such as its name, or image URL. This information is added to the rowItems array, which is utilised in a separate class to compile these strings onto 1 swipeable card, which is then displayed on the home page.

```
public void getRecipe() {

    Query colRef = db.collection( collectionPath: "Cookbook");
    colRef.get().addOnCompleteListener(task -> {
        if (task.isSuccessful()) {
            for (QueryDocumentSnapshot document : task.getResult()) {
                FirebaseDatabase userDatabase = FirebaseDatabase.getInstance();
                FirebaseAuth mAuth = FirebaseAuth.getInstance();
                String userId = Objects.requireNonNull(mAuth.getCurrentUser()).getUid();
                userDb = FirebaseDatabase.getInstance().getReference().child("Users").child(userId);
                DatabaseReference dbRef = userDatabase.getReference().child("Users").child(userId);
                dbRef.addValueEventListener(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        if(Objects.equals(userCookbook,  b: "All Cookbooks")) {
                            if(document.getString( field: "AuthorID") == null) {
                                DocumentReference authorID = db.collection( collectionPath: "Cookbook").document(document.getId());
                                authorID.update( field: "AuthorID",  value: "");
                            }
                            if(document.getString( field: "Author") == null) {
                                DocumentReference author = db.collection( collectionPath: "Cookbook").document(document.getId());
                                author.update( field: "Author",  value: "ChefSwipe team");
                            }
                            if(document.getBoolean( field: "Approved") == null) {
                                DocumentReference author = db.collection( collectionPath: "Cookbook").document(document.getId());
                                author.update( field: "Approved",  value: true);
                            }
                            if(document.getLong( field: "Likes") == null) {
                                DocumentReference author = db.collection( collectionPath: "Cookbook").document(document.getId());
                                author.update( field: "Likes",  value: 0);
                            }
```

In the case of this application, if we load the data for one recipe at a time, the animation of swiping one recipe to reveal the recipe underneath will not be seamless; it will have a 'jitter' effect, where the 'new recipe' data is only loaded once the 'old recipe' leaves the screen. From a design perspective, this is not attractive.

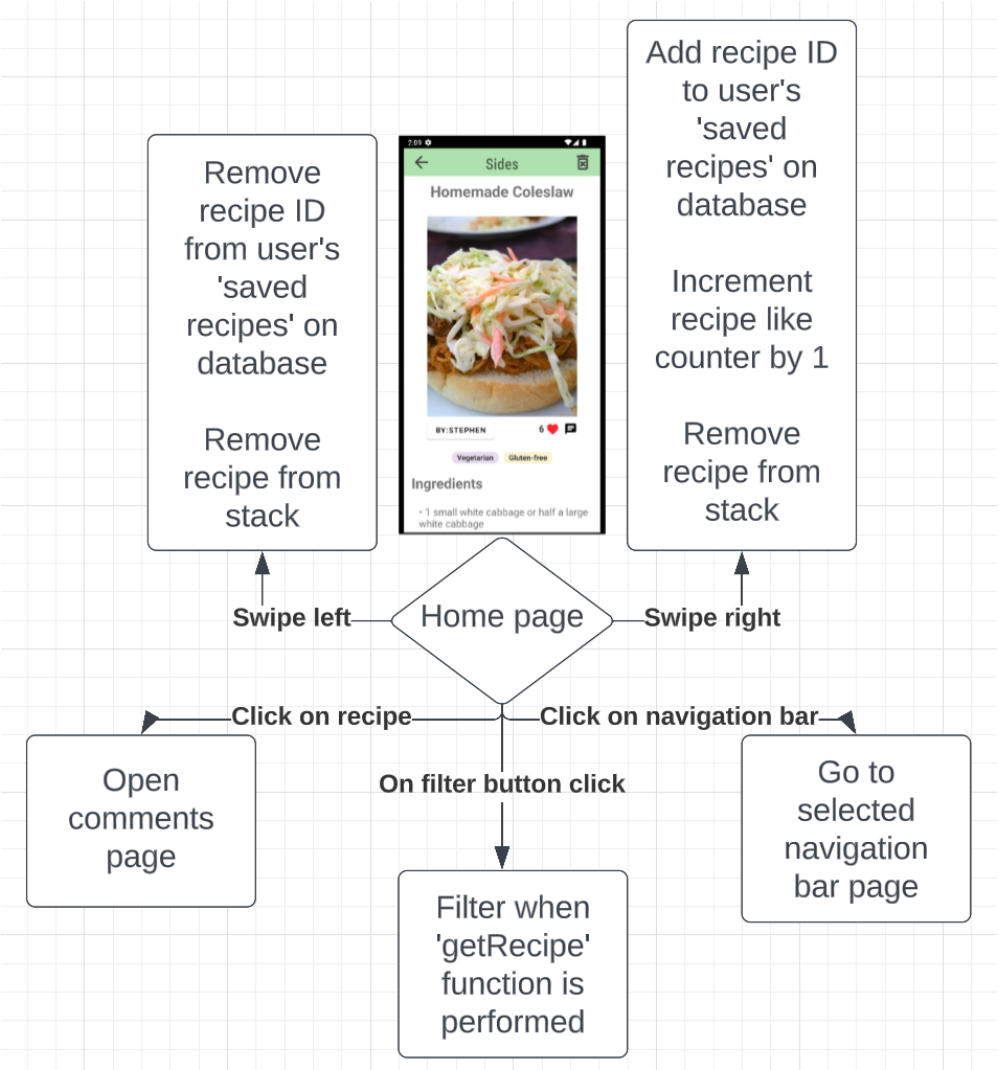
To solve this, a system was built in the form of this Cards 'rowItems' list, which allows the application to load multiple recipes at a time and add them to the list. This creates a seamless animation, and allows the application to simply remove the top item from the list when the user wishes to move onto a new recipe. Seen below is the section of the getRecipe function which adds the requested data to the rowItems list. This process eliminates the aforementioned animation jitter, whilst allowing for recipes to retrieved as per the 'filters' specified by the user.

```
    if (!dataSnapshot.child("Saved Recipes").hasChild(document.getId()) && Boolean.TRUE.equals(document.getBoolean( field: "Approved"))  && Objects.requireNonNull
        Item = new Cards(document.getId(), document.getString( field: "Name"), document.getString( field: "URL"), document.getString( field: "Tags"), document.getStrin
        rowItems.add(Item);
        updateArrayAdapter();
    }
}
else {
    if (!dataSnapshot.child("Saved Recipes").hasChild(document.getId()) && Boolean.TRUE.equals(document.getBoolean( field: "Approved")) && (Objects.equals(documen
        Item = new Cards(document.getId(), document.getString( field: "Name"), document.getString( field: "URL"), document.getString( field: "Tags"), document.getStrin
        rowItems.add(Item);
        updateArrayAdapter();
    }
}
```

Recipe filtering was added in order to allow for more customization of the feed page, by the user. The primary issue found with other recipe applications was the lack of customisable filters. For example, with the current filtering system, the user can select specific dish types, such as side dishes, along with dietary requirement filters. This enables them to find 'Vegan Deserts', or 'Gluten free mains', quickly and easily.

```
ImageButton filterButton = (ImageButton) findViewById(R.id.filterButton);
filterButton.setOnClickListener(view -> {
    AlertDialog.Builder tagBuilder = new AlertDialog.Builder( context: MainActivity.this);
    TextView tagTitle = new TextView( context: MainActivity.this);
    tagTitle.setText("Select a dietary tag filter");
    tagTitle.setPadding( left: 20,  top: 30,  right: 20,  bottom: 30);
    tagTitle.setTextSize(20F);
    tagTitle.setBackgroundColor(getResources().getColor(R.color.green1));
    tagTitle.setTextColor(Color.BLACK);
    ObjectAnimator animator = ObjectAnimator.ofInt(tagTitle,  propertyName: "textColor", Color.RED, Color.WHITE)
```

Originally, filtering was going to be implemented by means of a search bar on the title bar of the main page. However, the final filter selection buttons were implemented using spinners and dialog boxes, which were decided upon after multiple rounds of testing, as outlined in chapter 3.4. Following the opinions of the testers, it was decided that a search bar would lead to broad search terms, which may be hard to filter. Also, users may not be aware of what to search for. Hence, a drop-down spinner with pre-set selectable filters, and an alert box was utilised in the final product.

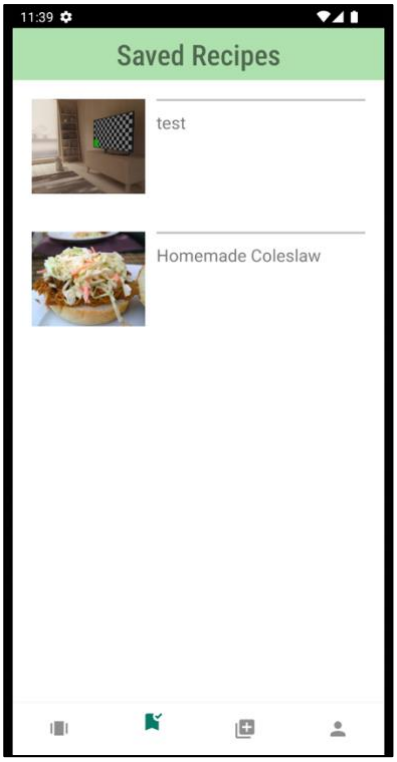


The above diagram outlines the home page functionality, depicting what will happen when certain actions occur, such as left or right swipes, along with clicks or filter button selection.

4.2.3 Saved Recipe Page Development

Once a recipe has been saved by a user, the application saves that specific recipe ID under the current user ID, essentially telling the program that the user has liked the recipe; and to make it available to them on the 'saved recipes' page.

This page can be accessed from the navigation bar, and makes use of multiple classes in order to display the saved recipe information on the page.

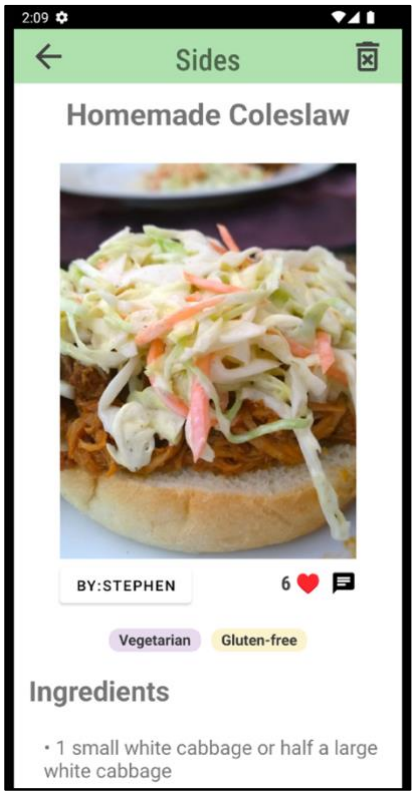


Once the 'Saved Recipes Activity' is launched; the 'SavedAdapter' retrieves saved recipe information utilising 'getter' functions found in the 'SavedObject' class. This information works in tandem with the correlating XML files, which by use of a ScrollView and RecyclerView, displays the saved recipes names and images.

As seen in the screenshot to the left, recipes which have been saved by the user are displayed in a neat list, which can be traversed by the user.

During the testing process, one tester suggested the ability to filter recipes within this saved list; in order to find specific saved dish categories, such as mains, sides etc. However, due to limitations with the current class system along with time constraints, this was not implemented. Should a user decide to view the ingredients, method and other details of a displayed recipe; the 'bundle' package passes the selected recipe ID to a separate activity, named 'RecipeViewActivity'.

```
@Override
public void onClick(View view) {
    Intent intent = new Intent(view.getContext(), RecipeViewActivity.class);
    Bundle bundle = new Bundle();
    bundle.putString("recipeID", mRecipeID.getText().toString());
    intent.putExtras(bundle);
    view.getContext().startActivity(intent);
}
```



Utilising this recipe ID, this new activity is able to request the other details associated with that recipe ID, which is then displayed in a neat and readable format using XML.

As seen in the related screenshot, the recipe is displayed in an easy-to-read manner, with all previous details about the recipe; alongside other additional touches such as a 'comments' button, or a link to the recipe author's profile.

The title bar of this page houses options to return to the previous page, or delete the saved recipe. Originally, the button for recipe deletion was contained on the Scrollable saved recipes page, however due to issues with some users accidentally mis clicking the deletion button; it was moved to this page as suggested by testers.

4.2.4 Comments Development

The comments page was not intended to be included in the app until a later date, but an overwhelming response from testers suggesting this feature led to its development and inclusion in the current version of the app. This feature is accessible by clicking on the recipe on the main page or by tapping the 'comments' button when viewing a recipe's details. Users can use the comments feature to share their thoughts on a recipe with others and provide feedback to the author. This adds to the social aspect of the app, emphasizing the idea that it can be used to network and meet new people in addition to finding recipes.

This feature works by allowing users to enter text into a 'EditText' box on the application. If the user clicks the 'post' button, the comment is sent to Firebase and added to a comments array, with the poster's username stored as part of the comment for easy display. Because each comment is a new entry in this array, it is simple to display on the comments page by using a RecyclerView that cycles through each element of the array, adding each entry as a new RecyclerView object as it goes.

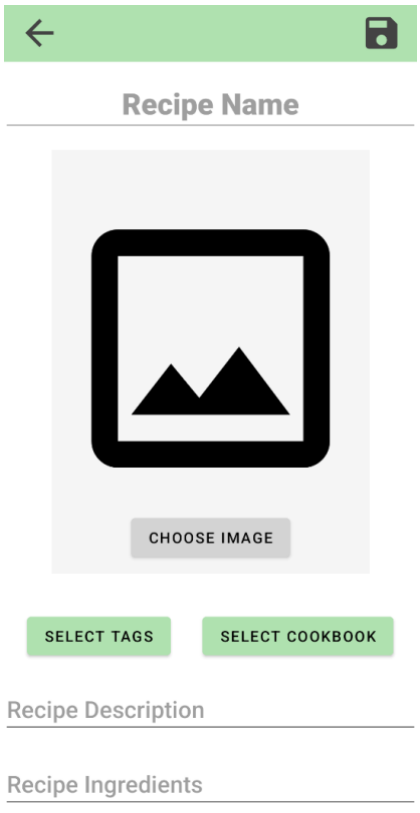
```
Approved: true
Author: "Stephen"
AuthorID: "dhnvWPXjmghBqrcPiG4xDz90SVp2"
Comments
  0 "Stephen wow, cool"
  1 "John epic recipe"
  2 "Stephen I love this"
```

As seen, this makes for easier reading by administrators, and is significantly easier to modify than a string which is concatenated to each time a new comment is posted. The ' ' string is utilised in this case to break the comment up into the poster's username, which is displayed above the comment.

```
// Replace the contents of a view (invoked by the layout manager)
@Override
public void onBindViewHolder(ViewHolder viewHolder, final int position) {

    // Get element from your dataset at this position and replace the
    // contents of the view with that element
    String username = localDataSet[position].split(" ", 2)[0];
    viewHolder.getUsernameView().setText(username);
    String comment = localDataSet[position].replaceAll(".+ ", "");
    viewHolder.getTextView().setText(comment);
}
```

The 2nd half of the comment, after the ' ' string is considered the 'comment' of the array element. This is displayed underneath the posters username.



As mentioned previously, this application also provides users with the opportunity to add their own recipes. This was implemented in the hopes of improving the experience of all users, and thereby increasing user retention. This functionality was implemented by adding a new page to the navigation bar, whereby the user can input the recipe information for processing. This page has a number of text boxes for inputting the information, such as the recipe name, method, description et cetera.

A 'choose image' button is present in order to upload a image for reading by the program. This image is uploaded to a Firebase storage 'bucket', which can then be retrieved at a later time to display.

Inputting this recipe information is required for recipe submission, as once the 'save' button is pressed, this information is gathered by the activity and uploaded as a new Firebase document. Once this has occurred, the recipe will be available for reading by the main page. However, until it is displayed to all users; it must be approved by an administrator as mentioned in chapter 3, section 2.

```
Map<String, Object> data = new HashMap<>();
data.put("Name", editRecipeName);
data.put("Ingredients", editRecipeIngredients);
data.put("Method", editRecipeMethod);
data.put("Desc", editRecipeDesc);
data.put("Likes", 0);
data.put("Approved", false);
data.put("Author", authorName);
data.put("URL", uploadedImage);
data.put("Tags", selectedTags);
data.put("Cookbook", cookbookText.getText().toString());
data.put("AuthorID", authorUserId);
data.put("Comments", null);
```

All newlines within these input boxes are replaced with a line-break string, ";@" which is used by the code to correctly display lines of methods, ingredients et cetera.

A hash map is used to put all this data into one firebase document, with a randomly generated ID number, automatically created once the hash map uploads all data by utilising the db.collection.put function.

Following this, this line of code places the new recipe ID under the current user's 'published recipes' section in the database. This is used to link the new recipes to the publisher's account.

```
userDb = FirebaseDatabase.getInstance().getReference().child("Users").child(userId).child("Published Recipes");
db.collection("Cookbook").add(data).addOnSuccessListener(onSuccess(documentReference) → {
                userDb.child(documentReference.getId()).setValue(true);
```

With this addition, it is possible to display a user's published recipes on their profile page. Similar to the way in which the 'saved recipes' page utilises a recycler view to display these recipes; this system first obtains recipe ID's from the user's profile on the Firebase database which are then cycled through and showcased in a scrollable list.

Each profile page displays information such as the user's name, their friend count and their published recipes in the aforementioned scrollable list.

4.5. Development Issues

This section will discuss some of the major challenges encountered during the project's development. Whilst not exhaustive, some of the most serious problems discovered, as well as difficulties with software and other tools, will be highlighted.

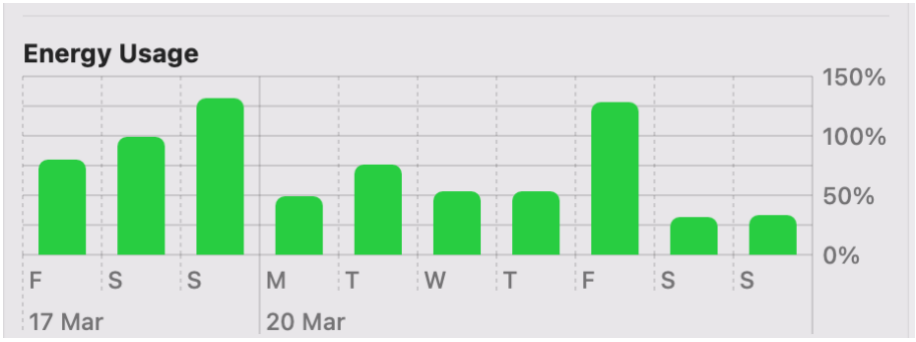
4.5.1 Android Studio

The primary issue with this application's development process were in relation to the shortcomings of AndroidStudio. Whilst it's widely acknowledged as an industry standard for Android app development, [75] is certainly has its issues. [76]

The first, and perhaps most prevalent issue is the very high hardware requirements, especially when compared to other IDE's such as PyCharm, which recommends 4gb of free RAM to run [77] in contrast to AndroidStudios 8gb recommendation. Along with high amounts of CPU power required to run effectively, these requirements can be a serious disadvantage when considering using AndroidStudio to develop applications. As ChefSwipe was developed on a MacBook Air, with 512gb of SSD storage, 8gb of RAM and an M1 processor; some bottlenecking was experienced during the development process. This bottlenecking would be amplified on less powerful or older machines, which can be a serious issue for some developers working with less RAM, or storage space.

Another issue that developers encountered was the slow installation times of AndroidStudio and plugins, which can be inconvenient for some; however, because the program only needed to be installed once, this was not an issue for this project.

Some significant issues whilst developing this application however were the large drain on battery life of any laptop whilst using AndroidStudio, and the system lag produced by AndroidStudio when running in the background. While the system lag caused was not a huge issue, as the generally AndroidStudio was closed when not in use; the impact on battery life of any laptop running the IDE was very noticeable. In the case of this projects development cycle, the battery of the aforementioned MacBook used to develop the application lasted a mere 4 hours, where it would usually last around 8 hours; around a 50% decrease.



This was especially noticeable when development of the project was going on whilst away from an outlet, as it became difficult to manage a sufficient battery level.

Furthermore, some issues presented themselves with regards the android emulator. These issues seem to be quite common among users of AndroidStudio, with problems ranging from very slow load times to issues regarding internet access. [78] These problems were present throughout the development of this project. Each time the emulator is launched, it takes a significant amount of time to fully load and be usable. Whilst this does not make AndroidStudio 'unusable', it is very inconvenient, and slows down the development and testing phases which can be a nuisance whilst under time pressure. In addition to this, emulators are naturally rather bug-prone, which can cause them to boot incorrectly or not at all. This is frequently easily remedied, and the official AndroidStudio documentation site provides excellent documentation for resolving these issues.

4.5.2 Firebase Notifications

Also, as mentioned, there were issues with internet connection on the emulator. This became a large issue when attempting to send notifications using Firebase's built-in messaging service to a device. Upon attempting to send these notifications, it was found that occasionally notifications did not arrive until several minutes after they were sent. As a result of this issue, it was decided to leave notifications out of the current version of the application, as it did not provide an engaging experience for users. Upon conversing with testers about this; many said that whilst notification's would be a great addition to the application, it would be frustrating to receive a notification a lengthy period of time after it was originally meant to be received. Hence, it was decided that notifications would be added at a later date; as discussed in the final chapter of this report.

4.5.3 SwipeCards issues

Yet another issue that was encountered during development was issues with the implementation of the 'SwipeCards' library. Due to the way in which this library functions, there were a number of issues with efficiently reading all documents available in a Firebase database collection in order to be displayed by the on the home page by SwipeCards.

```
public void getRecipe() {
    Query colRef = db.collection( collectionPath: "Cookbook");
    colRef.get().addOnCompleteListener(task -> {
        if (task.isSuccessful()) {
            for (QueryDocumentSnapshot document : task.getResult()) {
                FirebaseDatabase userDatabase = FirebaseDatabase.getInstance();
                FirebaseAuth mAuth = FirebaseAuth.getInstance();
                String userId = Objects.requireNonNull(mAuth.getCurrentUser()).getUid();
                userDb = FirebaseDatabase.getInstance().getReference().child("Users").child(userId);
                DatabaseReference dbRef = userDatabase.getReference().child("Users").child(userId);
                dbRef.addValueEventListener(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
```

These issues often presented themselves in the form of 'jittery' loading animations as recipes were swiped away, and slow loading times. To combat this, great care was taken when developing the 'getRecipe' function, used to add recipes to the SwipeCards stack. With this method in place, the 'clunky' animations are considerably minimized, with the only time it occurs being when the end of the recipe list is reached. When this occurs, getRecipe function must be reran; and the for loop to get all the documents is repeated.

4.5.4 Firebase payment

Another significant issue come with the utilisation of Firebase for database services. Whilst Firebase is a fantastic service, which offers many free features such as the hosting of images, user details etc. This project currently uses Firebase's free "Spark" plan, which has no costs when getting started. However, this plan has a limit of 50,000 to monthly active users, 5gb limits on storage, and so on. [84] With its current userbase, this is not an issue for ChefSwipe, however should the application require more storage, the plan may need to be updated to a 'pay-as-you-go' model such as the "Blaze" plan, which adjusts the monthly price of the subscription based on usage of the provided Firebase services, past the limits set by the

"Spark" plan. Should the application require these additional services, capital would need to be spent to support the application; this may require ads being run within the application to support the running costs incurred.

4.5.5 Issues with the shopping list system

As discussed in the 'future development chapter', a shopping list was to be added to the application, which would significantly increase functionality of the application due to the fact that it would reduce user reliance on other applications to create these lists. However, due to the way in which the ingredients list of recipes function; implementing this shopping list system would require a complete overhaul of this system.

```
<TextView
    android:id="@+id/ingredientsView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_constraintTop_toBottomOf="@id/ingredientsText"
    android:layout_marginTop="25sp"
    android:layout_marginLeft="30sp"
    android:layout_marginRight="15dp"
    android:textSize="25sp">
</TextView>
```

Currently, the ingredients list uses a string which is 'broken' into bullet points by a string, using the .split() function in java. This string is then displayed in a singular XML TextView.

However, to implement 'check boxes', the ingredients list would need to be split up, and added to an array per ingredient.

```
<CheckBox
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:ignore="ExtraText">
    android:id="@+id/ingredient1"
</CheckBox>
<CheckBox
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    android:id="@+id/ingredient2"
</CheckBox>
<CheckBox
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    android:id="@+id/ingredient3"
</CheckBox>
```

From here, each ingredient would require its own specific CheckBox XML view. Of course, this would be fine for recipe with a small number of ingredients. However, as recipes could potentially have anywhere from 2-100 ingredients; a large amount of these CheckBox views would be required, with the majority of them being mostly for redundancy.

There may be a different way if implementing this functionality, which is the reason for its inclusion in the 'future works' chapter, however due to these issues, it was decided to leave this service out of the current version of the application.

4.6. Conclusions

Software design, along with the correct software design methodology are an essential part of any project. It is often seen as a roadmap, which outlines the key points within the development cycle of a project.

Software design can be related to the 'foundation' of a project's code [58]. Once we have the 'foundation' of software design in place, other aspects are considered, such as modularity. Modularity is a simple but important concept, whereby code is broken down into smaller sections and reused as necessary, as opposed to one large incomprehensible block. Some other important aspects within software design include:

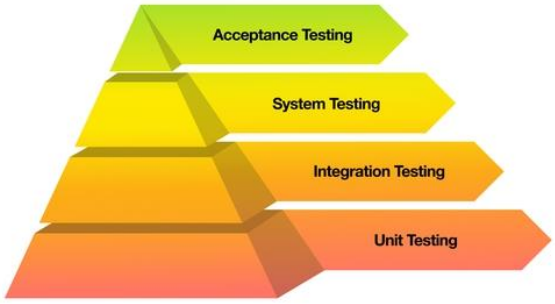
1) Maintainability
2) Flow and functionality
3) Portability

This project has already outlined the portability aspect of software design, in the decision to build for the Android platform. Whilst this decision limits the possibilities for deployment, removing the opportunity for the application to be ported to the Apple App Store, android devices account for over 1 billion devices in the world today (over 70% of the market) [59] from this, we can see that portability has been taken into consideration, however it was decided to build the application for 70% of the market as opposed to building for the minority of the market; the >30% of active iOS devices.

Good software design practices are of great significance when building a large-scale application or project, as they ensure that code is easier to write and understand thanks to good maintainability; which in turn aids in reducing the number of bugs introduced. Well-documented and maintained code is generally less error-prone, [60] which is of interest to developers in the long-run.

5. Testing and Evaluation

This chapter outlines product testing and evaluation of development progress. This chapter will examine project progress and prototypes, in order to aid in finding areas for improvement. This chapter covers the operating software as specified and illustrated in Chapter 4, 'Experiment Development', and reviews the application in its current and final stage of development.



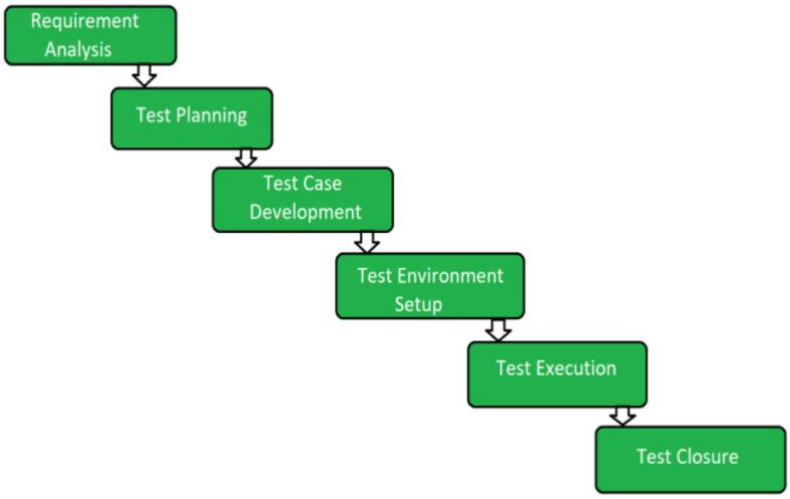
5.1. Introduction

Testing is a continuous process in this application, as per the Agile software methodology. Testing will occur at the addition of each significant feature, via Android Studio's built-in emulation system. [27].

The use of this emulation feature makes testing the application very straightforward, as it cuts out the need for a physical Android device. If required, this device would have to be used continuously during development for the mentioned constant testing. By testing the application at regular intervals, bugs are easily found and dealt with, as small amounts of code are being worked on at a given time between tests.

5.1.1 Test to Break

In addition to constant testing, the 'test to break' approach was also taken. This QA approach attempts to recreate a 'worst-case' scenario, where a user performs an action within the application which causes an error, or crashes the app. Using this approach, the application is put under heavy load, where multiple actions are performed in quick succession, and the tester attempts to 'break' the app using unexpected inputs.



The above image, courtesy of geeksforgeeks [57] outlines the software testing life cycle, or STLC. This model assumes multiple teams working on a project, but can also be applied to smaller projects. This process is slightly more drawn out than a simple test, however it involves requirement gathering and the sorts, whereby information required for the test is gathered. This information, for example, might be the use-case of the application. Testers can be given a persona to follow the use-case of in order to test the app from the perspective of a given 'character'.

Evaluation and testing of an application is a highly important stage in the development cycle, as it forces developers to consider the progress made on the application, contrasted with the

time spent on development. This can help create a more refined focus, so that expectations can be adjusted accordingly.

This stage is also an important time to consider each component of the application, and ask if it is carrying out its role in a successful and efficient manner. If it is not, re-evaluation of that specific feature must be carried out in order to remedy this issue.

5.1.2 Evaluation

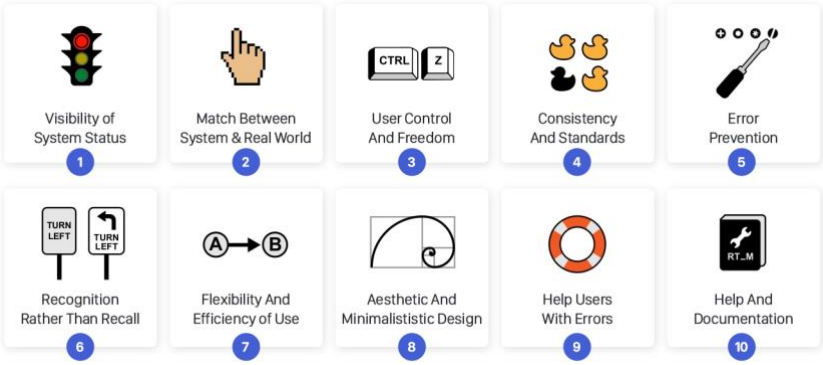
Evaluation may also be carried out by external testers. Once the application is in a stable state, the project can be handed over to external testers for evaluation.
These external testers will use the application in the same way that any other user would. This will be the opportunity to gather feedback on the application in its state, learn from real potential customers which features should be edited, or what design choices need to be altered in order to establish a more fine-tuned and well-polished end product. This phase of evaluation is crucial, as it takes into consideration the thoughts and opinions of outside sources, which have had no input on the creation of the application to that point.

5 features often deemed important regarding evaluation of software are as follows:

1) Finance: does this project have a high return on investment? (ROI)
2) Customer satisfaction: are clients satisfied with the application? (in its current state)
3) Product engagement: Does the application have high user retention?
4) Agile metrics: Are features on track, and updated as per Kanban et cetera?
5) Deliverables: Are features being developed on time, and as scheduled?

(KPIs courtesy of KMS Solutions [56])

5.1.2 Neilson's Heuristic Model



Following this testing methodology came real-world testing in the form of external users. This portion of evaluation was carried out primarily using the Neilson's heuristic model. This model is essentially a set of 10 principles which are utilized to analyze any potential downsides to the current design choices present in the application. These principles are:

1. Visibility of system status: Users must be kept informed about what is going on in the application at all times, I.E the application should not become unresponsive whilst

loading, without notifying the user.

2. Match between system and the real world: The application communicate in a way which is familiar to users, with understandable concepts and words. Information should be arranged in a logical manner.

3. User control and freedom: Users may select unintended functions by accident, necessitating the existence of an 'emergency exit' option, so that the application does not complete the unintended action. Undo and redo functionality is encouraged.

4. Consistency and standards: Users should not have to question if different words, contexts or actions imply the same thing. Platform conventions should be observed, especially with features such as icons or navigation.

5. Error prevention: No matter how effective error messaging may be; a well design-designed application that avoids a problem from occuring in the first place is still always preferred

6. Recognition rather than recall: Users should recognise actions and choices, instead of requiring them to recall niche information from another page of the application. In other words, it is preferred that the user can work out how to utilise features themselves without having to learn and remember how.

7. Flexibility and efficiency of use: Shortcuts may speed up interaction with the application for expert users, but are not an essential aspect for novice users to learn in order to use the application.

8. Aesthetic and minimalistic design: Interfaces should not contain irrelevant information, and should stay as de-cluttered as possible. Every additional unit of information in an interface competes with the relevant pieces of information, lowering their relative visibility.

9. Help users recognize, diagnose and recover from errors: Error messages should be written in straightforward language, in order to describe the problem, and offer a solution.

10. Help and documentation: In an ideal world, the application would not need explanation further to what is present in the application. However, should this not be the case, documentation should be available to access by users to explain aspects of the application.

This chapter will outline user testing carried out as per these principles, including a number of tester-specific opinions regarding the different heuristics in relation to the application.

These testing methodologies are all carried out in the hopes of improving the user's experience with the finished project.

5.2. System Testing

As part of the system testing phase of this project, testers were provided a document in which they were asked to rate the severity of each Neilson's heuristic principle on a scale of 0 to 4. This scale considers that a score of 0 indicates that there are no difficulties with this region, while a score of 4 indicates that there is a usability catastrophe.

The goal of carrying out this testing in such a manner is to identify problem areas with the application, which can be improved upon. Included is a screenshot of an expert user evaluation form which was filled out after a tester had used the application for a 10 minute duration. This testing was carried out in multiple phases, the first 2 of which are documented below. After these testing phases; testing continued as recommended as per Neilson's heuristics, until all issues presented were resolved or reduced in scale.

5.2.1 System Testing Phase 1

Nielsen's Heuristics	Severity Rating	Problems
1. Visibility of System Status	1	Add an animation to show when a user has liked something
2. Flexibility and efficiency of use	2	Change icons like 'saved' list and filter button
3. Recognition rather than recall	1	Change icons to be more universal, and add some hints on how to use it
4. Help and documentation	3	No help on using the app, need to add maybe a help page & tips on swiping and saving recipes
5. Match between system and real world	0	Good and reacts to when user swipes, familiar design to dating apps which is easy to understand, interactive
6. User control and freedom	1	Fix navigation issues and inconsistencies in methods of navigation (nav bar vs separate buttons on saved recipes)
7. Consistency and standards	1	Add a navigation bar to the saved recipe view page, as it's confusing having buttons on top instead of in a navigation bar
8. Aesthetic and Minimalist design	1	Improve some text sizes for readability & change icons to be more universal, but otherwise good
9. Error prevention	0	Delete button moved to 'recipe view' so user doesn't accidentally delete a saved recipe
10. Help users recognise, diagnose and recover from errors	1	Add some text to tell user when there's no recipes available (maybe due to internet) or when they've ran out of recipes to swipe on

As seen in this form, there are a number of areas which should not pose a significant problem, whilst some areas require attention and should be addressed before the final application is published. One such area, graded with a '3' on the Neilson's Severity Rating scale is 'Help and Documentation'. The reason for such a poor score was due to the fact that there was no documentation or tips present on the application, which confused the tester.

The tester in this case was required to work out what the application did, feature by feature as opposed to immediately understanding the elements of the interface. This was also an issue due to the inaccurate and unconcise icons for certain features of the application; namely within the navigation bar.

To remedy these issues, icons were changed throughout the application based on user feedback in order to allow for recognition of features, rather than recall of their outcomes. After carrying out the expert evaluation after suggested changes were implemented and considered, the tester submitted a new form, which can be viewed below:

5.2.1 System Testing Phase 2

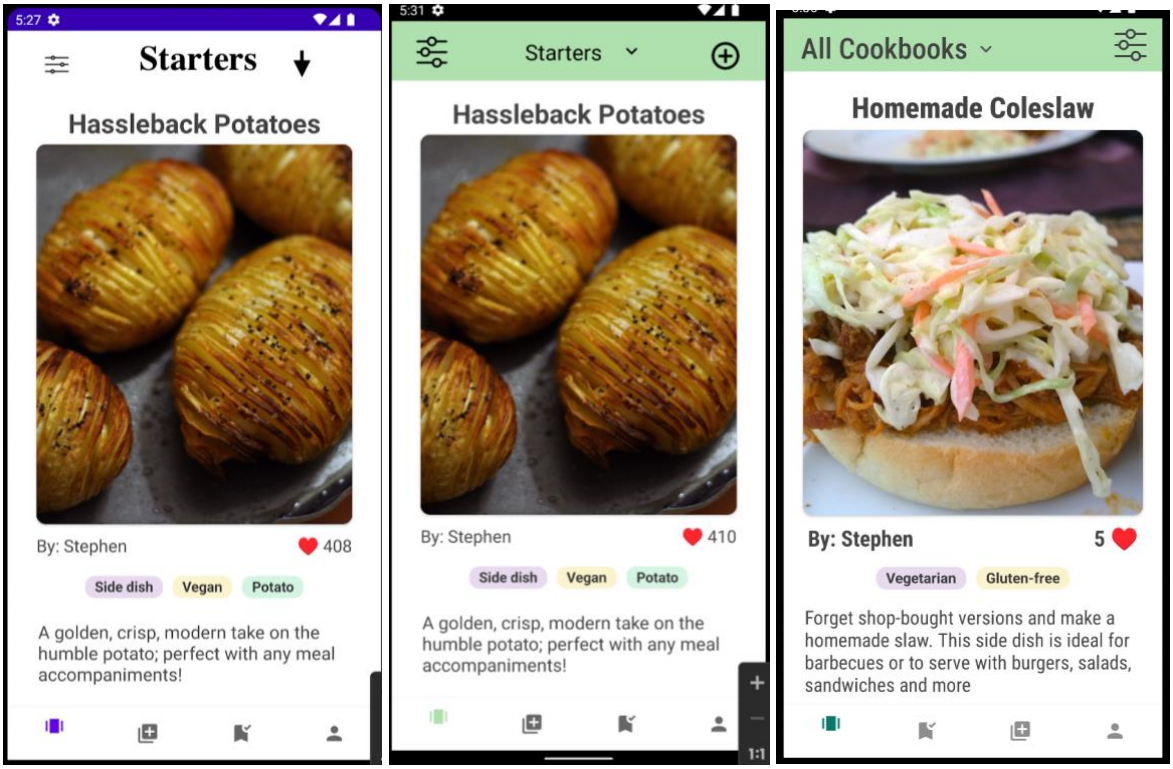
Nielsen's Heuristics	Severity Rating	Problems
1. Visibility of System Status	1	Add more animations maybe to make it easier to understand when you've submitted recipes
2. Flexibility and efficiency of use	0	No problems with the icons anymore, looks much nicer than it did
3. Recognition rather than recall	0	Icons look good and I understand what I need to click a bit more now
4. Help and documentation	2	Could still use some tips or a page which shows you what to do, but easier to use now anyway
5. Match between system and real world	0	Easy to use, same as before works as you'd expect it to
6. User control and freedom	0	Navigation bar was fixed and it's much easier to exit parts of the app now like the recipe creation page
7. Consistency and standards	0	Back buttons work better now since they're in the top bar of the page instead of on the bottom
8. Aesthetic and Minimalist design	1	Much easier to read text, still some text on the saved recipes page which is a little bit hard to read
9. Error prevention	0	No errors when I was using the app it worked well, it was a bit slow and choppy when I went to a different page though
10. Help users recognise, diagnose and recover from errors	0	Added the text to say there's no saved recipes and so on, makes it easier to understand. Still no errors

As per this round of testing, we can see the tester is much happier to use the application, and has identified fewer problem areas after the updates had been implemented. Once again however, one specific area which they still seem to have an issue with is the 'Help and Documentation' heuristic. This is something which must be considered on future iterations of the application, as according to the tester, a 'tips' page or splashscreen would be of great benefit.

Whilst Neilson's Heuristics is a very useful tool to test and evaluate during the development processes, this methodology does have its drawbacks. [72] Some of which downsides include:

5.3. System Evaluation

The following 3 screenshots outlines a number of iterations of the 'title bar' design, where user feedback was utilized in order to choose to right placement, styling and icons for this feature.

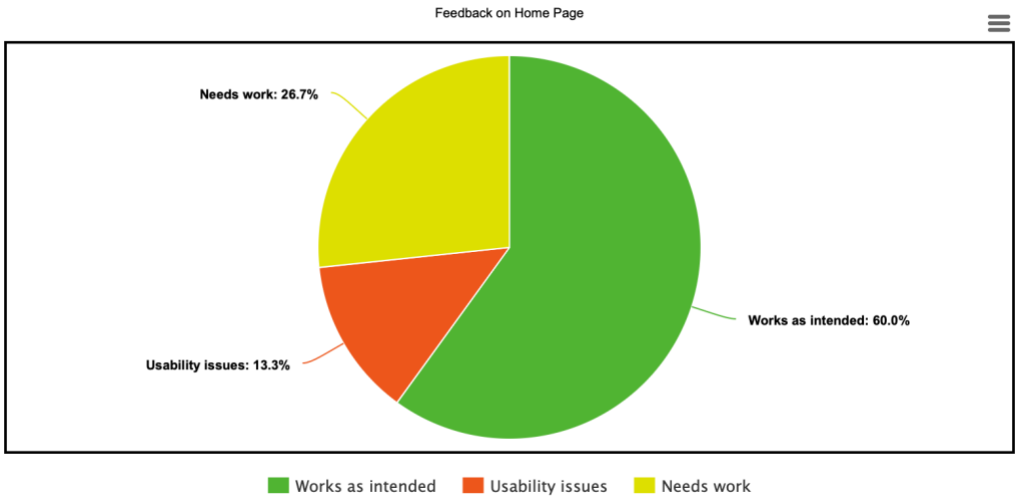


As seen in the 1st screenshot, the prototype title bar was implemented. This mock-up did not have any functionality, and it's absence of color caused it blend in with the app's white backdrop. However, testers concluded that, with a few tweaks, this would be an sufficient title bar design for the application.

Screenshot 2 showcases a number of tweaks to the title bar, once the button functionality had been created and linked up to the present buttons. The colour scheme of the application was also changed from a deep purple to a natural green. Whilst purple colour themes may suggest a trendy, successful application; it wsa decided that a natural and earthy green would be more inviting for users of all ages. [67] Whilst changes were successful with testers; the overall design of the title bar looked significantly more cluttered than the original and so was not suitable for the app's final edition.

In the 3rd screenshot, we see the final iteration of the title bar, along with some minor tweaks to text and colours in the hopes of improving font and icon visibility. As seen, the title bar has been decluttered, with filter buttons taking up less space, where they are clearly separated from the 'swipe' section of the home page. This design was very successful with testers, who thought it would be excellent for the completed product.

Following the production of this 3rd screenshot, a survey was carried out where users were asked to test the final iteration of the page. They were asked to give feedback based on their experience with the application home page in the state presented. These feedback tiers were; 'Works as intended', 'Needs work', and 'Usability issues'.



As seen in the above pie chart, a small number of users, 13.3% stated that they had usability issues with the home page. These issues were due to bugs with device internet connection, alongside incorrect recipe document fields, which have been corrected thanks to user feedback. 26.7% of users stated that the home page "needs work", with the majority of this feedback revolving around that fact that there was a lack of explanation of what to do, and how to interact with the page. This has yet to be fixed in the current version of the application, which assumes the user understands the basic concept of how to interact with an application with this generic design. Finally, 60% of users stated that the home page 'works as intended', and there were no significant issues present when they performed testing on the application.

The following table outlines current issues and suggestions as outlined by testers, which are to be implemented or fixed in future iterations of the application.

Type	Issue/suggestion	Description
Issue	Slow animations	Animation when switching pages is a bit slow sometimes
Suggestion	Tutorial	Need a tutorial on how to use the main page
Suggestion	Affiliate links	Should add affiliate links to be able to sell tools and items for recipes
Suggestion	Shopping list	Should add a shopping list like a 'to do' list so you don't have to use other apps on your phone when shopping
Issue	Filter problems	Recipes can only have 3 tags but a recipe could potentially have 4+ possible tags. This means sometimes they don't show up when filters are added
Suggestion	Ingredient conversions	Should be able to convert ingredients, for example grams to pounds instead of needing to Google the conversion

User evaluations such as the session outlined above proved to be very beneficial throughout the development phases of this project, as predicted based on research into the importance of usability testing. [69] Thanks to AndroidStudio's aforementioned built-in android emulator, [70] it was possible to perform on-the-fly changes to the application as testing was in progress. This method, sometimes referred to as 'DevTest' [71] was carried out by having testers use the emulator to interact with the program, where their feedback was recorded. Changes would be quickly prototyped in a short period of time, where, for example colour schemes were changed or icons were edited. Once these changes had been implemented, the tester would once again use the emulator and provide further feedback. This cycle would continue until the tester was satisfied with the state of the interface, or feature.

5.4. Testing and Evaluation Conclusions

Overall, the testing and evaluation portion of any project is of the upmost importance, as a simple bug, no matter how small could cause catastrophic failure of the application. This must be avoided at all costs; hence, the importance of continuous testing is made evident.

As also outlined in section 5.1, the 'testing' period is a fantastic time to test the application using personas, and specific use-cases. This allows developers to get an in- depth perspective into the application from the eyes of a 'real' potential customer. With this information, the application's features can be evaluated further.

Following on from this, thanks to the evaluation phase of this process, developers can learn from the information and data they have gathered on the program, and work to remedy any issues that may have arisen. This also provides the opportunity to go through any proposed design changes, from anything as simple as font size, all the way to the functionality of a search bar, for example.

Tester opinions and reviews were compiled into one complete list, which was then used throughout the remainder of the development process. It was the intention to implement any significant suggestions made by testers, as all opinions were valued greatly. Incorporating tester ideas will aid in ensuring that the user experience of the application resembles a finished and polished product.

The 'DevTest' method of testing proved incredibly useful in the testing and evaluation portion of this application, however it can be very time consuming. This evaluation method is also not as efficient as other testing methodologies, as it requires a tester to be present for the development phases, and requires the developer to be present for the testing phases. Whilst this can be a disadvantage to the DevTest method, it seemed to work well for this particular project.

6. Conclusions and Future Work

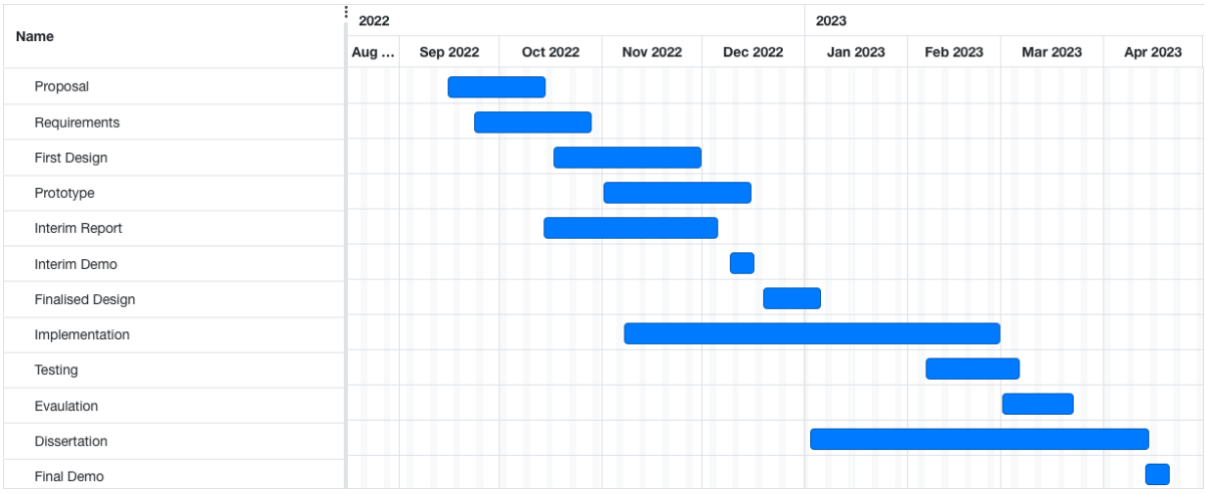
The project's future work is summarized in this chapter, along with the conclusion that has been reached after many months of development and research that have been carried out as part of this project.

6.1. Introduction

This chapter is divided into sections, the first of which will highlight the future plans for the development cycle of this project will be highlighted. This section will include features that may be worked on to enhance and improve the published application. It will also entail any design changes that may have to be carried out in future iterations of the application. This section will also possess information on future application testing and deployment. Because this application is being developed for the Android platform, the finished application could be uploaded to the Google Play store. This chapter will investigate the viability of this option.

Secondly, the project's conclusion will be discussed. Many things have been discovered over the span of this project, such as flaws in the chosen IDE, AndroidStudio, and disadvantages of using a service such as Firebase. These and other insights will be discussed in the final conclusion, along with in chapter 4.5, 'development issues'.

6.1.1 GANTT Chart



6.2. Future Work

The plan development plan for this project is to coincide with the provided wireframes and plan for development as outlined in Chapter 3 and 4. This includes the development of any and all features not currently present in the prototype of the application.

Some, of these features include:

- Forums or discussion boards
    - Forums or discussion boards for users would promote greater interactivity among the community, allowing users to have conversations and communicate regarding recipes and other elements of the application. Though not a significant feature, this would strengthen the sense of community within the application.
- Notifications
    - This would allow for users to get notified when, for example; their friend posts a recipe, or a recipe which may be of interest to them is posted. Currently, this issue is not implemented due to issues with the emulator's Android version accepting notifications, alongside problems linking with Firebase User ID, as outlined in chapter 4.
- Personalised feed page
    - A personalised feed page would allow for users to get recommended recipes based on their currently or previously saved recipes. This feature may increase user retention, however would be rather difficult to implement due to its reliance on machine learning.
- Social sharing features
    - Options to share recipes to other applications, such as WhatsApp [73] or Facebook [31]. This may have significant benefits for user population, as it may entice potential users; whom have seen the shared post, to download the application.
- Convert measurements
    - In the current iteration of the application, measurements are provided in the recipe author's chosen format, such as metric or imperial. For example, as some people use different measurements, a user may not know what 100g of sugar equates to in cups (1/2 a cup). It was proposed by a tester that a system should be implemented to automatically convert measurements to the specific user's chosen measurement type. However, this feature was too complex to implement on the time constraints present.
- Shopping list
    - A shopping list feature was to be added to this application, which would aid users by providing them a simple way to create a list of required ingredients for recipes they wish to create. However, due to the implementation of the recipe list in the current iteration of the application; a complete overhaul of the ingredients list would be required in order to implement this. Again, due to the time constraints present, this feature was left out of the submitted version of the application.
- Affiliate links
    - Yet another potential addition to the app would come in the form of affiliate links, which would be links to tools used in recipes, such a whisks, strainers etc, which may be used to generate revenue for the application should a user

choose to make a purchase after utilising an affiliate link. These could be provided by sites such as Amazon [85], and would add a means to fund features such as more Firebase storage for increased user capacity.

As is stated within the software methodology section, an Agile development plan will be put in place to carry out this development process. Tasks will be split into smaller sub- parts in order to stay on track, and to coincide with the GANTT chart presented below, in section 5.3.1.

In addition to the programming of this application being worked on, documentation will be continually updated to outline any features as they are added. This information will be available on the project's GitHub page, as previously stated within this report [16]. This documentation will keep track of major changes to the development process, highlighting any information regarding the development progress of the features outlined above.

Finally, as this is an Android application, the possibility arises to publish 'ChefSwipe' to the Google Play store, where it can be downloaded by anyone, all over the world. The process of publishing an application is very straightforward, and requires a one-time-free of $25 to set up a developer account. Once this account is created, the app may be uploaded, at which time Google will begin the review process of the application. [74] The application is not currently live on the Google Play store, but the possibility of publishing it is very exciting, and it is planned to be done in the near future.

6.3. Conclusions

In conclusion, having set out to create an Android recipe-oriented application which functions similarly to popular dating applications such as 'Tinder' [52]. The goal was simple; to create a unique and engaging user experience by which users could discover new culinary delights and develop their skills as a chef.

After the research carried out outlined in this document; alongside the complete development process, the application is in a finished and satisfactory state. This application possesses the required and fundamental features as originally proposed, alongside many others which were implemented to further improve upon the user experience.

The application also has various sophisticated functions, in addition to the basic features already mentioned. Such features include recipe creation, for future sharing with other users of the application, and the filtering of these recipes based on the specific requirements of the users. The purpose of adding these features was to dramatically improve user experience, which was proven to be effective thanks to of project's testing procedure.

The lengthy development process of this project included many different phases, starting with the planning of the project, with research into existing solutions and other final year projects. Following this was the design phase, during which the project's layout was thought-out and planned. This phase included user interface design as well as code architecture, which was supplemented by a chapter regarding the development process of the project. This chapter showcased some of the elements of the application, with screenshots of the code and running application. As in any project came testing, which ensured a high quality output, and ensured that all requirements set out were adhered to. As a result of this testing phase, it was confirmed that all initial requirements were successfully met to a sufficient

standard. With this in mind, it is safe to assume that the application development phase was a success; those whom decide to make use of the application can now find new culinary dishes, learn new techniques and further enhance their cooking abilities.

To summarize, this project successfully created a recipe-oriented application which achieves the goal of functioning similarly to popular dating applications. This application, according to testers and users alike, provides a useful platform for not only culinary experts, but for average home-cooks to utilise in order to connect with others and further their cooking skills. It is hoped that in the future, with further development, this application will be a useful tool and a staple of home-cooking.

Bibliography

[1] BBC Good Food [Online] Available from:
www.bbcgoodfood.com
[Cited 30/Sept/2022] Author: BBC

[2] SuperCook - Recipe Generator [Online] Available from:
https://play.google.com/store/apps/details?id=com.supercook.app&hl=en_IE&
[Cited 30/Sept/2022] Author: Supercook

[3] Google Firebase [Online] Available from:
https://firebase.google.com/docs
[Cited 01/Oct/2022] Author: Google

[4] Android Studio Documentation [Online] Available from:
https://developer.android.com/docs
[Cited 01/Oct/2022] Author: Google, JetBrains

[5] Importance of App Design [Online] Available from:
https://blog.duckma.com/en/sample-blog/mobile-app-design-critical/
[Cited 01/Oct/2022] Author: Redazione DuckMa

[6] Android Studio Emulator Documentation [Online] Available from:
https://developer.android.com/studio/run/emulator
[Cited 01/Oct/2022] Author: Google, JetBrains

[7] Java vs Kotlin [Online] Available from:
https://kruschecompany.com/kotlin-vs-java/
[Cited 01/Oct/2022] Author: Patrick Böllhoff

[8] eBook statistics [Online] Available from:
https://wordsrated.com/ebooks-sales-statistics/
[Cited 09/Oct/2022] Author: Danny McLoughlin

[9] Instagram [Online] Available from:
Instagram.com
[Cited 09/Oct /2022] Author: Meta

[10] How Social Media Firms Moderate their Content [Online] Available from:
https://knowledge.wharton.upenn.edu/article/social-media-firms-moderate-content/
[Cited 09/Oct/2022] Author: Knowledge at Wharton

[11] How does Content Filtering Work? [Online] Available from:
https://www.techtarget.com/searchsecurity/definition/content-filtering
[Cited 10/Oct/2022] Author: Peter Loshin

[12] Adding Firebase to an Android App [Online] Available from: https://firebase.google.com/docs/android/setup
[Cited 11/Oct /2022] Author: Google

[13] Object-Oriented Programming in Java [Online] Available from: https://www.freecodecamp.org/news/object-oriented-programming-concepts-java/
[Cited 11/Oct/2022] Author: Patrick Cyubahiro

[14] Benefits of Using Java for Android App Development [Online] Available from: https://www.explorateglobal.com/blog/java-app-development-benefits/
[Cited 11/Oct/2022] Author: Explorate

[15] Advantages of XML [Online] Available from: https://www.ibm.com/docs/en/i/7.3?topic=introduction-advantages-xml
[Cited 11/Oct/2022] Author: IBM

[16] Project page [Online] Available from: https://github.com/StephenHalligan/Chef-Swipe
[Cited 02/Dec/2022] Author: Stephen Halligan

[17] Advantages of GitHub [Online] Available from: https://apiumhub.com/tech-blog-barcelona/using-github/
[Cited 11/Oct/2022] Author: Ekaterina Novoseltseva

[18] E-cookbooks: Love 'em or hate 'em? [Online] Available from: https://www.eatyourbooks.com/blog/2014/02/25/e-cookbooks-love-em-or- hate-em
[Cited 11/Oct/2022] Author: Susie

[19] eBooks vs. Print books [Online] Available from: https://www.investopedia.com/financial-edge/0812/e-books-vs.-print- books.aspx
[Cited 11/Oct/2022] Author: Margaret James

[20] eBook Global Market [Online] Available from: https://www.futuremarketinsights.com/reports/global-eBook-market
[Cited 11/Oct/2022] Author: Future Market Insights

[21] eReaders vs. Books [Online] Available from: https://commercialwaste.trade/e-readers-vs-books-better-environment/
[Cited 11/Oct/2022] Author: Breton Towler

[22] eBooks vs. Print Books [Online] Available from: https://theprintauthority.com/ebooks-vs-printed-books/
[Cited 11/Oct/2022] Author: The Print Authority

[23] Mycookbook.io API [Online] Available from: https://www.mycookbook.io
[Cited 12/Oct/2022] Author: Techno Foodies

[24] History of Cookbooks [Online] Available from:
https://eatwell.healthy.ucla.edu/2021/03/08/the-history-of-cookbooks-and-where-we-are-today/
[Cited 13/Oct/2022] Author: uclahci

[25] Papers books VS eBook Stats [Online] Available from:
https://www.tonerbuzz.com/blog/paper-books-vs-ebooks-statistics/
[Cited 13/Oct/2022] Author: Rob Errera

[26] Increase in Baking Cookbook Sales article [Online] Available from:
https://www.npd.com/news/press-releases/2021/the-great-us-baking-craze-continues-with-42- increase-in-baking-cookbook-sales-npd-says/
[Cited 13/Oct/2022] Author: NPD

[27] Android Studio Emulation [Online] Available from:
https://developer.android.com/studio/run/emulator
[Cited 14/Oct/2022] Author: Google, Jetbrains

[28] Is interactivity a major factor in an app's success? [Online] Available from:
https://habr.com/en/post/486270/
[Cited 14/Oct/2022] Author: Patricia Neil

[29] Struggling to Stay Vegan article [Online] Available from:
https://theminimalistvegan.com/struggling-to-stay-vegan/
[Cited 15/Oct/2022] Author: Minimalist Vegan

[30] Recipe APIs [Online] Available from:
https://rapidapi.com/blog/recipe-apis/
[Cited 17/Oct/2022] Author: RapidAPI staff

[31] Facebook home page [Online] Available from:
http://facebook.com
[Cited 18/Oct/2022] Author: Meta

[32] Tinder "Swipe left, swipe right — but why?" article [Online] Available from:
https://uxdesign.cc/swipe-left-swipe-right-but-why-tinder-ux-ui-simple-dating-mobile-app- swiping-design-4d2295d80407?gi=c9f9bdb4bf44
[Cited 18/Oct/2022] Author: Josep Ferrer

[33] Twitter functionality [Online] Available from:
https://www.enjoyalgorithms.com/blog/design-twitter
[Cited 18/Oct/2022] Author: EnjoyAlgorithms

[34] Newspapers facing extinction within 20 years [Online] Available from:
https://www.telegraph.co.uk/news/2021/03/05/newspapers -face-extinction-within-20-years- cannot-update-like/
[Cited 18/Oct/2022] Author: Telegraph

[35] Importance of Wireframing [Online] Available from:
https://www.zebedeecreations.com/blog/the-importance-of-wireframing/ [Cited 19/Oct/2022] Author: Karen Oliver

[36] Real-time chat app built using Firebase [Online] Available from:
https://www.javatpoint.com/creating-a-real-time-chat-application-using- firebase
[Cited 19/Oct/2022] Author: Javatpoint

[37] Why physical books still outsell eBooks [Online] Available from:
https://www.cnbc.com/2019/09/19/physical-books-still-outsell-e-books-and-heres-why.html
[Cited 21/Oct/2022] Author: Lucy Handley

[38] PEW Research [Online] Available from:
www.pewresearch.org
[Cited 22/Oct/2022] Author: PEW Research

[39] BBC Good Food website - mobile application [Online] Available from:
https://www.bbcgoodfood.com/get-bbc-good-food-app
[Cited 24/Oct/2022] Author: BBC

[40] BBC Good Food app store page [Online] Available from:
https://play.google.com/store/apps/details?id=uk.co.bbc.goodfood2&hl=en_IE&gl=IE
[Cited 24/Oct/2022] Author: Immediate Media Co.

[41] Application monetization [Online] Available from:
https://www.appsflyer.com/resources/guides/app-monetization/
[Cited 28/Oct/2022] Author: ApplsFlyer

[42] Cookpad website [Online] Available from:
https://cookpad.com/us
[Cited 01/Nov/2022] Author: Cookpad Team

[43] Cookpad app store page [Online] Available from:
https://play.google.com/store/apps/details?hl=en&id=com.mufumbo.android.recipe.search
[Cited 01/Nov/2022] Author: Cookpad

[44] Object oriented programming definition [Online] Available from:
https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP
[Cited 04/Nov/2022] Author: Alexander S. Gillis

[45] Benefits / advantages of Object oriented programming [Online] Available from:
https://www.geeksforgeeks.org/benefits-advantages-of-oop/
[Cited 04/Nov/2022] Author: archisoni999

[46] VS Code requiring log-in every time [Online] Available from:
https://github.com/microsoft/vscode/issues/141312
[Cited 06/Nov/2022] Author: GitHub, kgfly

[47] Twitter search function [Online] Available from: https://help.twitter.com/en/using-twitter/twitter-search
[Cited 07/Nov/2022] Author: Twitter

[48] Most common programming languages [Online] Available from:
https://idratherbewriting.com/2014/12/22/most-common-programming-languages-tech-writers-in-my-survey-know/
[Cited 08/Nov/2022] Author: Tom Johnson

[49] SwipeCards library GitHub page [Online] Available from:
https://github.com/Diolor/Swipecards
[Cited 12/Nov/2022] Author: GitHub, Diolor

[50] Google Firebase Terms of Service [Online] Available from:
https://firebase.google.com/terms
[Cited 12/Nov/2022] Author: Google

[51] Trello board home page [Online] Available from:
http://trello.com
[Cited 14/Nov/2022] Author: Trello

[52] Tinder home page [Online] Available from:
http://tinder.com
[Cited 17/Nov/2022] Author: Tinder

[53] TikTok home page [Online] Available from:
http://tiktok.com
[Cited 17/Nov/2022] Author: TikTok

[54] Top 6 Software Development Methodologies [Online] Available from:
https://blog.planview.com/top-6-software-development-methodologies/
[Cited 21/Nov/2022] Author: Brook Appelbaum

[55] Agile Manifesto [Online] Available from:
https://agilemanifesto.org
[Cited 21/Nov/2022] Author: Agile

[56] Evaluating Software Development [Online] Available from:
https://blog.kms-solutions.asia/how-to-evaluate-your-software-development-project-success
[Cited 26/Nov/2022] Author: KMS Solutions

[57] Software Testing Life Cycle [Online] Available from:
https://www.geeksforgeeks.org/software-testing-life-cycle-stlc/
[Cited 01/Dec/2022] Author: pp_pankaj

[58] Why is Software Design important? [Online] Available from: https://www.mindbowser.com/why-software-design-is-important/ [Cited 01/Dec/2022] Author: MindBowser

[59] How many android users are there worldwide? [Online] Available from: https://sortatechy.com/android-users-are-there-worldwide/ [Cited 03/Dec/2022] Author: Sortatechy

[60] Importance of Keeping code DRY [Online] Available from: https://deegloo.com/the-great-importance-of-keeping-the-code-dry/ [Cited 03/Dec/2022] Author: Mateo Mustapic

[61] Image Copyright on Social Media [Online] Available from: https://blog.hootsuite.com/understanding-image-copyright/ [Cited 04/Dec/2022] Author: Christina Newberry

[62] Copyright – DETE [Online] Available from: https://enterprise.gov.ie/en/what-we-do/innovation-research-development/intellectual-property/copyright/ [Cited 04/Dec/2022] Author: Department of Enterprise

[63] Copyright and Regulated Rights Act, 2000 [Online] Available from:3 https://www.irishstatutebook.ie/eli/2000/act/28/section/140/enacted/en/ht ml [Cited 04/Dec/2022] Author: Irish Statute

[64] Is Society Too Dependent on Technology [Online] Available from: https://www.uopeople.edu/blog/society-too-dependent-on-technology/ [Cited 06/Jan/2023] Author: University of the People

[65] Sustainable Development Goals [Online] https://unglobalcompact.org/library/5181 [Cited 20/Jan/2023] Author: United Nations

[66] "The Importance of Good UI/UX in Mobile App Development" https://www.forbes.com/sites/forbestechcouncil/2019/05/23/the-importance-of-good-uiux-in-mobile-app-development/?sh=3ff7b3952712 [Cited 23/Jan/2023] Author: Adam Fingerman, Forbes

[67] What is scrum? https://www.atlassian.com/agile/scrum [Cited 26/Jan/2023] Author: Claire Drumond

[68] SDLC – Waterfall Model https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm [Cited 26/Jan/2023] Author: Tutorialspoint

[69] 5 Amazing Benefits of Usability Testing https://pollthepeople.app/usability-testing-benefits/ [Cited 16/Feb/2023] Author: Owen Fay

[70] The Pros and Cons of Android Studio and App Tools
https://www.pangea.ai/dev-mobile-app-resources/the-pros-and-cons-of-android-studio-and-app-tools/
[Cited 17/Feb/2023] Author: Octavia Drexler

[71] What is development testing (DevTest)?
https://www.netapp.com/devops-solutions/what-is-development-testing-dev-test/
[Cited 17/Feb/2023] Author: NetApp

[72] What is heuristic evaluation
https://worship.agency/what-is-heuristic-evaluation
[Cited 17/Feb/2023] Author: Worship

[73] WhatsApp home page
https://www.whatsapp.com
[Cited 18/Feb/2023] Author: Meta

[74] How to Upload an App to Google Play Store
https://appinventiv.com/blog/how-to-submit-app-to-google-play-store/
[Cited 18/Feb/2023] Author: Avinash Sharma

[75] Why Android Studio is Awesome
https://medium.com/@agicent/why-android-studio-is-awesome-c606c94366e6
[Cited 23/Feb/2023] Author: Agicent

[76] AndroidStudio reviews
https://www.trustradius.com/products/android-studio/reviews?qs=pros-and-cons#reviews
[Cited 28/Feb/2023] Author: TrustRadius, AndroidStudio user community

[77] PyCharm system requirements
https://www.jetbrains.com/help/pycharm/installation-guide.html#requirements
[Cited 03/Mar/2023] Author: PyCharm

[78] Troubleshoot known issues with Android Emulator
https://developer.android.com/studio/run/emulator-troubleshooting
[Cited 03/Mar/2023] Author: AndroidStudio team

[79] “Why are some apps so full of ads?”, Quora
https://www.quora.com/Why-are-some-apps-so-full-of-ads
[Cited 14/Mar/2023] Author: John Panzera, Quora

[80] Mob Kitchen homepage
https://www.mob.co.uk
[Cited 17/Mar/2023] Author: Mob kitchen, Mouth Group Limited

[81] Peckish application announcement page
https://www.mob.co.uk/life/introducing-peckish
[Cited 17/Mar/2023] Author: Mob kitchen, Mouth Group Limited

[82] Mob kitchen Instagram page
https://www.instagram.com/mob/?hl=en
[Cited 17/Mar/2023] Author: Meta, Mob kitchen

[83] Google Android vs Apple iOS: Data Reveals Which OS Dominates Market Share
https://www.digitalinformationworld.com/2022/04/google-android-vs-apple-ios-data.html
[Cited 19/Mar/2023] Author: Digital Information World, Statcounter.com

[84] Google Firebase Pricing
https://firebase.google.com/pricing
[Cited 24/Mar/2023] Author: Google

[85] Amazon Affiliate Program: How to Become an Amazon Associate to Boost Income
https://blog.hubspot.com/sales/amazon-affiliate
[Cited 24/Mar/2023] Author: Meg Prater