Applying Mobile Learning and Usability Design Principles to Create a Better Learning Experience

Conor Thorne

A Final Year Project

Presented to the University of Dublin, Trinity College

in partial fulfilment of the requirements for the degree of

B.A. (Moderatorship) in Computer Science

Supervisor: Dr. Inmaculada Arnedillo-Sanchez

April 2021

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Conor Thorne

May 3, 2021

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Conor Thorne

May 3, 2021

Applying Mobile Learning and Usability Design Principles to Create a Better Learning Experience

Conor Thorne, B.A. (Moderatorship) in Computer Science

University of Dublin, Trinity College, 2021

Supervisor: Dr. Inmaculada Arnedillo-Sanchez

While there is no distinct definition for mobile learning (m-learning), the term generally refers to learning while mobile and learning on a mobile device. This project researches the design principles of mobile learning and mobile usability, implements them in the context of a cooking mobile application and evaluates the effectiveness of the newly designed system. By doing this, we have demonstrated the validity of mobile learning and usability design research, by applying the design fundamentals to create a more effective and better learning experience. We first conducted research on mobile learning and usability, specifically their definitions, design fundamentals and evaluation methods. We then performed various evaluations using user testing. Our analysis of the user testing showed that by implementing the design principles of m-learning and usability research, our app provided a better learning experience for users than apps already available. We concluded that designers should use mobile learning research to aid the design of learning apps in order to increase the effectiveness that their app's have on learning, as well as improvement the overall usability of their learning app.

Acknowledgments

I'd like to give a special thanks to my supervisor, Inmaculada Arnedillo-Sanchez, for her continued support, advice and mentoring throughout the course of this project. I would also like to thank the participants in the evaluation of the application, who together helped improve the project tremendously. Finally, I'd like to thank my parents, for whom I owe everything.

Conor Thorne

University of Dublin, Trinity College
April 2021

Contents

List of Figures

Chapter 1

Introduction

1.1 Project Goal & Objectives

The goal of this project is to research the design principles of mobile learning and mobile usability, implement them in the context of a cooking application and evaluate the effectiveness of the newly designed system, comparing it with the mobile apps for cooking currently available.

This project aims to reflect the effectiveness of mobile learning research, by applying it in a real world context. The background research will focus on examining mobile learning design, usability design, mobile learning and usability evaluation methods, currently available mobile cooking apps and the technologies needed for the project. This research will then be used to influence the design and implementation of a mobile learning app used to learn cooking lessons. The app will then be evaluated, using the evaluation methods researched as part of this project. The results from the evaluations will be used to influence the design of the app and also to compare the effectiveness of the system's final implementation at teaching lessons, in comparison to the already available apps. The project aims to demonstrate that by applying mobile learning and usability research, a more effective solution for learning can be implemented resulting in a better learning experience.

1.2 Context of Problem

The overall goal of the project is to provide a system for users to learn to cook for themselves. An app that successfully provides this service could potentially solve issues with unhealthy lifestyles. By using a mobile app to learn how to cook food at home, users

would be eating home cooked meals and avoiding unhealthy alternatives. According to a 2019 National Health Survey, commissioned by the Department for Health, 60% of Irish adults are overweight or obese. The study also found that 34% of Irish adults are trying to lose weight. The study cited "poor dietary choices" as one of the main reasons for being overweight of Health (2019) . An NHS study conducted by Hruby A. (2016) demonstrated that adults who are overweight are at a more significant risk of developing serious health problems such as diabetes, cardiovascular diseases, types of cancers and premature death. The study also cited that "maintaining a healthy weight is in large part a function of lifestyle choices" and that a major factor of obesity was an unhealthy diet. Healthy dietary patterns are proven to lower the risk of chronic diseases, cardiovascular diseases and cancer Cena and Calder (2020). A study on the impact of online cooking courses and nutrition resources revealed that online system could incur a positive change in participants' eating and dietary habits Adam M. (2015).

These research studies clearly indicate that being overweight is influenced by an unhealthy diet. An unhealthy diet and food lifestyle will lead to a significant number of health issues and in some cases, death. Research also indicated that a mobile application, which provides cooking lessons, could be used by people who wish to pursue a better dietary lifestyle, by cooking at home using a mobile application to learn recipes. Utilising the power of online learning resources, the app could encourage better nutritional behaviors, reducing users overall risk of obesity.

1.3 Personal Motivation & Goals

The primary motivation of the project is to encourage healthier eating habits and lifestyles by cooking at home. It was our goal to make use of modern technology, innovative design techniques and academic research on mobile learning and mobile usability to develop a mobile application that effectively teaches users how to cook. We think that cooking for yourself is the first step to leading a healthier lifestyle and we wanted to mix our passion for design and gastronomy to create an environment that could effectively encourage healthier eating habits. My own personal goals were to learn how to read and analyse research papers, learn how to develop a mobile application, follow practices learned from design projects, to try user testing and the iterative design approach to development. Most importantly, I aimed to encourage and teach people, who previously believed that they could not cook, that they can and as a result live healthier lives. Essentially, I wanted to use academic research to guide the design and implementation of a mobile app that makes it easy for people to learn cooking and as a result encourage healthier lifestyles.

Chapter 2

Background Research

The aim of the background research of the project is to examine what the research reveals about the design of mobile-learning apps and what testing methods are used to evaluate such systems. We also examine how other apps available approach their design.

As a result, this review primarily focuses on educational design theory, specifically mobile learning and mobile micro learning. We aim to perform an examination into usability, more precisely mobile usability design, with the goal to compile the information on the fundamental design aspects of mobile learning and to compare this to the systems already available. These results will be used as a guideline to design the cooking application. By using mobile learning research to influence the design, the system incorporates the design fundamentals essential to learning through mobile devices. The design will also be influenced by using the evaluation methods reviewed to evaluate prototypes, allowing the system to use user-feedback to evaluates itself. Eventually, the final implementation will be evaluated to examine if the incorporation of mobile learning design components resulted in a more effective learning experience.

For background research, we examine the state of the art for project development cycles, development environments, version control and document editors. We then use this research to decide on how we will approach the project and what software we will use.

2.1 Educational Design Theory & M-Learning

The purpose of this section is to define mobile learning, examine the fundamental design principles associated with mobile learning and to discuss the context and relevance of mobile learning in this project.

2.2 Defining M-Learning & Micro Learning

Mobile learning, also known as *m-learning*, has no distinct definition. This is likely due to the ambiguity in the term *"mobile"* but also because m-learning is a relatively new field in research. The term *mobile* refers to both the mobility of the user and the learning, which takes place using mobile technologies (Hashemi M. (2011)). Geddes (2004) focused on the aspect of learning using mobile devices, defining m-learning as "the acquisition of any knowledge and skill through mobile technology". Mobile Learning has also been referred to by König et al. (2015) in a context where the mobility of the learner is not important, instead defining m-learning as "the use of mobile devices and applications in the learning environment". Hashemi seems to agrees with definition by König et al. (2015) that mobile learning is the use of mobile handheld devices to facilitate and support teaching and learning, but he disagrees in the context of environment, emphasising the importance of the mobility of the location in which m learning can take place. According to Grant (2019) the most definitive method of defining mobile learning is to separate it into four categories, where each category is defined by specified characteristics. All of the four categories, used by Grant to define m-learning, share attributes that the aforementioned authors also used to define m-learning. These terms synonymous with m-learning that most mobile learning authors seem to agree on are web-based learning, using a mobile device to learn and mobility of the learning environments.

Another field of research, which is related to mobile learning, is mobile micro-learning. Mobile-micro learning is a form of mobile learning which emphasizes that the lessons of the m-learning system are between 30 seconds and 5 minutes. Systems like CodeAcademy, Udemy and Khan Academy are all examples of mico m-learning service (Jahnke et al. (2020))

2.3 Fundamentals of M-Learning Design

For this project, it is important to clarify the design principles of mobile learning using both research papers and empirical studies. These design principles can then be considered to be requirements of the cooking app design.

A major study conducted by Jahnke et al. (2020), focused on clearly defining *"the underlying design principles of academic literature and industry professionals that are inherent in existing mobile-microlearning systems"* with the intent of helping other researchers, teachers and designers to create more "effective and meaningful learning experiences" on mobile devices. The study was conducted by performing an academic literature review of all papers concerning mobile learning from 2013 to 2018, alongside interviews with industry professionals to confirm what was found from academic and industry reports. The findings, which were the result of a triangulation of scores from sets of interviews, academic literature and industry reports found that the following eight principles could be used to describe the inherent design aspects of mobile learning systems

Design Principle	Description
Interactive Micro-Content	Engage users by requiring **action** when using content
Chunked Courses	Duration of lessons short, between 30s and 90s. Bite sized lessons, micro lesson focused on single learning objective/topic. **Low complexity**.
Instructional Flow	Clear sequence of instructions for lesson, **diverse representation of content** eg. presentations, games, audio and videos.
System Design for Mobile Apps	Support on different systems, **push notifications**, tracking of learning progress eg. likes, views etc. **searchable** micro lessons part of larger picture available, easily updatable microlessons. Available for **offline** use. **Updatable lessons.**
Supporting Learner Needs	**Moment of Need:** if a user needs aid with a lesson it can be easily accessed through supplemental lessons. When user needs help there are existing micro lessons aligned with learner needs
Supportive Social Structures	Learner can get help through peers, community or teacher
Affordable Subscription Model	Non cost sensitive lessons

Figure 2.1: Design Principles of Mobile Learning, Each principle is listed with a short description, adapted from Jahnke, et al (2019)

Similarly, Grant (2019) attempted to define mobile learning through its most common characteristics. This was done by collecting and reviewing mobile learning research papers. The inherent design principles were then categorized by separating the repeating

fundamentals associated by scholars when describing mobile learning systems. Grant's principles are shown below:

Design Principle	Description
Learner is Mobile	User must use characteristics of **self-learning approach** lille self-regulation, self-directedness etc.
Device is Mobile	Device used to learn must be a **phone**, tablet etc.
Data Services are Persistent	Services like **Wifi**, **cellular** and **Bluetooth** should be connected
Content is Mobile	Learning should include **formal instructions**, trainings, **media**, data etc.
Tutor is Accessible	**Knowledgeable expert** like a tutor, coach etc. should be available
Physical Networked Cultures Impact Learning/Learner	Description of how physical, networked cultures and contexts affect user
Learner is Engaged	Description of how learner **engages** in learning environment

Figure 2.2: Design Principles of Mobile Learning (Grant M.M 2019)

By comparing the design principles, according to Jahnke et al. (2020) and Grant (2019), it can be seen that both studies define similar design principles for m-learning systems. It can also be noted that both sets of principles contain design aspects that focus on learner needs, mobile content design and supportive social structures.

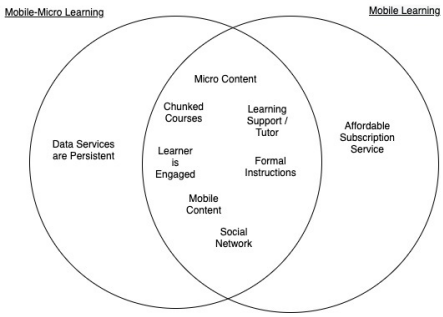


Figure 2.3: Intersection of Mobile Design Principles according to Grant and Jahnke

2.4 Context of M Learning in the Project

Most researchers seem to agree that two defining aspects of m-learning are that

- **Teaching** takes place on a **handheld/mobile device**
- **Lessons** can be taken in **any location** at anytime the learner requires

This goal of this project is to design and implement an app which can be used to learn how to cook recipes using a mobile device. It appears that the m-learning design principles would work as a good set of design requirements for our app. This is because the goals of the app align with the defining aspects of m-learning. Thus, according to Jahnke et al. (2020), by incorporating the mobile learning principles as requirements of our design, we will create a "more effective and meaningful learning experience". While predictably users will be using the recipe lessons in their kitchen, they could also be used in the shop, on the bus or practically anywhere. For example, the ingredients of a recipe is part of the lesson because user's must learn what ingredients are combined to form the recipe. It is likely that users will need to access the lesson while in the store purchasing the ingredients, thus the mobility of the system is an important requirement. In conclusion, our design will follow an m-learning approach due to the similarities in the goals of a typical m-learning app and our app. Next, we will examine usability and explain why it is important in the context of m-learning.

2.5 Usability and M-Learning

This section aims to explain why usability is important in the context of m-learning, what the definition of usability is and the defining aspects of usability.

2.6 Context of Usability in M-Learning

Usability is important in the context of mobile learning. In an analysis conducted on an educational mobile app, Huilcapi-Collantes et al. (2020) found that there was a major correlation between the user's perception of the app's usability and its impact on their learning experience. In a content analysis study of mobile apps, aiming to alter user food purchasing behaviors, it was found that a if an app required "significant effort" to use the system, users would not use the app again Flaherty et al. (2017). Flaherty goes on to state that mobile apps can be effective but "improvements in mobile app design are required to maximise their potential effectiveness". Clearly, usability is important in the

context of mobile learning, with the research indicating that there is a correlation between a user's perception of usability and the impact the mobile app has on learning. In order to maximise the usability of the system, It is important to understand the definition and attributes of mobile usability, which are then used as design requirements of the system, similar to our research approach to m-learning.

2.7 Definition of Mobile Usability

According to Shackel (2009), the usability of a system is defined by how effective, learnable, flexible and subjectively pleasing it is. Nielsen (1994) expanded on Shackel's definition, but instead focused on other attributes synonymous with usability; easy to learn, efficient to use, easy to remember, few errors and subjectively pleasing. Krug (2005) continued the definition of usability by focusing on what he referred to as the number one principle of usability in the context of websites: do not make the user think (also referred to as low cognitive load). More recently, Weichbroth (2020) argued that the term usability tended to be ambiguous and that there was a necessity for a systematic literature review, in order to create a more refined definition which could then be used in the context of mobile phone usability. The study found that while there was no formal definition for mobile usability, the vast majority of authors (88%) defined usability using the ISO 9241-11 definition "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" ISO (2018).

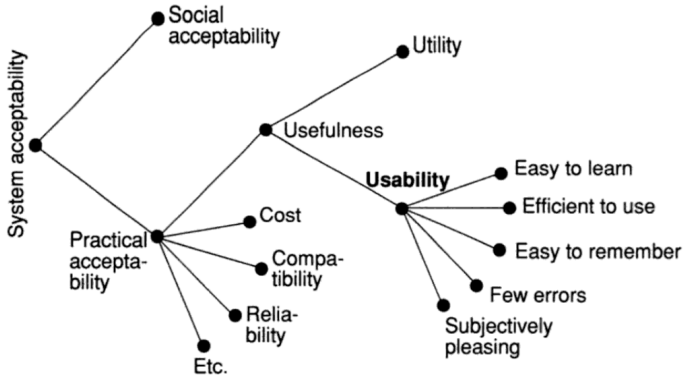


Figure 2.4: Nielson (1993) on Usability

2.8 Design Principles of Mobile Usability

As mentioned, the above authors defined usability using linguistic definitions but also focused on the attributes synonymous with the term as a more useful approach for defining usability. According to a study conducted by Weichbroth (2020), the following terms are the most synonymous with the definition of usability (in the context of mobile applications)

- Efficiency (70%)
- Satisfaction (66%)
- Effectiveness (58%)
- Learn-ability (45%)
- Memorability (23%)
- Cognitive Load (19%)

These six attributes can be said to be the *Design Principles of Mobile Usability*. Similar to our approach with the mobile learning design principles, these principles will be used as part of the requirements for the mobile system we are designing. By incorporating the principles of usability into our design requirements, our system will maximise the user's perception of usability and thus maximise the effectiveness of their learning experience Huilcapi-Collantes et al. (2020)

2.9 Evaluating M-Learning & Usability

The goal of this section was to examine what methods are used to evaluate m-learning and usability. As the goal of the project is to implement the design principles of mobile learning and usability, and then evaluate the results, it is necessary to research the most accepted forms of evaluation in the interest of evaluating our system appropriately but for also maximising the learning and usability of the system by using user-feedback from evaluations to improve our app.

2.9.1 Mobile Learning Evaluation

There are methods for evaluating mobile learning apps. One approach, taken by Vavoula (2005), was to evaluate the effectiveness of a mobile learning app by using diary recordings. Participants of the study would write their opinions on the learning process in a diary. Vavoula (2005) noted that the diary was good for gaining a better understanding of the contexts in which the app was used in. The diary evaluation method was also useful for understanding the user's opinions of the application. The study "provided some useful insights into the practice of mobile learning and how it compares to non-mobile learning". It was found that the mobile learning was more interactive, more communicative and more collaboration than traditional methods. Sarrab et al. (2016) took a different approach to evaluating m learning. Instead of using user testing, they created a *Quality Model* by combing previous quality models, technological advances and changes in mobile technologies to create a new model that would "evaluate and increase the successful development of new mobile learning products." The model captured both the abstract and generic technical aspects of m learning.

According to the paper, each technical aspect should be a part of the application. The evaluation of an app using the quality model is performed by awarding each technical quality aspect a score on a scale of 0-5, 5 meaning the app has incorporated the technical aspect perfectly and 0 meaning no implementation has been made.

Another questionnaire based approach, similar to Vavoula (2005) in that it utilises user-feedback, is the CE-CAM questionnaire proposed by Cota (2016) which aims to evaluate both the pedagogical and user interface usability of a system. Similar to the quality model, the questionnaire allows a user to respond within a range of 0-5, with 0 being strongly disagree and 5 being strongly agree. The CE-CAM questionnaire developed by Cota was used by Huilcapi-Collantes et al. (2020) to evaluate the effectiveness of an m learning app that promoted visual literacy. *"Constructs"* were created, where each construct was a

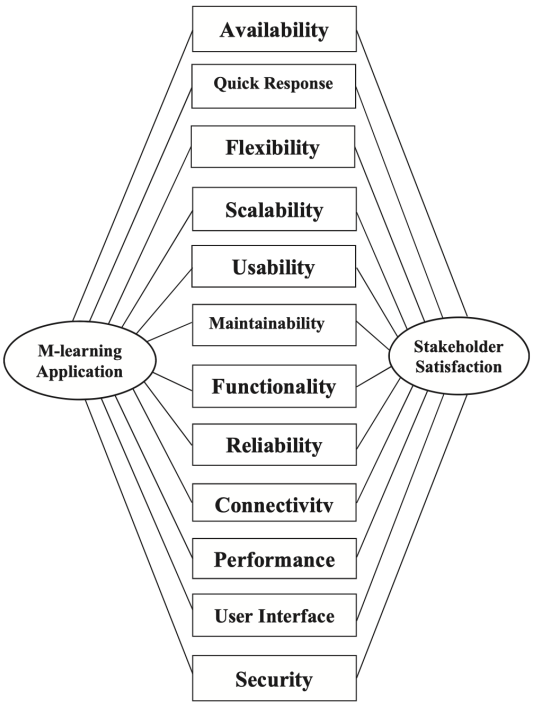


Fig. 1. Proposed technical quality model.

Figure 2.5: Technical Quality Model Sarrab et al. (2016)

number of items which focused on a specific aspect of the design. A judge would then award a score to each item within a construct ranging from 0 (worst) to 5 (best). Below is an example of the *"Educational Construct"* from the study.

Table 2: Educational Content evaluation results

	Educational Content	N	Mean	Sx
1	The content is organized in small modules or units.	5	5	0.00
2	Learning objectives are well defined at the beginning of a module or unit.	5	4.80	0.44
3	The required prior knowledge is made known, if necessary.	5	4.20	1.30
4	The explanation of the concepts is presented clearly and concisely.	5	4.40	0.89
5	The modules/units are organized according to the level of difficulty (from easy to difficult).	5	4.40	1.34
6	There are links to external resources related to the content and adapted for mobile devices.	5	3.60	1.67
			4.40	

Figure 2.6: Educational Construct Huilcapi-Collantes et al. (2020)

2.9.2 Usability Evaluation

There are a number of approaches to evaluating the usability of systems. One approach is the System Usability Scale. The System Usability Scale (also referred to as the SUS)

is a questionnaire that was developed by Brooke (1995) as a "quick and dirty" formal evaluation method, designed to quickly reflect the usability of a system. The SUS questionnaire is made up of ten statements. Each statement is scored on a scale between 1 and 5, 1 representing "Strongly Disagree" and 5 representing "Strongly Agree". Users are asked to complete the questionnaire after interacting with a system. The overall score of an SUS survey is calculated by summing the results from each question, it ranges from 0 to 100, with a higher score reflecting a higher level of usability. The SUS is an extremely reliable form of evaluation, a 10 year data study conducted by Bangor et al. (2008) found that "the SUS is a highly robust and versatile tool for usability professionals".

	The System Usability Scale Standard Version	Strongly Disagree					Strongly Agree
			1	2	3	4	5
1	I think that I would like to use this system frequently.		o	o	o	o	o
2	I found the system unnecessarily complex.		o	o	o	o	o
3	I thought the system was easy to use.		o	o	o	o	o
4	I think that I would need the support of a technical person to be able to use this system.		o	o	o	o	o
5	I found the various functions in this system were well integrated.		o	o	o	o	o
6	I thought there was too much inconsistency in this system.		o	o	o	o	o
7	I would imagine that most people would learn to use this system very quickly.		o	o	o	o	o
8	I found the system very awkward to use.		o	o	o	o	o
9	I felt very confident using the system.		o	o	o	o	o
10	I needed to learn a lot of things before I could get going with this system.		o	o	o	o	o

Figure 2.7: SUS Questionnaire Lewis and Sauro (2018)

As mentioned before Weichbroth (2020) defined usability as a compilation of the following attributes; efficiency, satisfaction, effectiveness, learn-ability, memorability and cognitive load. In the paper, Weicbroth notes several empirical evaluation methods that can be used to evaluate each attribute of usability. We created the table below to display each usability attribute along with a short definition and the measurements proposed by Weichbroth for evaluating the attribute.

Usability Attribute	Definition	Measurements
Efficiency	Completion of tasks with speed and accuracy	• Duration on screen • Time required to complete task • User error rate
Satisfaction	User's comfort and pleasure using system	• Survey with predefined statements
Effectiveness	Ability of user to complete a task in a given context	• Number of successfully completed tasks • Number of steps required to complete task • Number of taps required • Number of back buttons used
Learnability	Degree of ease a user can figure out a new system	• Number of attempts to solve task • Number of assists needed during a task • Number of errors
Memorability	Degree of ease a user can remember how to use an app and perform tasks	• Asking a user to perform a task after becoming more proficient with the app • Asking a user to perform a task after a prolonged period of time without use and comparing the results
Cognitive Load	Amount of cognitive effort exerted by users when performing tasks on the app	• Controlled observations • Surveying • Thinking aloud

Figure 2.8: Evaluating Usability Principles using Emperical Evaluation According to Weichbroth (2020)

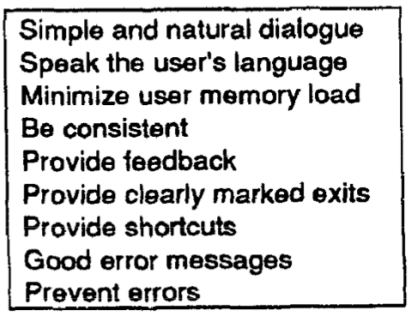
The measurements proposed by Weichbroth can be separated into two categories: *Task Observation* methods and *User Feedback* methods. Task Observation methods takes place during the evaluation of a system. Users are asked to complete certain tasks using the system and the subsequent data is recorded by the evaluation supervisor. The following are examples of task observation data

- Time Required to Complete Task
- Duration of Time on Screen
- Number and Types of Errors Per Task
- Number of Users Making Particular Error
- Number of Users Completing a Task Successfully

Weichbroth argues that these task observation methods are useful for measuring the efficiency, effectiveness, learn ability and cognitive load of a system.

User feedback methods are evaluation methods like surveys, questionnaires and interview, where users give direct, subjective, feedback on their opinion of the system. An example of a questionnaire for evaluating system usability is the SUS Questionnaire. *"Think-Aloud"* assessment is another form of user feedback, where users are asked to talk through their thought process while completing tasks. Weichbroth argues that Think-Aloud assessment is a useful for evaluating a user's cognitive load.

Another method of usability evaluation that differs to Weichbroth and Brooke's proposals is Heuristic Evaluation, proposed by Nielsen and Molich (1990). Heuristic evaluation is neither an empirical or formal evaluation technique. Instead, heuristic evaluation has a number of experts examine an interface alone, which ensures no bias, and each expert notes what they think is good and bad about the interface. Nielsen created the approach by defining nine principles of interface evaluation which could then be used for performing heuristic evaluation. The results of a test performed by Nielsen revealed that these nine principles reveal the majority of problems when reviewing a user interface design.



Experiment (short name)	*No. Evalu-ators*	*Total Known Usability Problems*	*Average Problems Found*
Teledata	37	52	51%
Mantel	77	30	38%
Savings	34	48	26%
Transport	34	34	20%

Figure 2.9: Nine Usability Metrics & Results of Four Experiments Nielsen and Molich (1990)

A typical heuristic evaluation has experts use the principles to identify issues with the system by filling in a report. The reports are then aggregated into a single report. Nielson found that by compiling the results of several evaluators into one, the single evaluation could better reflect the usability of the system. It was demonstrated that the aggregation of 3 to 5 heuristic evaluators could identify the majority of usability issues with a system. Nielson went on to reform these 9 heuristic principles, into 10 principles Nielsen (2005) which we have displayed below, each with a short description.

As mentioned before, Huilcapi-Collantes et al. (2020) used a CE-CAM questionnaire to evaluate the quality of their m learning app. This particular questionnaire is useful in terms of usability because it can evaluate both the pedagogical usability and the user interface usability, which is ideal for an m-learning app as both factors are important.

Heuristic Principle	Description
Visibility of System Status	System keeps users informed with what is going on
Match Between System and Real World	Use of normal language vs. system orientated terms
User Control and Freedom	Support of quick undo/redo emergency exits for users
Consistency and Standards	Similar words, situations or actions throughout system
Error Prevention	Careful design avoiding errors
Recognition Rather than Recall	Minimize user memory load by user interface
Flexibility and Efficiency of Use	Cater for both expert and novice user, tailor frequent actions
Aesthetic and Minimalist Design	Clear visibility and clean interface
Help Users recognize, diagnose and recover from errors	Error messages displayed in clear language
Help and Documentation	Easy access to help documentation

Figure 2.10: 10 Heuristic Principles of Design Nielsen (2005)

2.9.3 Conclusion of Evaluation Research

The research indicated that dictionary entries Vavoula (2005), Technical Quality Models Sarrab et al. (2016) and the CE-CAM Questionnaire were popular methods of evaluation for mobile learning systems. The SUS questionnaire (Brooke (1995), Lewis and Sauro (2018)), task observation methods, userfeedback methods Weichbroth (2020) and heurisitc evaluation (Nielsen and Molich (1990), Nielsen (2005)) were applicable methods for evaluating system usability. Each evaluation method focuses on specific aspects of the system it is testing, thus depending on the goal of the evaluation, a different approach should be taken. For example, if the goal of the evaluation is to gather quick feedback on user's perception of usability, the SUS is the best choice because it can quickly be performed and it widely accepted as a reliable questionnaire. If the goal of evaluation is to measure both the usability and the mobile learning effectiveness of the system, the CE-CAM Questionnaire is useful because unlike other methods it evaluates both of these concepts. The evaluation research from this section will be later used to test the system we develop, in order to gather user feedback, which will influence the system design, and also to evaluate the final implementation. Before then, we will use some of the evaluation methods researched in this section to evaluate the currently available applications.

2.10 Existing Mobile Applications

After researching what mobile learning is, the design principles of mobile learning, mobile usability and how both are evaluated, it is necessary to examine the currently available cooking apps to understand their issues. These issues could help us realize how our system should be designed. The following section discusses why it is necessary to evaluate existing mobile cooking apps, the evaluation approach we will take and the results of each system evaluation.

2.10.1 Issue with Existing Mobile Cooking Apps

To further examine the truth behind Flaherty et al. (2017) hypothesis, that the mobile apps available are currently ineffective due to their design flaws (section 2.2.1), and to also gather requirements data for our system, we will analyse the most popular cooking applications available on the app store and perform a design analysis and evaluation. After examining the fundamental design principles of mobile learning (section 2.2.1), we can use the review of evaluation methods (section 2.3), to test if these apps successfully incorporate mobile learning design principles. If the results showed that the apps are not employing mobile design principles, this would lend credence to Flaherty's hypothesis and also provide useful requirements data for our system.

2.10.2 Mobile Learning Evaluation Approach of Existing Apps

We downloaded the top three available cooking apps on the Apple App Store and will perform an analysis of each one. Using the research from the section 2.3, we will evaluate the mobile learning aspect of the apps using two methods; the Technical Quality Model Sarrab et al. (2016) and using the design principles of mobile learning according to Grant (2019).

The quality model will be used to evaluate each system because it is useful for revealing definitive issues with the apps and it will present a broad overview of the missing features of each system. Similar to the approach of heuristic evaluation, we have created a report, which is a table that lists the technical aspects of Sarrab's model, a brief description of each principle and a score to be filled out along with a comment section. The score is between 0 and 5, 5 meaning the technical aspect is implemented perfectly and 0 meaning no such implementation exists. The quality model was adjusted to only consist of aspects which are most relevant to the cooking mobile apps, for example maintainability is not a relevant design aspect for this project and so it was removed.

Technical Aspect	Description	Score	Examples / Comments
Availability	Accessibility associated with m learning		
Quick Response	Fast system		
Flexibility	Customization options		
Scalability	Accommodation of changes in system		
Usability	Ease of use		
Functionality	Good functionality of system based on user needs		
Reliability	reliable functionality of system		
Connectivity	Connection between user and device		
User Interface	Clear simple user interface		

Figure 2.11: Adapted Technical Quality Model Report used for evaluating the currently available cooking apps

2.10.3 BBC Good Food

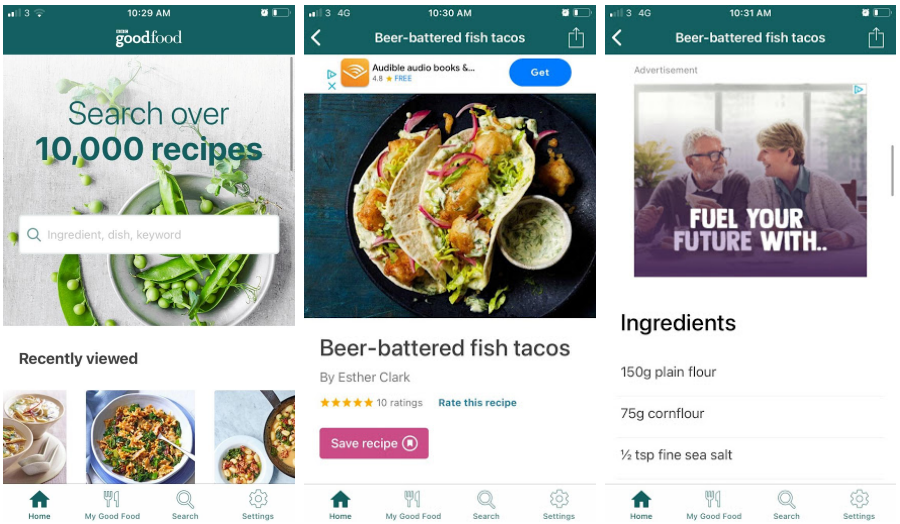


Figure 2.12: Home Page, Recipe Page and Ingredients Page

Good Food definitely has the best interface out of all the applications currently available on the app store. The pros of the system are the search bar for searching recipes, a useful save recipe button displayed with each recipe and the nutritional information available with each recipe. There are some major issues with the app which could negatively impact learning. Many advertisements are displayed, which are distracting, and at some times take up massive amounts of screen space. There is no shopping list functionality for saving recipe ingredients. No photos or multimedia is used to teach the cooking process or preparation process. Cooking instructions for each recipe are difficult to follow and appear in walls of text. Generally, the system has a confusing navigational

interface, for example the toolbar display search as a separate tab but the home page already displays the search.

Technical Aspect	Description	Score	Examples / Comments
Availability	Accessibility associated with m learning	2	• Content accessible on mobile • Content not available offline
Quick Response	Fast system	4	• Overall quick response
Flexibility	Customization options	0	• No customization of recipes or content available
Usability	Ease of use	3	• Ease to navigate system • Good search navigation • No home page displaying all recipes, must use search bar to discover recipe
Functionality	Good functionality of system based on user needs	2	• Ability to search and save recipes • Displays nutritional info of each recipe • Lessons do not provide proper media and interactive instructions • No shopping list functionality
Reliability	reliable functionality of system	4	• After use system only displayed one random error which was solved quickly
Connectivity	Connection between user and device	1	• No real interaction in lessons between user and device, enabling cooking mode just stops screen from closing down
User Interface	Clear simple user interface	3	• Relatively nice interface, clear and clean. • Toolbar is not entirely useful • Many advertisements displayed, take up large amount of screen space

Figure 2.13: BBC Good Food M-Learning Quality Model Report, shows an overall score of 19/40

2.10.4 Recipe World

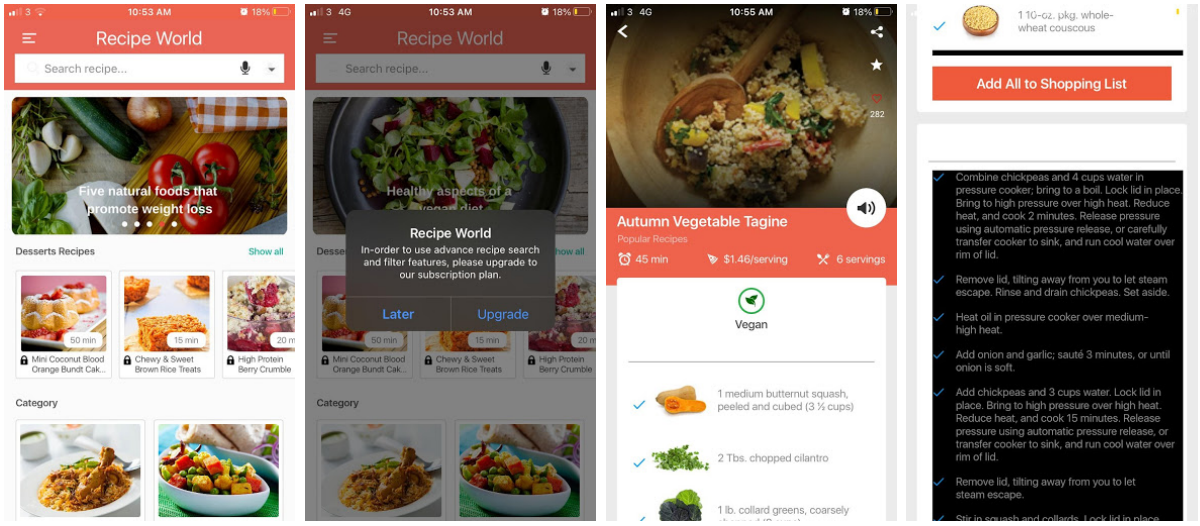


Figure 2.14: Home Page, Subscription Page, Recipe Page, Method Page

Recipe World has some major fundamental flaws in its design. Although each recipe lesson has a predicted pricing per serving, there is no indication of how pricing is measured

for recipes. The steps of each recipe in the method section has a terribly layout and is difficult to read. Many features of app are blocked behind a paywall, which is an unrealistic subscription service for students/lower income users. Without a subscription users cannot use the search bar to navigate recipes and 90% of recipes become unavailable. Again, there is no use of multimedia to aid recipe instruction but each ingredient has an image beside it, which is a nice feature.

Technical Aspect	**Description**	**Score**	**Examples / Comments**
Availability	Accessibility associated with m learning	2	• Content accessible on mobile • Content not available offline
Quick Response	Fast system	3	• Overall average response
Flexibility	Customization options	0	• No customization of lessons or content available
Usability	Ease of use	2	• Difficult to navigate system • bad search navigation • Confusing interface with subscription lessons mixed with free lessons • Nearly impossible to read instructions for lessons due to design
Functionality	Good functionality of system based on user needs	2	• Ability to search • Displays cost info of each recipe • Lessons do not provide proper media and interactive instructions
Reliability	reliable functionality of system	2	• No clear indication if recipe is free or not • App crashed for no reason
Connectivity	Connection between user and device	1	• No real interaction in lessons between user and device
User Interface	Clear simple user interface	2	• Confusing, cluttered interface • Lesson interface design is not good
Overall Result	Individual scores / total	14/40	

Figure 2.15: Recipe World M-Learning Quality Model Report with an overall score of 14/40

2.10.5 Yummly

Yummly is an app that does a lot of things right and a lot of things wrong. Some major issues with Yummly are the method for recipes. Yummly attempts to display each recipe instruction with an image/video, which is a great idea, but for most recipes it is unavailable, rendering it useless. Yummly attempts to allow shopping online for recipes but does not work in Irish locations. Some recipes actually redirect you to a website where the recipe is, off the app, which not useful for people who have no internet and is fundamentally a bad design. Finally, many advertisements take up screen space and also pop up on user's screen. One positive aspect of Yummly's system design are the alert methods to help people with cooking times, which could be improved by changing it to be automatically set for each recipe instead of having the user do it. The shopping list is well designed and executed, allowing for arrangement by category or by recipe.

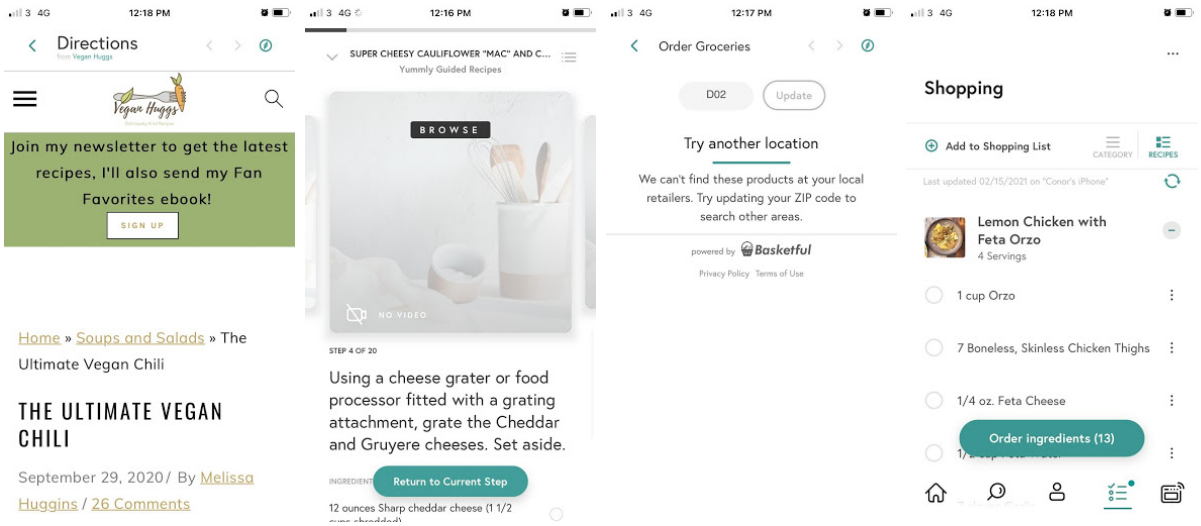


Figure 2.16: Recipe Page, Order Page, Method Page, Shopping List

Technical Aspect	Description	Score	Examples / Comments
Availability	Accessibility associated with m learning	2	• Content accessible on mobile • Content not available offline
Quick Response	Fast system	3	• Overall average response
Flexibility	Customization options	0	• No customization of lessons or content available
Usability	Ease of use	2	• Difficult to navigate system • bad search navigation • Confusing interface with subscription lessons mixed with free lessons • Nearly impossible to read instructions for lessons due to design
Functionality	Good functionality of system based on user needs	3	• Ability to search • Displays cost info of each recipe • Lessons try but do not provide proper media and interactive instructions
Reliability	reliable functionality of system	2	• No clear indication if recipe is free or not • App crashed for no reason
Connectivity	Connection between user and device	3	• Each lesson has a step by step mode which interacts with user
User Interface	Clear simple user interface	3	• Confusing, cluttered interface • Lesson interface design is close to okay, but too many pitfalls result in a half baked design
Overall Result	Individual scores / total	18/40	

Figure 2.17: Yummly World M-Learning Quality Model Report with an overall score of 18/40

2.10.6 Mobile Learning Comparison with Available Apps

The final evaluation of the available apps was a simple graph which displayed what mobile design principles, according to Grant (2019)), the available apps had successfully implemented in their respective design. The graph contains another app named 'Tasty', the analysis of this app was not included in this essay because we decided the other three individual reports sufficed. The graph simply illuminates that most apps only follow the m-learning principles that are inherent with mobile apps, such as Mobile Content and Device is Mobile. While one app was free, the other three required subscriptions to access

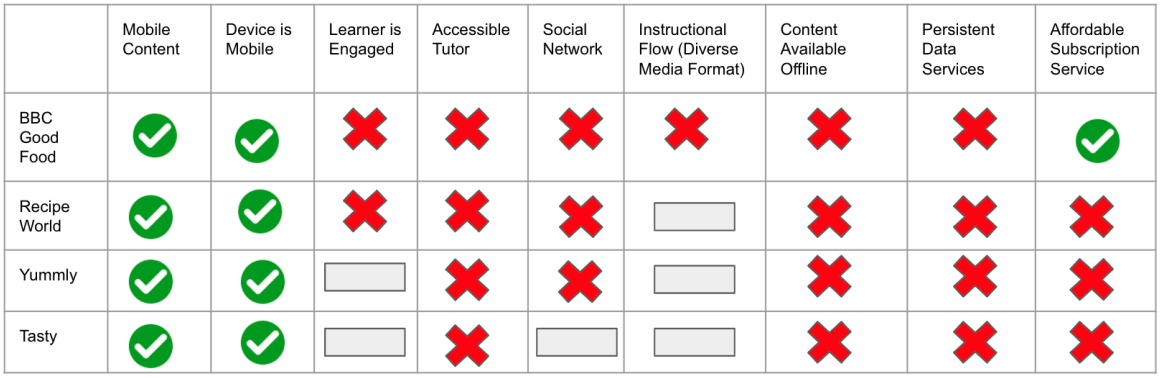


Figure 2.18: Comparison of Available Apps and Mobile/Micro Mobile Design Principles

all of the lesson content. The grey rectangles represent an attempt at implementation, for example both the Yummly and Tasty designs had each instructions in a lesson aided using a visual media. Unfortunately, it was found that many recipes did not have media accompanying them and instead were filled with empty photos or placeholders [Figure 18.2, Figure 16.2]. It was also the case that no apps offered lessons offline.

2.10.7 Conclusion of Evaluation

Our evaluation demonstrates that the mobile apps currently available do not incorporate mobile design principles into their system. Our graph (Figure 20) shows that the four apps fail at implementing every design principle, except two principles which are inherent to applications mobile devices. Each app performed poorly in the Quality Model Reports, where no app achieved a score greater than 50%, with the highest score being 19/40 and the lowest being 14/40. Our analysis also revealed major flaws within the system design of each app such as missing media, external links to other websites, incomprehensible lesson instructions and cluttered user interfaces. Our evaluation results grant Flaherty et al. (2017) hypothesis some validity, due to the poor scores of each app which demonstrates their limited learning design. The evaluations also provided us with design requirements that we can incorporate into our system, which we will discuss in further detail below. The quality model reports and the comparison table will also be used to compare our final implementation of our cooking app with the currently available apps.

2.11 Technologies Used

An important aspect of the research was to research the technologies that could be used to build the system. We would make decisions about the development environment, repository system, database, development methodology and document editors based on our research in this section. A brief review was done on the state of the art regarding each of these technologies.

2.11.1 Development Environment

The three main development environments that are available for developing a mobile application are Android Studio, Xcode and React Native.

Android Studio
Android Studio is the official IDE for the android operating system. Android apps are currently programmed in Kotlin. Android is the most popular operating system for mobile devices and android studio is a unified environment, meaning the program will run on all android devices.

Xcode
Xcode is the offical IDE for iOS devices. The main programming language is SwiftUI. Xcode offers unique design tools to construct and edit user interfaces such as the built in visual previews, which increases the efficiency of editing interfaces. Refactoring user interfaces is quick in SwiftUI, by breaking down one page (a view) into multiple views. Xcode is also the native development environment for all iOS devices, meaning programs run very efficiently because they have been developed for these particular devices, this is known as "native performance" and has been adopted by design teams in large development companies like Instagram, Facebook and Twitter.

React Native
React native is an IDE which develops apps for both android and ios. Programmed in JavaScript, react native provides a system which allows you to code in one language and ide for deployment on cross platform mobile devices.

2.11.2 Database

AWS, Firebase and MongoDB all seem like good options for the system database and we have previous experience using each application. All three systems are online and

would require an internet connection to retrieve stored information. All three offer similar services, AWS offers fifteen different database engines and is developed by Amazon. Firebase, developed by Google, is used for mobile app backend infrastructure. MongoDB is a document database, meaning it mainly stores data in JSON like documents. A final option for the database is a locally stored database. On a mobile, this could be achieved by using a locally stored JSON file or storing the database directly on the user's phone, within their documents. This would mean that the data could be accessed offline and customized by the user without effecting other user's data.

2.11.3 Online Repositories & Document Editors

We decided to use Github has the primary repository for the project's code. The reason for choosing Github was that we had previous experience using the software. The main reason for using a repository was to copies of the program we were developing, in order to avoid accidentally deleting our work, but also to be able to return to previous prototypes and use them for testing. Google drive was also used to maintain a collection of Mentor Notes, Meeting Notes, Project Diary, Research Papers, Brainstorming, Documents, Prototypes and other useful documents. Google drive was chosen over other services like Dropbox because we had experience with drive, it was linked with our college account and it provides great functionality through its document, presentation and spreadsheets. Google drive also offers shared folders, which was useful for sharing the documents with our supervisor, so at any time they could view the documents easily. Overleaf is cloud-based text editor which uses LaTeX, a software system for document preparation. Overleaf was chosen to write the report because it's cloud based system ensured the report would not accidentally be deleted, the document preparation offered by LaTeX results in a great looking report document and we had experience using it for previous projects. Draw.io, a flowchart solutions software application, which was chosen to be used for all graph drawings in the project.

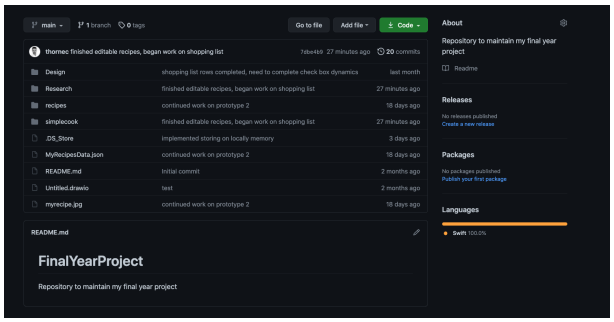


Figure 2.19: Project Repo

2.12 Project Ethics

This section discusses the ethical considerations that we will undertake as part of the project.

2.12.1 Ethics Canvas

We began the process of identifying ethical areas of significance by filling in an ethics canvas, using the template from the Final Year Project module.

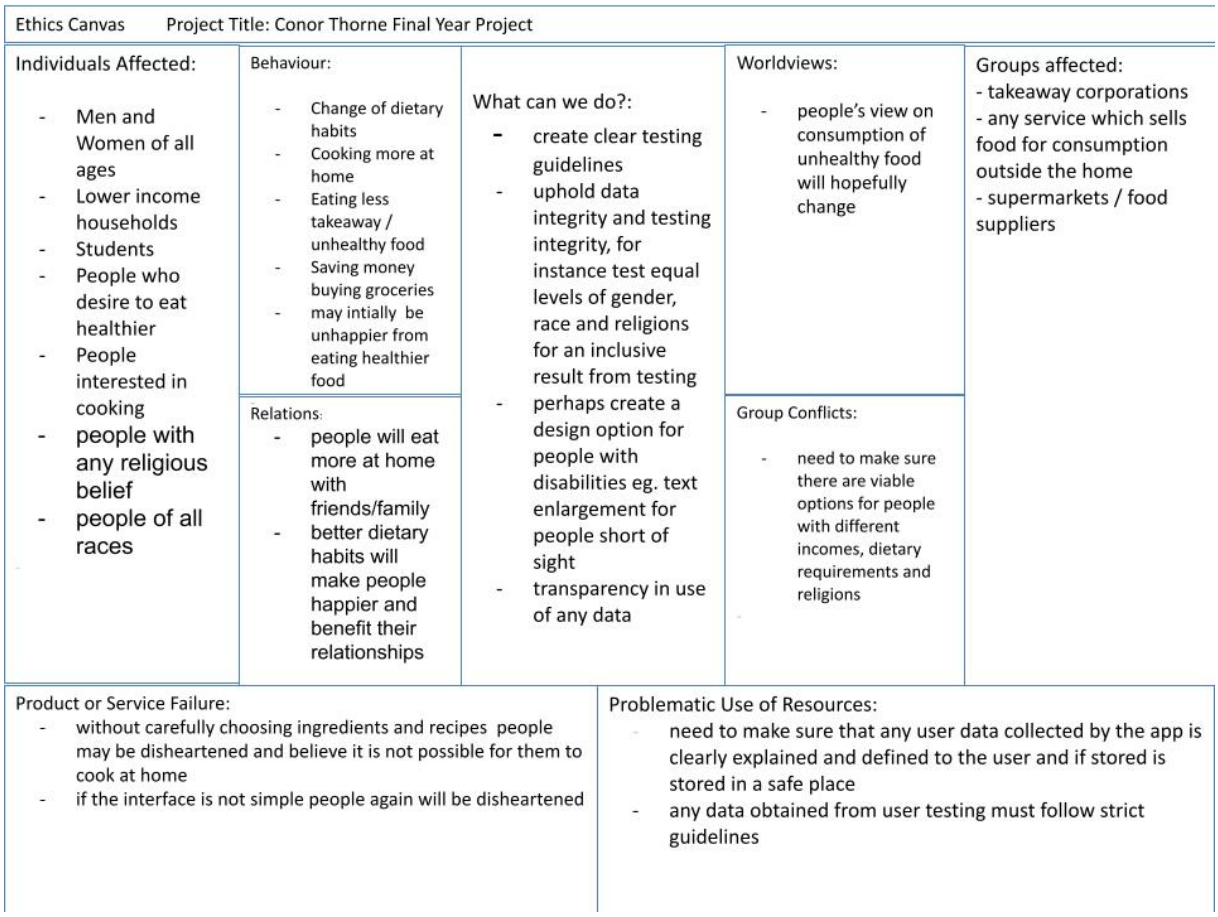


Figure 2.20: The Ethics Canvas helped identify the individuals & groups who would be effected by the system, possible conflicts, behavioral changes, problems and what we could do to ensure an ethical approach throughout the project

2.12.2 Ethics Application

Our project supervisor, very early on, indicated that the testing and evaluation stage would require a research ethics application to be approved by the Trinity Computer

Science and Statistics Research Ethics Committee. The SCCS Ethical application consisted of

- Participant's Information Sheet
- Participant's Consent Form
- Research Project Proposal (who are the intended participants, how they will be recruited, if they are under the age of 18)
- Intended Questionnaire and Interview Protocols

The application was filled out and approved by our supervisor, It was then submitted to the committee, who responded quickly commenting on changes to the application that were required. The second draft was submitted, including the changes, and the ethics application was approved by the REC (Research Ethics Committee) on Thursday 18th March. Our project then had permisson to run testing on users and it also gave us confirmation that all of the correct ethical considerations had been correctly analysed.

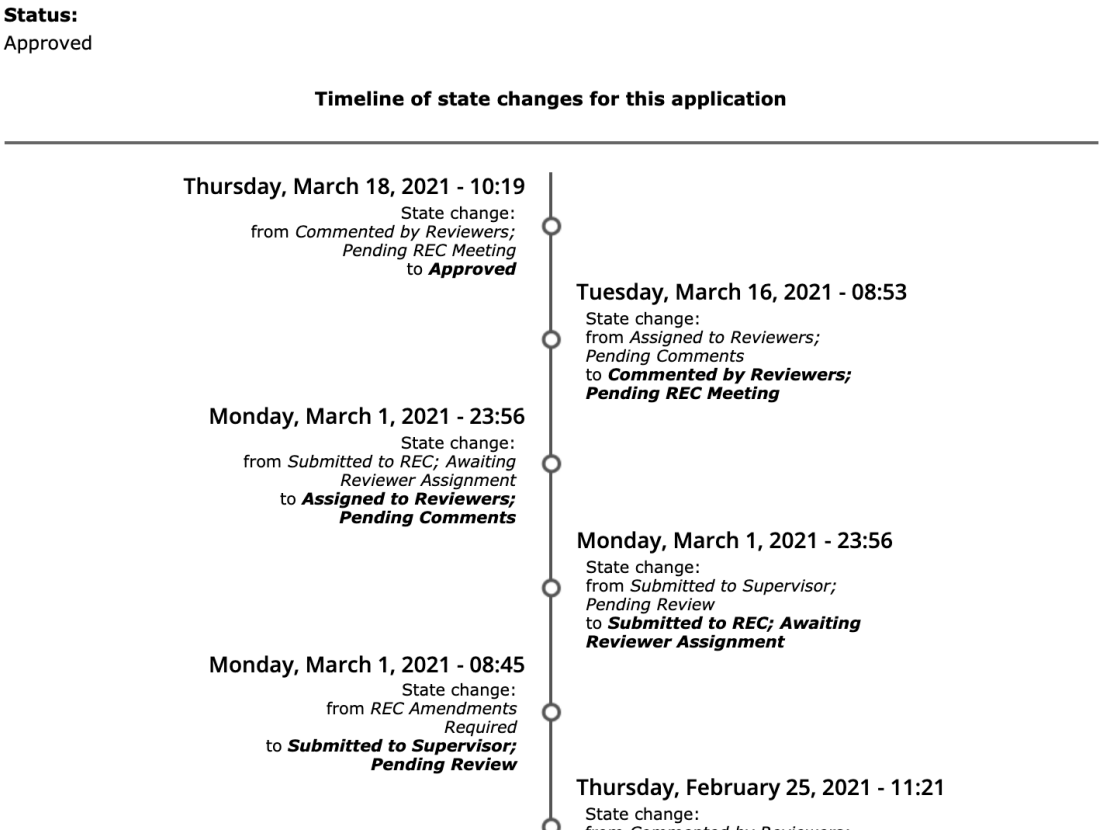


Figure 2.21: Timeline of Research Ethics Application

2.12.3 Data Ethics

The final ethics consideration concerned the data displayed in the system. In order to perform testing, some recipes would need to be available in the app, which would consist of images and text. The focus of the project is on the technical research and implementation and thus the actual recipe instructions and visual imagery are only important for receiving user feedback. It is not the purpose or goal of this project to create recipes or recipe images but both were needed to perform testing. The images that were used were either owned by the author or taken from a copyright free web page called Unsplash.com, which provides stock photography. By using our own images or copyright free images we will avoid any ethical data issues and will also comply with fair use laws.

2.13 Conclusions

From our research, we compiled information on the definitions and the fundamental design aspects of mobile learning and usability and the different evaluation methods used for assessing usability and mobile learning. We also compiled information on the existing mobile applications available for cooking, performing multiple forms of mobile learning evaluation on each system. The results from our evaluations demonstrate that the designs of these apps are not following, what the research considers, the fundamental design principles of learning technology and mobile usability. It is our goal to use the background research reviewed in this chapter to influence the design and evaluation of the system we will implement. The mobile learning and usability reviews provided useful requirements for the design of the system. The review of different evaluation methods will help with testing our system, which will contribute to the design. Examining the available mobile applications provided additional design requirements for our system and useful reports which can later be used to compare against our system. A brief review of relevant technologies and their benefits was also performed, which will be used later to decide what systems and approach to take during the project. The next section discusses the design process of the system, which was heavily influenced by the research from this chapter.

Chapter 3

Design Process

In this chapter, we will explain the methodology behind the design process of our project, the requirements of the system, how they were gathered and the prototypes which led to the final design. Our goal was to design and implement a mobile learning app and the design process explains how we developed the technical aspects required by such a system.

3.1 Design Methodology

We chose to follow the *Iterative Design Approach* as our project design methodology, which is a form of agile development. The iterative approach focuses on implemented features in increments. Each increment concentrates on designing specific functionality, which is then implemented into a prototype. The prototype is then tested with users and the design is refined using the user-feedback. We chose the iterative design approach for a number of reasons. One benefit of the iterative approach is that it will allow us to refine our design using user-feedback. The user can be considered to be part of the design team. This is ideal for our project because we aim to maximise our system's usability and learning effectiveness. Thus by being able to incorporate user feedback we will be able to improve each iteration of our design. Also, because some of the requirements of the system are derived from research, it is important to utilise user feedback, in order to refine these design elements from research into more practical implementations. Another benefit of the iterative approach is that it allows for flexibility and adaptation of design. Our requirements for the project may change or evolve and the iterative approach will allow us to adjust the system design accordingly. Finally, the iterative approach will allow us to quickly design and implement the project, avoiding time spent on documentation. The scope of this project is short, thus a requirement of the design approach is to make

the most effective use of our time.

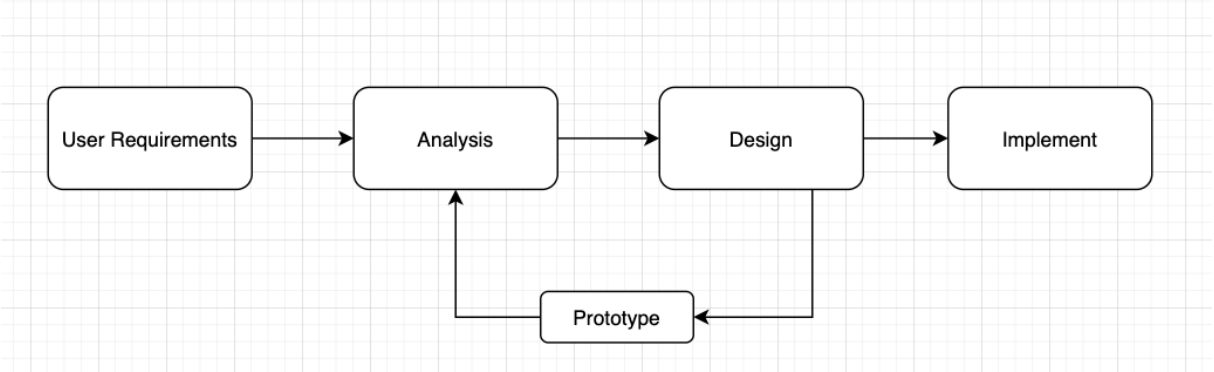


Figure 3.1: Iterative Design Process, which follows a a requirement, analysis, design, prototype cycle

The above diagram illustrates the design process. User requirements are defined, the requirements are analysed and incorporated into a design. The design is implemented using a prototype, which is analysed using testing. The requirements are then refined and a new design is implemented. This cycle repeats until a final implementation is complete. The iterative design process begins with documenting the systems requirements.

3.2 Requirements Analysis

The requirements analysis phase is the first step of the design process. Our aim is to understand the users, their goals, tasks and the contexts in which all of the aforementioned will take place. Establishing requirements is a difficult task. As a designer, we must differentiate what a user wants versus what a user needs. Users are the people who will be using the system, in our case, a cooking application on a mobile device. Individual users posses their own perspectives on what they require of a system and it is our task to weight the benefits of each requirement, refine them into a set of stable requirements for the app to fulfill. Because of this, we will perform requirements research using multiple data gathering methods.

As mentioned, different data gathering techniques will be performed to define the requirements of the system. The following section outlines the data gathering techniques we will use to understand the system and user requirements. The information we will gather will then be used to create prototypes. These prototypes will be used for testing, which is the final data gathering technique used to aid the design of the system.

3.2.1 M Learning & Usability Requirements

The first data gathering technique was performed as part of the background research, specifically the research of mobile learning and mobile usability. It is our project's goal to implement the design principles of mobile learning (section 2.1.2) and the design principles of mobile usability (section 2.2.3) into our system. Thus, both sets of design principles should be included as part of our system's requirements. We will use both the design principles of mobile learning Grant (2019) and the design principles of mobile micro learning Jahnke et al. (2020) to form the initial design requirements of the system, along with the design principles of mobile usability according to Weichbroth (2020). The following requirements gathering method performed was hierarchical task analysis.

3.2.2 Hierarchical Task Analysis

Hierarchical task analysis involves breaking down the tasks and goals of a user of the system. By understanding a user's goals we can realize how the application should support them. Task analysis involves brainstorming; what are user's goals? how are they trying to achieve these goals? what knowledge is required to complete their task? and what is the hierarchy of actions taken in order to reach their goal?

From our task analysis, we found that the goal of a user in the system is to cook a recipe for themselves. It is assumed that users wish to achieve this because they wish to live healthier lives, eat healthier home cooked meals or save money. The way in which user will go about cooking a recipe is quite simple. Users will use the application to find a recipe they want to cook and they will add the ingredients of the recipe to their shopping list. Then, the user will then go to their supermarket and use the shopping list to retrieve the correct ingredients. The user will then bring the ingredients to where they plan on cooking the meal, most likely in a kitchen, and use the app to follow the lesson instructions and visual aids of the recipe to prepare and cook the meal.

The diagram (figure 3.2) shows a hierarchical task breakdown of each task a user will complete in order to reach their goal of cooking a recipe. The goal is defined as "eat(ing) a home cooked meal" and each step is a sub task of this goal. Two "plans" are written. Because it is not always necessary for users to complete all tasks to reach their goal, for example, if a user already has the ingredients they can skip straight to cooking the meal, demonstrated in plan 0. The hierarchical task analysis will help us to put the user's goals and tasks at the forefront of the design requirements, which will result in a system design orientated towards user needs.

Hierarchical Task Analysis

- Goal is to eat a home cooked meal
- 1. Find Recipe
  - 1.1 Open application
  - 1.2 Search for a recipe
    - 1.2.1 use search bar
    - 1.2.2 look through saved recipes
    - 1.2.3 navigate through recipe list
- 2. Get ingredients
  - 2.1 select ingredients needed and add to shopping list
  - 2.2 go to shop and use shopping list to get required ingredients
- 3. Follow recipe instructions
  - 3.1 enter cooking mode
  - 3.2 Follow instruction on page using visual aid and written instructions
  - 3.3 click on screen to enter next stage
  - 3.4 repeat steps 3.2 and 3.3 until all instructions of recipe have been executed

Plan 0: do 1-3 when ingredients are already in possession
Plan 1: do 1 - 1.2.2 - 2 - 3 when recipe is saved and user already knows what they want to cook

Figure 3.2: Hierarchical Task Analysis Example

3.2.3 UML Use Case Diagrams

Use case diagrams are a form of UML (Unified Modeling Language) which is used to illustrate how a user will interact with a system. Use case diagrams are useful for summarizing scenarios, goals and the scope of systems. Use Cases are ideally used at the beginning of the requirements gathering process, mainly because they only outline the applications of the system and do not go into much detail on functionality.

The system use case diagram for our app only has one actor, the user, who has a goal of cooking a recipe. There are three use cases, each with the following goals;

- **Find Recipe** - Users can use the search bar, navigate a list or access recipes through a saved section
- **Get Ingredients** - In order to cook a recipe, users must select the ingredients they need, add the ingredients to a shopping list and use the shopping list to remember what they need to purchase
- **Follow Recipe** - Once the user has chosen a recipe and gotten the ingredients, they must read the instructions of the recipe, enter cooking mode and follow each instruction.

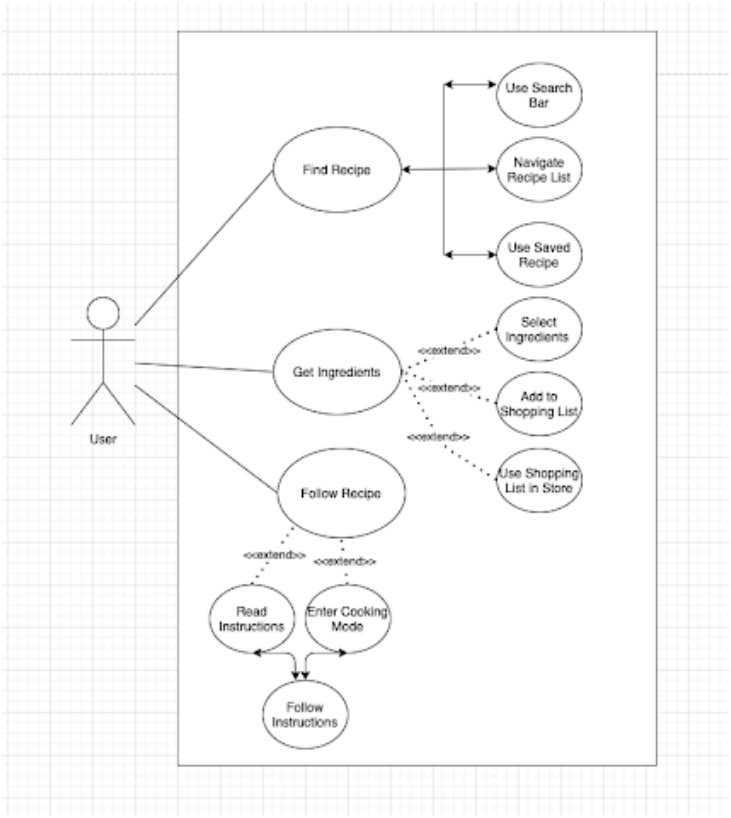


Figure 3.3: System Use Case Diagram

The use case diagram is useful for brainstorming the initial requirements of the system, the user's general goals and how a user would interact with the system.

3.2.4 User Stories

A user story is a requirement gathering technique which focuses on the perspective of the system from a user. It is usually informal and contains, what some may say, useless information to paint a clear picture on who the user is. It is a part of agile development. The user stories will provide context for us to use to remain focused on the user aspect of the system. An important aspect of iterative approach will be creating a design that puts the user first. User stories will help us remain focuses on the user experience and will aid us in understanding the different contexts in which the app may be used.

> Rob is a student from Cork and studies at Cork University. It's Rob's first time living away from home and he can no longer rely on his mother's cooking. Unfortunately, rob is a terrible cook and has no idea how to cook even the simplest dish. Determined to avoid taking away food and looking to save money, Rob has downloaded the simple cook application from the app store, recommended by a friend. Rob still hasn't got his wifi in the apartment working yet, but luckily he downloaded the app using the bus wifi on the way home from University. Rob searches through the app for a meal that he likes and is relatively easy to cook. Rob spots a nice chicken meal and saves the recipe to his favourites. Rob scans through the instructions and reckons he'll be able to cook it, Rob already has some olive oil and garlic so he needs to pick up the chicken fillets and chilies from the shop. He adds the ingredients to his shopping list and goes to the shop. At the shop, rob opens the shopping list and ticks off the chicken and the chillies after he gets them from the shelf. Rob goes home, opens the cooking mode of the app and begins to follow the instructions.

Figure 3.4: User Story

The user story above illustrates the story of a student who uses the app to save money and eat healthier. While the user story goes into no detail about how the system function, eg. click on search bar, many requirements can be derived from a user story like Robs

- Users can have limited experience cooking
- Users need to be able to use app without wifi
- Users should be able to see the cost and difficulty to cook a meal from search page
- Users should be able to save a meal to a special section for referal to later
- Users need a shopping list to make sure they select the correct ingredients
- Users need a hands free mode to follow lesson instructions while cooking

The user stories we wrote were of great help to understand the systems' functional and non functional requirements from the perspective of different users, not designers. Beyond defining requirements, user stories will be constantly referred back to during the design process, mostly to refresh and refocus the project's scope.

3.2.5 Online Surveys

An Online Survey is a useful method for gaining an understanding of people's opinions of systems they currently use. A survey consists of asking a question, recording answers and drawing statistics and requirements from the answers. At first, we aimed to do in person surveys to gather real secure data but because of the current pandemic, this was not possible. Instead we came up with the idea of finding online forums where people were surveyed on what they disliked about recipes websites and apps. We found two such discussion threads on reddit, a forum based social network. We took these two surveys and analysed people's responses to the question. The following were the most persistent and popular complaints of recipes apps/sites

- Too Much information (Background Stories, Blog like pages instead of educational)
- Advertisements
- Subscriptions
- Bad Formatting (eg. click through several pages to reach recipe)
- Expensive Recipes
- No videos / visual guides with recipes
- photos of cooking recipe scattered around the page or too many photos
- no nutritional information
- no pricing or availability of recipes
- no search bar for recipes/ingredients

From these online surveys, we were able to gather data from a far greater number of people's opinions (both threads had over 500 responses). We were also able to get a better feeling of the requirements of users, by noting that these features that were prominent in the currently available apps should be avoided. It was interesting to note that a lot of the complaints of users surveyed eclipsed what we had found from our research such as the lack of visual guides and subscriptions impeding the learning process.

3.3 Prototypes

After performing the data gathering techniques discussed above, we began to implement these requirements into a system design, using prototyping.

Prototypes are preliminary versions of a design and are an integral part of the design process, because they are cheap, fast and can provide feedback from users very quickly. Prototypes can vary in fidelity, meaning percentage of features covered and degree of functionality. Prototypes are useful for demoing screen layouts and task designs. In this project, prototypes will be used as part of the requirements analysis. In our case, the design of the system is the most important part of the project. After gathering requirements for the system, we created low fidelity, hand drawn prototypes. These prototypes will be analysed and then new, higher fidelity prototypes will be created using the feedback and the additional requirements gathered from the analysis. This process will be continued, with a prototype being created, tested and analysed through testing for the entire duration of the project.

3.3.1 Low Fidelity Prototypes

The conceptual design for the app was realized using a hand drawn prototype. Paper prototypes are useful because they are fast to produce and they are cheap, mainly because there is zero code needed to test the interface on users.

The primary objectives of a hand drawn prototype is to test the users' understanding of the system. User's understanding can be measured by evaluating the navigation/flow of the app's layout ie. *"were users able to navigate the app and recognize buttons and labels and their use?".* Testing using paper prototypes is simple but it requires some imagination. Users will be asked by an evaluator to perform tasks, pretending the paper interface is real. If a user taps a button or swipes, the evaluator will tell them what has changed, for example if a user presses a recipe card at the home page, the evaluator would tell them that they were now viewing the recipe details page.

The hand drawn prototypes will also help us visualize an initial design and aesthetic of the system. These low fidelity prototypes above will be used as the blueprints for developing high fidelity prototypes in Xcode.

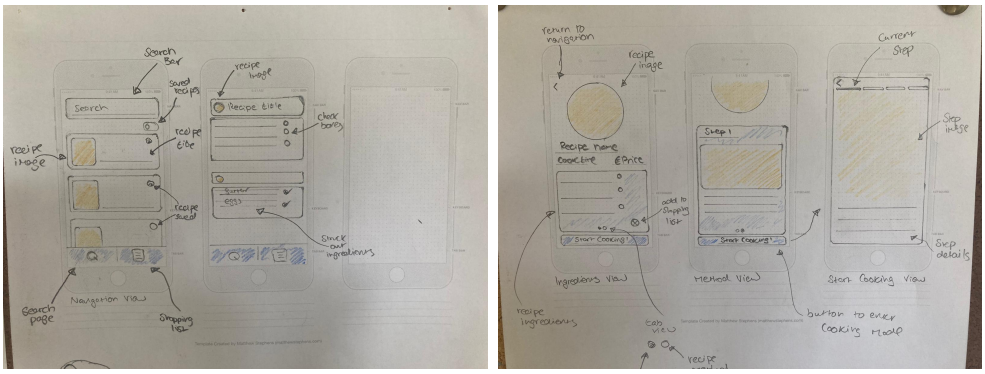


Figure 3.5: Home Page, Recipe Page and Shopping List Prototypes

3.3.2 High Fidelity Prototypes

The higher fidelity prototypes will be implemented in Xcode, these prototypes will have a higher degree of functionality but still have a lot of placeholder data or non-functional buttons etc. The higher fidelity prototypes are similar in utility to the drawn prototypes, they are useful for testing and gathering feedback from users. The higher fidelity prototypes will take longer to implement but the users in testing will feel more comfortable using them because they resembled something closer to a real app, as opposed to a drawing on paper.

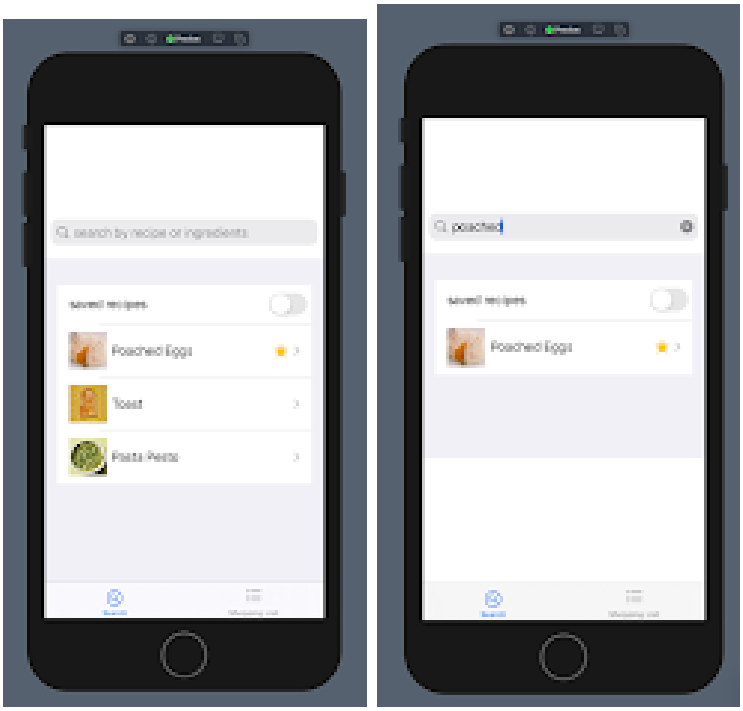
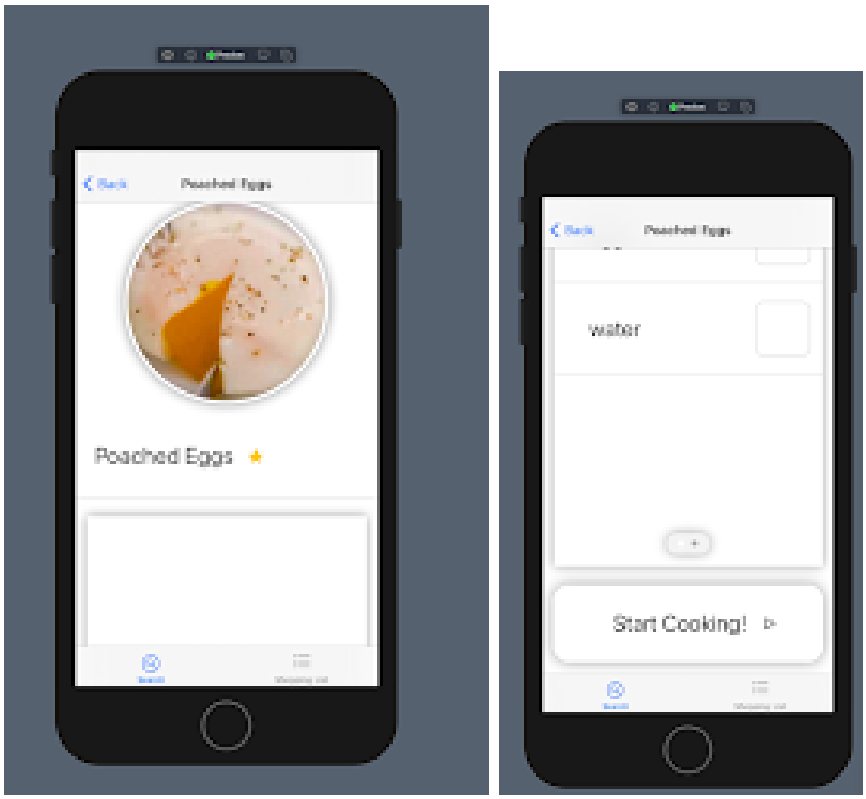


Figure 3.6: Home Page, Recipe Page and Shopping List Prototypes

Our supervisor encouraged us to develop a digital prototype as soon as possible for



a few reasons. Mainly because they would help brainstorm ideas and increase creativity. The low fidelity prototypes allowed us to be creative and focus on the bigger picture of the app's aesthetic, while incorporating the system and user requirements. While the higher fidelity prototypes helped us show the system to users and incorporate their feedback into the design. Overall, prototypes were fundamental to the success of this project. Initially, a conceptual design prototype got the ball rolling on design, each iteration of prototype resulted in more data being gathered, a better understanding of what worked and what didn't and a clearer sense of how users interpreted and felt about the system. The prototypes eventually resulted in the final design of the system.

3.4 Final Design

This section aims to demonstrate the final design of the system. Some explanation of the design choices are mentioned in this section, but a more detailed breakdown of the mobile learning & usability design implementations is discussed in the next section, Design Analysis.

3.4.1 Home Page & Recipe Details

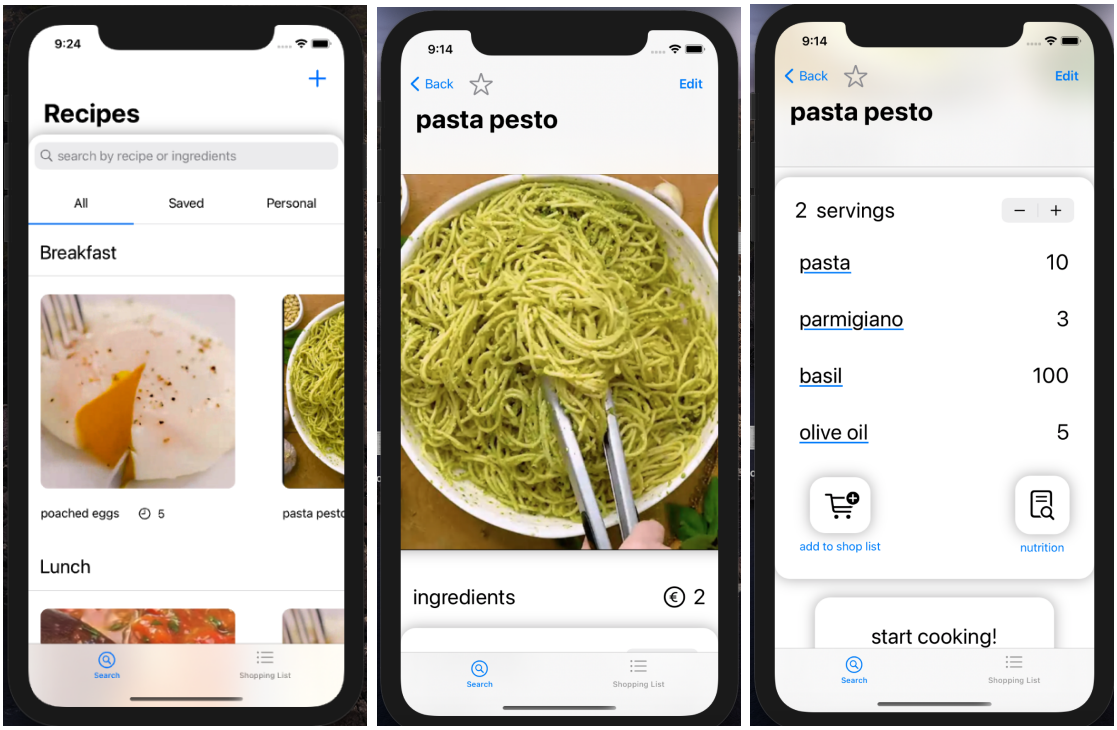


Figure 3.7: Home Page & Recipe Details View

The homepage displays all the recipe lessons in the application. A user can filter through the recipes by using the search bar to show recipes by entering the lesson name or an ingredient that is in the recipe. Users can further filter lessons by using the sliding tab bar to select all recipes, saved recipes or personal recipes, which are recipes that are added by the user. Lessons can be added using the blue plus button in the top right corner of the screen. Upon clicking into a recipe lesson, the lesson details are displayed. Users can click the star to favourite/save a recipe and click the edit button to change the recipe details, this aspect is discussed in further detail below. The ingredients of the recipe are displayed and the serving sizes of each ingredient, which can be altered by clicking the stepper. Each ingredient can be clicked, which will display the dictionary definition of

the ingredient. A user can click the shopping cart button to add the ingredients to their shopping list and they can click the nutrition button to view the nutritional information about the dish. A start cooking button is displayed which upon clicking begins the lesson.

3.4.2 Dictionary, Nutritional Page & Shopping List

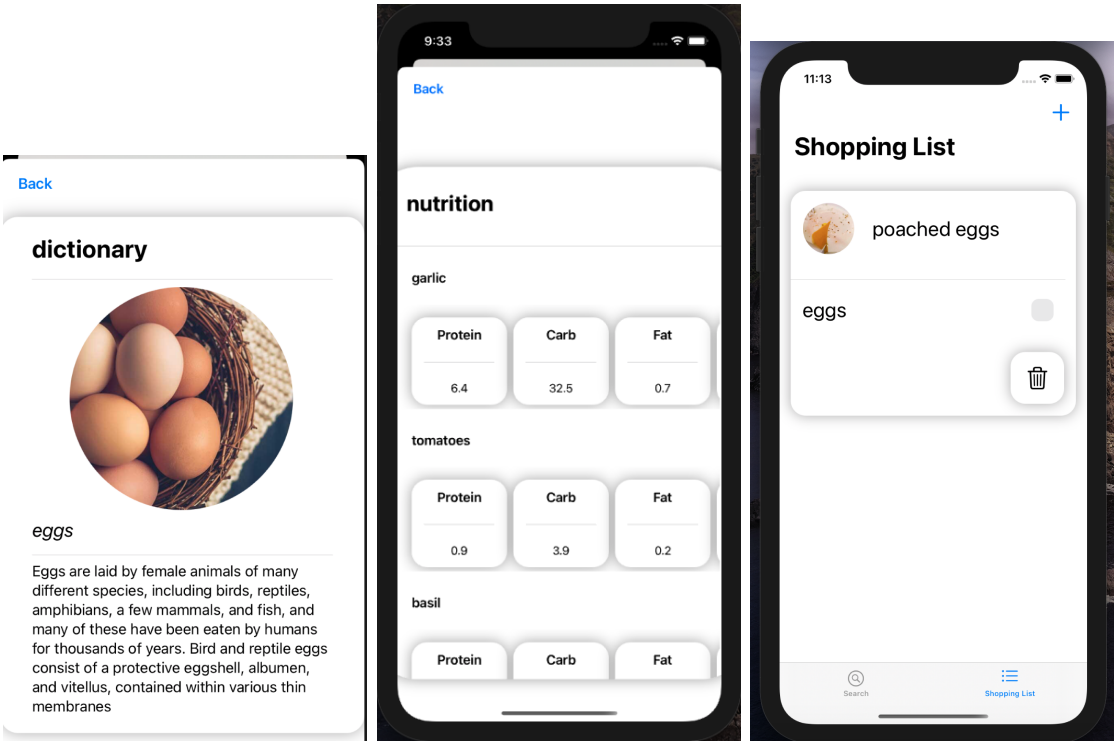


Figure 3.8: Home Page & Recipe Details View

The dictionary helps users to understand cooking items and terms that they might not know. If this was not a built in function in the app, a user would have to exit the app and google the definition or just skip the word and continue on the lesson without a full understanding of the lesson materials. Upon clicking a word (which is underlined in blue to indicate it is defined in the dictionary) a full screen cover displays an image of the word and a definition. Many apps display the nutritional information by default in their recipe details but we decided to have users click the button to view the nutritional info. This reduced the amount of information on the screen, from testing we discovered only 0% to 5% of users actually want to know the information. Also by displaying the nutritional info on a separate screen the information could be more eloquently displayed. The shopping list aims to display the ingredients a user needs to cook a recipe. Each ingredient is listed under the header of the recipe it is purposed for. A user can click the checkbox to indicate that they have the ingredient and they can also click the plus button

to alter the quantity needed, to remove ingredients from the list or to add ingredients to the list. The bin button can be used to remove the recipe from the shopping list.

3.4.3 Cooking Mode

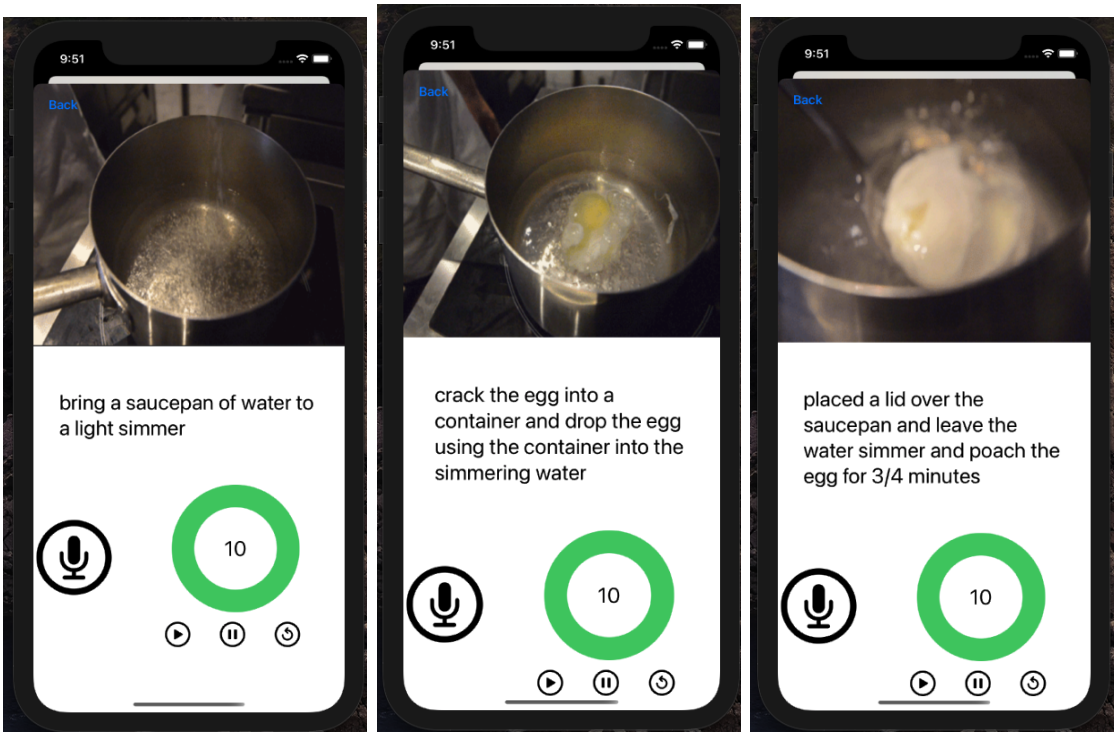


Figure 3.9: Different Instructions in a Cooking Lesson

The cooking mode displays each instruction in the recipe lesson. Initial prototypes displayed the instructions of the recipe in the recipe details page and in the cooking mode, but we decided that the instructions should only be shown in cooking mode because we wanted to encourage users to enter a lesson mode as opposed to using the instructions on the recipe details page. Each instruction is accompanied by an image, which demonstrates the instruction being carried out. Below the image is the instruction in text, similarly to the ingredients display, words underlined in blue can be viewed in a dictionary display. A timer button is displayed, which allows users to enter a timer view. Users can select a time for the timer and click play to begin the timer, pause to stop the timer or restart to start to the timer again. It was also our goal to implement voice command but unfortunately this was beyond the scope of our project.

3.5 Add Recipe, Edit Recipe & Edit Shopping List

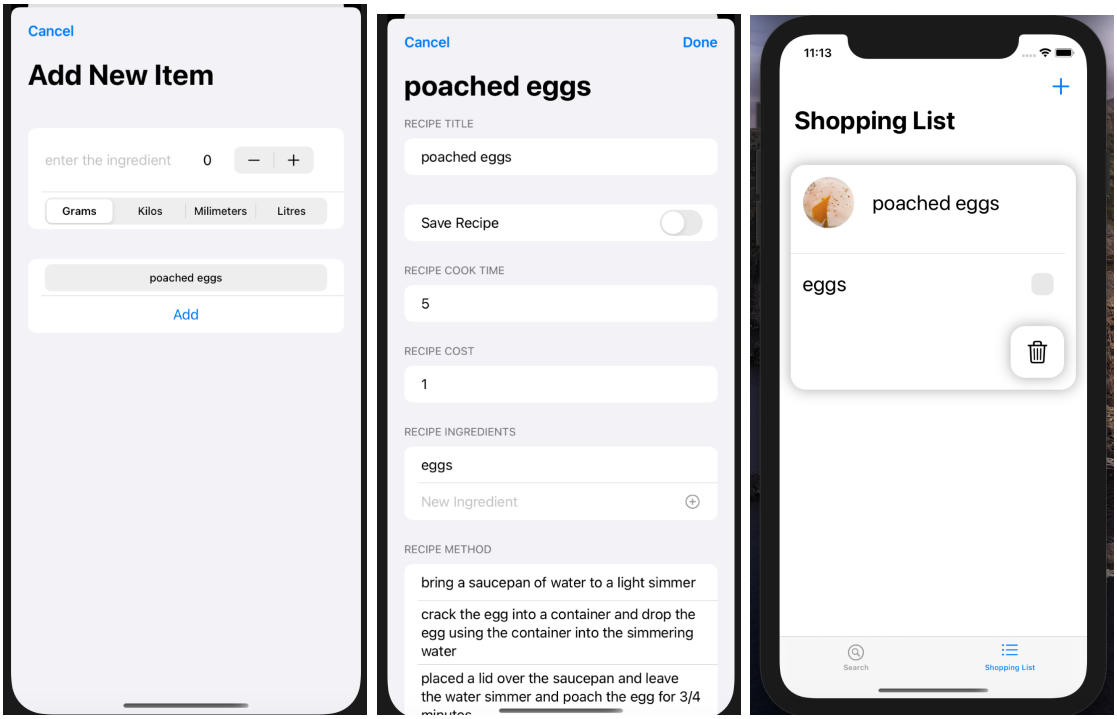


Figure 3.10: Different Instructions in a Cooking Lesson

These three pages share similar functionality. In order to improve user customization, we decided to allow users to add their own recipes to the system and edit recipes already in the system and their custom recipes. Each page uses the ios form element, this results in a data entry page that users would already recognize. Each page has different sections where users can enter the relevant information. Different input methods are used to make it easier for user to enter the correct data, for example a text field is used for the recipe title and a toggle button is used to indicate if the recipe is saved or not.

3.6 Design Process Conclusion

In this section, we explained the iterative design approach followed as our project methodology. This involved gathering system design requirements using different data gathering techniques. Literature research, task analysis, use case diagrams, user stories and online surveys. These requirements were used to create an initial prototype of the system. We began with low fidelity. hand drawn prototypes to gather initial feedback from users. The prototypes were analysed and then new, higher fidelity prototypes were created using the feedback from testing and the additional requirements gathered from the analysis.

Eventually, we reached a final implementation which successfully incorporated all of the requirements of the system. We then explained a brief overview of the final design. The next section, Design Analysis, goes into detail about some of the design requirements of the system and how they were implemented.

Chapter 4

Design Analysis

The previous section discussed the requirements of the system and the final design. This section focuses on explaining the different design aspects of the final design. We want to clearly explain how each element of the design of the system is influenced by m-learning, usability or both.

This section will discuss how each mobile learning technical aspect, using the adapted technical quality model developed by Sarrab et al (2016), was implemented in the design. The section will also perform the same design analysis but with a focus on the usability aspects of the design, using heuristic principles developed by Nielson (2005). It is important to note that some principles are present in both the mobile learning model and the usability heuristics, for example flexibility is both an aspect of m-learning and a heuristic principle of usability. In this case, the principle is discussed in one of the sections and is omitted from the other, in order to avoid needless repetition.

4.1 Mobile Learning Design

This section explains how each of the mobile learning design principles, according to Sarrab's technical quality model, are implemented in our final design.

4.2 Availability

All information and data in the system are available without an internet connection, with the exception of the nutritional information, because it uses an API to retrieve the nutritional info. It was an option to implement a system that used an online database to retrieve lesson information, but we instead opted to make sure that the lessons were

available anywhere at anytime, embracing the mobility and availability aspects of mobile learning. Another element of availability is accessibility, we decided that one major method we could implement to increase the accessibility of the app was to use SwiftUI's accessibility elements. Coded into the program, the accessibility elements increase the availability of the application to users with disabilities. Used in conjunction with the built in VoiceOver service (which is installed in every iOS device) each screen in the app is programmed to be read aloud to users, who may not be able to see or read well. A demonstration of this can be found here: https://github.com/thornec/FinalYearProject.

4.2.1 Quick Response

One major reason we chose SwiftUI and Xcode was because of the native performance, which means the app would perform very quickly because it was specifically programmed to be used on a specific device. By avoiding packages, using a locally stored database, using only native elements and using only hand written static code the app has very quick response time, resulting in no loading times on any of the pages.

4.2.2 Flexibility

Flexibility, in terms of m-learning, is concerned with the customizability of the data within lessons. Our system is completely customizable, all data in every lesson is entirely flexible. Users can add their own recipes, edit every detail of any recipe in the system and edit any shopping list item to their liking. By implementing these features users could tailor lessons to their preference, for example if a user is following the poached eggs lesson and decides after competition of the lesson they prefer their eggs to be cooked for longer, they can customize the instruction so that it instructs 5 minutes of cooking instead of 3.

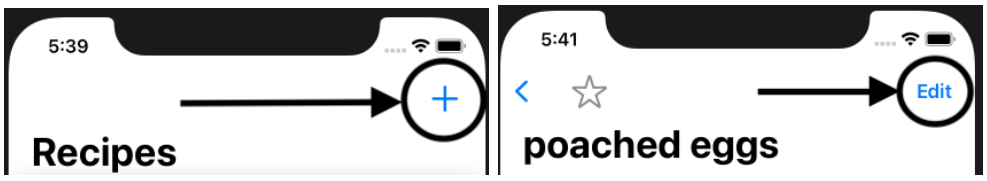


Figure 4.1: Customization Buttons in App

4.2.3 Functionality

The system was designed to incorporate every piece of functionality that would fulfill all over the user's requirements. The major aspects of functionality in the app are the search bar, tab bar filter, shopping list, dynamic editing of data, cooking mode, nutrition api

and timer. Minor aspects are the incrementation and decrementation of serving sizes, deletion of shopping list items and favourite recipe function.

4.2.4 Reliability

The system reliability was an important aspect of the development process. It was important to ensure that the edited and added data was retained once the app was closed, this was achieved and is discussed in more detail in the implementation section. At the beginning of testing, many bugs and crashes were discovered but due to the number of evaluations we performed the finished implementation has never crashed during any testing and is totally reliable.

4.2.5 Connectivity

Connectivity, in terms of mobile learning, is focused on how the user is connected to the device during lessons. It was important to understand how we could achieve a connection between the user and the lesson. We implemented a cooking mode, which requires the user to adjust the screen using a hand gesture. By forcing a user to swipe from instruction to instruction and displaying each instruction to cover the entire screen there is a better connection between the user and the lessons then there would be if the instructions were statically displayed on the screen.

4.2.6 User Interface

The clear system userface was difficult to implement, purely because recipes can contain so much information. We avoided any branding and used mainly black and white colors to reduce user's cognitive load. Buttons were used to replace static information, for example instead of always having the nutritional information displayed on a recipe details page it is accessed through a button, reducing the amount of information on one screen by spreading it out over two. The interface is discussed in more detail in the next section, usability design.

4.3 Usability Design

This section explains how each of the mobile usability design principles, according to Nielson (2005), are implemented in our final design.

4.3.1 Visibility of System Status

Our design keeps users informed by utilising navigation bar elements. Navigation bar elements are objects that appear at the top of the screen and help users to understand where they are in the app and how to return to previous pages. In our application the navigation titles are present in every page which reminds users where they are in the app.

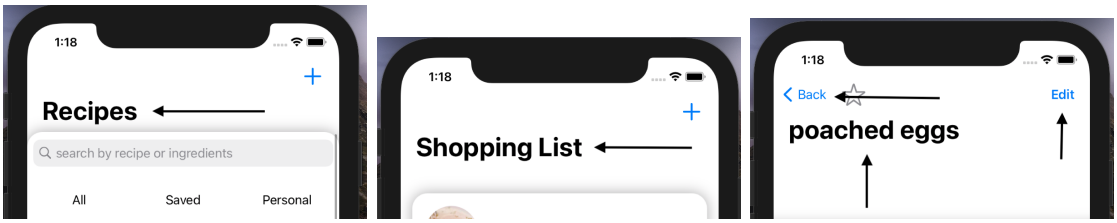


Figure 4.2: Navigation Bar Elements Shown with Arrows

The bottom toolbar is separated into recipes and shopping list. By separating the app through functionality, users intuitively can navigate more freely knowing if they are looking for recipes they should be in one tab and if they need the shopping list it is in the other tab. One design aspect implemented because of feedback from testing was alerting user's to successful actions. Users in testers were pressing the button to begin the timer or to add a recipe to the shopping list and were unsure if the action was successful. We implemented pop up messages that occur when this happens, to show user their action was successful (see Figure 36).

Figure 4.3: Bottom toolbar which seperates system functionality, the blue highlights if the user is in the recipe section of the app or the shopping list section

4.3.2 Match Between System & Real World

All of the language in the system uses simple English. Any error message or system message uses purely real world phrases and any use of complicated technical language was avoided.

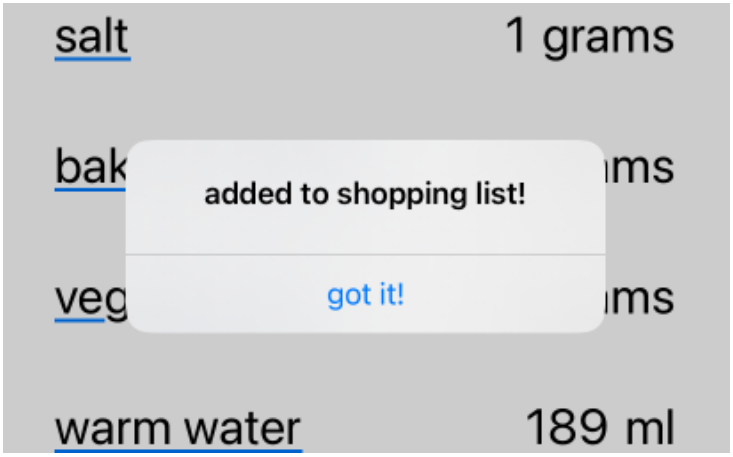


Figure 4.4: Use of friendly & natural language

4.3.3 User Control & freedom

As mentioned, the use of navigation bar items like back, cancel, done allows users to navigate freely throughout the simple and easily cancel any actions or return to previous pages.

Figure 4.5: Cancel and Return Buttons

4.3.4 Consistency & Standards

The consistency of the design of the system was important because we not only wanted the system to be consist within itself but also with all other applications. We focused on using ios conventions such as SF Font, SF Symbols and built in xcode elements. By using font, symbols and elements that users have already used before in other apps the overall cognitive load of a user is reduced and the intuitive understanding of the system for a user is greatly improved. Another consistency between the app and other apps is that any button is highlighted in blue text, we found after implementing this testing demonstrated users intuitively clicked words that were blue, knowing that they were buttons.

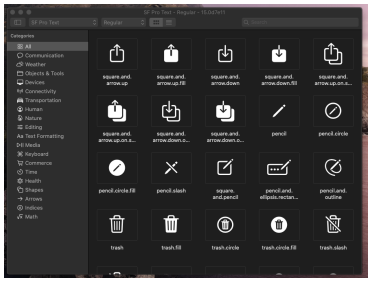


Figure 4.6: SF Symbol package used for finding recognizable symbols to improve the system consistency and standards, aswell as the recognizablility

4.3.5 Error Prevention

Our careful design worked to reduce any errors a user may make. There are no real system errors a user can cause. The only errors a user can incur is through accidentally clicking the wrong page, the chance of this happening is reduced by using navigation titles and using cancel buttons.

4.3.6 Recognition Over Recall

Our user interface minimizes memory load by using labels and symbols. Labels explain what action will occur from using a button, this means that a user does not have to try remember what a button does because it is written below it. Symbols also clearly explain what information means, user recognize symbols like stars, clocks and + from other systems and generally know what functionality they utilize. Also, by avoiding packages and using only native elements, users already intuitively understand how to swipe through pages, change tabs and edit data using forms.

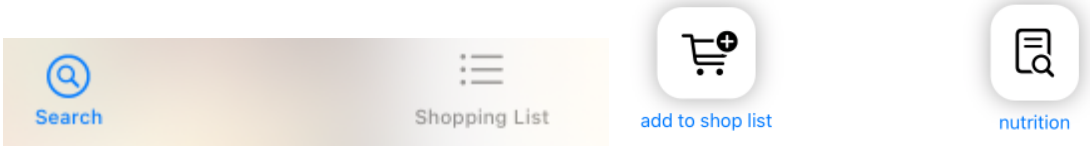


Figure 4.7: Cancel and Return Buttons

4.4 Limitations of Work

Part of the challenge of the project was to try implement as many features as possible to fufill the requirements of usability and mobile learning. Unfortunately, due to mostly time considerations, there were a few features we could not completely implemented that we had initially decided were system requirements. In this section we give a short description of each design element, its purpose and why it was not part of the final implementation.

4.4.1 Voice Command

In the cooking mode, we wanted users to be able to use voice command to navigate from instruction to instruction. This was to achieve a ”hands free” form of usability, so that if a user could not drag the screen to move to the next instruction they could use voice commands to navigate the screen. Unfortunately, our experience with programming an app was limited and this was a feature that would require a lot of time and programming to successfully implement. We decided that our time would be better spent implementing other features within the same time frame it would require to implement this rather nice but non-essential requirement.

4.4.2 Android Implementation

One aspect of availability that was beyond the scope of this project was an implementation of the device for android. Unfortunately, the only way that would have been possible would have been to develop the system using React native. As mentioned in section 2.5.1, we decided that the drawbacks of using React native did not outweigh the benefits developing in SwiftUI. Our approach was to develop the perfect version of the app and the final implementation could be then used as a simple blueprint to copy the system onto android. While the app is available and adaptable to all different ios devices (including iPad) it is not yet implemented on android.

4.4.3 Custom Photos

One aspect of customizability that we failed to implement was the changing of photos of recipe and users uploading their own recipe photos. Unfortunately, we attempted to implement these features but they were beyond our programming ability and we were unsuccessful at implementing perfectly so we decided to leave them out.

4.5 Design Analysis Conclusion

Our goal was to implement all of the design requirements and we wanted to explain how each element of the design successfully fulfilled this goal. The explanation of the mobile learning design requirements and the mobile usability requirements went into detail on where they were implemented in the system and how the implementations provided the functionality to fulfill the design principles. We also discussed the design goals that were not part of the final implementation due to various project limitations. In this section we also mentioned the evaluation of the prototypes but not in much detail. The following section focuses entirely on the various forms of evaluation that were used during the design process and design analysis.

Chapter 5

Implementation

The following chapter focuses on the implementation of the system. Each section focuses on a specific aspect of the system, its functionality and how it was implemented. The section also discusses key challenges faced during the process of implementing the system.

5.0.1 Development Environment

This decision was one of the most important parts of the project because this would be the main development environment that we would be working on for many months and the particular tools each IDE provided would have a major impact on the outcome of the app. It was decided to use XCode to develop the app. One major drawback of this decision was that the app would as a result not be available on android devices, which would have an impact, in terms of usability and mobile learning, on the "accessibility" of the device, but the advantages of using SwiftUI far outweighed the drawbacks

- The project focused on user interface design and it was apparent that Xcode and SwiftUI provided by far the most supportive tools for developing flawlessly designed interfaces and many iterations of interfaces
- An issue with developing using react native or android studio is that performance of the app is effected because it is not running native performance. We wanted the app to run flawlessly so that testing could provide the best results, making SwiftUI the better choice
- Xcode provided many native ios interface elements such as SF Symbols, User interface elements
- SwiftUI provided the ability to code accessibility elements, which could be used along with voice read to create the most accessible app possible, which is discussed

in further detail here:

- Apple have great resources online and user interface guidelines

5.1 System Architecture & Navigation

The system architecture of the app is a conceptual model that defines the structure and behavior of the app. It is one of the most important aspects of the app, the ease user's find to navigate through an app is an important part of the usability of the app and thus the effectiveness of the mobile learning. We opted to use a traditional hierarchical navigation layout, which is where a user can make choices to progress through different views of the app, returning back to their original home. The layout is split by functionality, a bottom tab separates the two layouts. One is for recipes and the other is for the shopping list. The design of the navigation was influenced by the following

- Number of Taps Required to Complete Task
- Use of Standardized Components
- Clear Path
- Categorizing Items by their Function

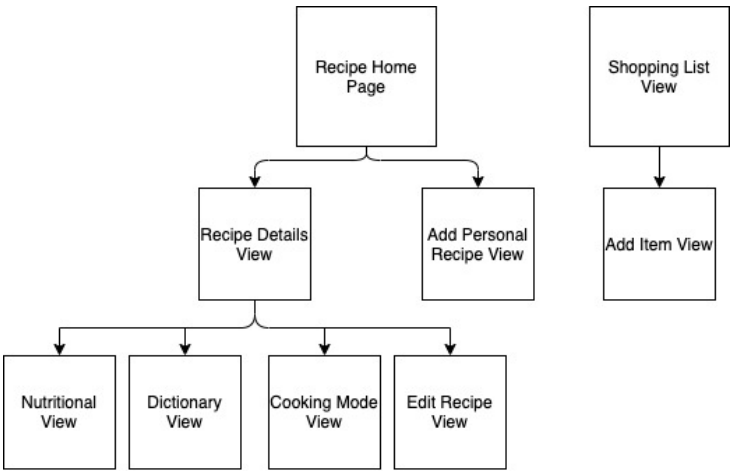


Figure 5.1: System Architecture Layout

The hierarchical system, shown above, makes it easier for users to navigate because they know that they can proceed forward and if necessary proceed backwards.

5.2 Views

In SwiftUI a view is a user interface and in most cases views are made up of multiple views. Throughout the project, when implementing an interface design the following steps were taken; create a SwiftUI view, implement the features of the design (text, image etc.) and breakdown the view into smaller sub views. This makes code more readable, less dense and sub views can be used multiple times throughout the app. The following diagrams demonstrate the home page view, which is made up of the search bar view, sliding tab view and category row view. Each of these views are also made up of smaller views, for example the category row view is made up of recipe card views. Views are essentially

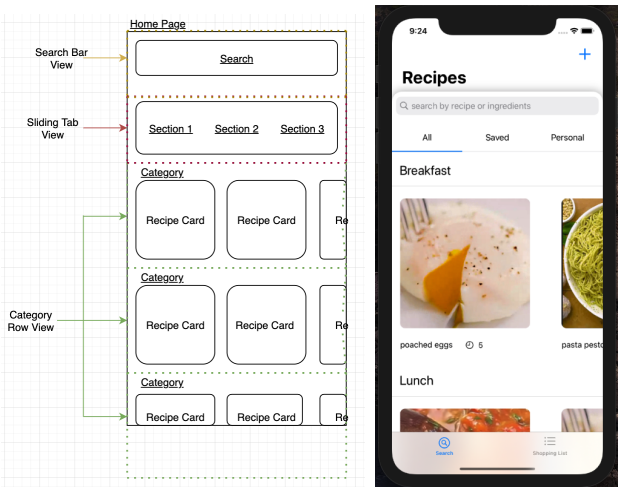


Figure 5.2: Home Page View Layout

reusable user interface components. Throughout the system, views were broken down into smaller components as much as possible, using less code in the app, making performance better and reusing the same views throughout the app to achieve more consistency.

The above image shows the recipe row view, which is a combination of the recipe cards for the category (breakfast, lunch or dinner) on horizontal scroll.

The code is the home page view and demonstrates the view breakdown, with the searchbar view, sliding tab view and category views being called (the category view being called depends on the selected tab "all", "saved" or "personal").

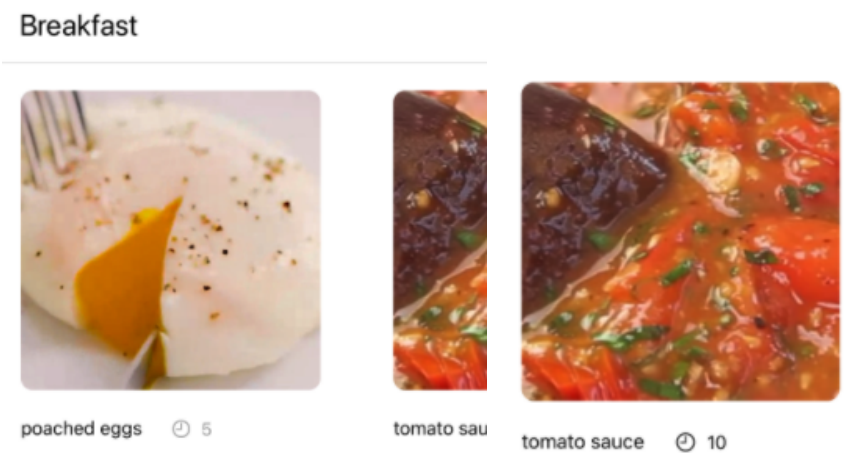


Figure 5.3: Recipe Row and Recipe Card

```
simplecook 〉 simplecook 〉 Views 〉 Recipes 〉 CategoryHome.swift 〉 No Selection
85      }
86      // filtered version displaying custom dinner recipes only
87      var personalDinner: [MyRecipeModel] {
88          filteredDinner.filter { recipe in
89              (recipe.isCustom)
90          }
91      }
92
93      var body: some View {
94          NavigationView {
95              ScrollView(.vertical) {
96                  VStack{
97                      // search bar view
98                      SearchBar(text: $searchText, placeholder: "search by recipe or ingredients")
99                      // sliding tab view
100                     SlidingTabView(selection: self.$selectedTabIndex, tabs: ["All", "Saved", "Personal"])
101                     // category rows for "all"
102                     if(selectedTabIndex == 0){
103                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist, category: "Breakfast", recipes: filteredBreakfast)
104                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist, category: "Lunch", recipes: filteredLunch)
105                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist, category: "Dinner", recipes: filteredDinner)
106                     }
107                     // category rows for "saved"
108                     if(selectedTabIndex == 1){
109                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist, category: "Breakfast", recipes: savedBreakfast)
110                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist,  category: "Lunch", recipes: savedLunch)
111                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist,  category: "Dinner", recipes: savedDinner)
112                     }
113                     // personal
114                     if(selectedTabIndex == 2){
115                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist,  category: "Breakfast", recipes: personalBreakfast)
116                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist,  category: "Lunch", recipes: personalLunch)
117                         MyRecipeRow(recipesBinding: $recipes, shoppinglist:$shoppinglist,  category: "Dinner", recipes: personalDinner)
118                     }
119                 }
120                 .font(.title)
121                 .background(Color.white)                    // make color white
122                 .foregroundColor(.black)                   // sets color of text
123                 .cornerRadius(20)                          // rounds corners
124                 .shadow(radius:9)                          // shadow
125             }
```

Figure 5.4: Home Page View Code

5.3 Search Bar

The search bar view provides the functionality of searching recipe by either ingredient or title. It functions by filtering through a list of recipes, using the currently searched terms as a filter. If a recipe contains the searched term entered into the search box, it will be displayed to the user. The search results are also influenced by the sliding tab

view. Depending on the current tab, the search will be further filtered. For example, if the current tab is "saved" and the user has searched for "tomatoes", only recipes that have tomatoes and are saved will be displayed. State variables are used to hold the search term entered into the search bar. The filteredSearch variable computes the filtered version of recipes to be displayed using the current search term (converted to lowercase to avoid inconsistencies). The filteredSearch variable is then used to calculate filtered lists of each recipe according to category, so that the list can be passed to the myreciperow to be displayed [Figure 45.]

```
28      // state variables for search
29      @State private var searchText : String = ""         // current search term value
30      @State private var showSavedOnly = false            // save mode variable
31
32      // compute filtered version of recipe list
33      var filteredSearch : [MyRecipeModel] {
34          recipes.filter { recipe in
35              (recipe.title.lowercased().contains(self.searchText.lowercased()) || self.searchText.isEmpty) && (!showSavedOnly || recipe.isSaved)
36          }
37      }
38
39      // filtered version displaying saved breakfast recipes only
40      var filteredBreakfast: [MyRecipeModel] {
41          filteredSearch.filter { recipe in
42              (!showSavedOnly && recipe.categoryName == "Breakfast" || recipe.isSaved && recipe.categoryName == "Breakfast" )
43          }
44      }
45
46      // filtered version displaying custom breakfast recipes only
47      var personalBreakfast: [MyRecipeModel] {
48          filteredBreakfast.filter { recipe in
49              (recipe.isCustom)
50          }
51      }
```

Figure 5.5: Code for Filtering Recipe List using Search Term

The search bar UI used the UISearchBarDelegate, a protocol developed by Apple. This was, instead of designing a custom search bar ui, because the design was more consistent with other apps and users would be more familiar with this search bar.

5.3.1 Sliding Tab View

The sliding tab view allows the user to alter what type of recipes are being display. It makes it easier for a user to filter recipes into the following three categories.

- All Recipes
- Saved Recipes (Personally Saved Recipes)
- Custom Recipes (Recipes Added by the User)

The sliding tab works as a state variable "selectedTabIndex". The three categories are represented by 0,1 and 2. Depending on the value, different recipes are displayed in the breakfast, lunch and dinner rows. For example, if the selectedTabIndex is 2, that means

that the current tab is custom recipes and so a filtered version of the recipes only of recipes made by the user will be displayed on the screen.

5.4 Recipe & Shopping List Model

In the system, a list of recipes and shopping list items are the two objects that are displayed. The *MyRecipeModel* and *MyShoppingData* are the data models that define the attributes that make up each object. The recipe model has all the attributes associated with a recipe, such as title, category, ingredients, servings, recipe image, method instruction images etc.
The shopping list model has similar attributes, but a more refined version because the shopping list items have no need for method instructions etc.

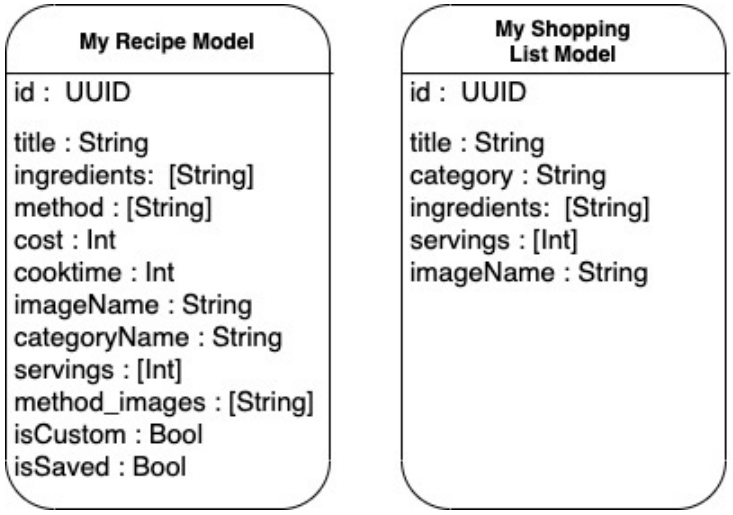


Figure 5.6: Recipe and Shopping List Models

5.5 Local Database

We decided that in order to fulfill the mobile learning and usability requirements a locally stored database. Instead of the app needing to access a cloud using an internet connection, the database of recipes and shopping list is stored locally on the user's device. This also means that once the app is downloaded, users can edit and change recipes and add their own without augmenting the recipe data of any other user because it is all stored locally. It also means that user's could access the app and its data at any location, boosting availability. The entire database for all recipes and shopping list items is locally stored on a user's phone. Initally, we decided to store the recipes on a json file and use the json

as a locally stored database. We sucessfully set up a data model (Recipe) and writing code to load information from a json into a list of recipes.

```
    // load data from database into arrays
    @Published var recipes: [Recipe] = load("recipesData.json")

}


// load method fetches JSON data using filename
func load<T: Decodable>(_ filename: String) -> T {
    let data: Data

    guard let file = Bundle.main.url(forResource: filename, withExtension: nil)
    else {
        fatalError("Couldn't find \(filename) in main bundle.")
    }

    do {
        data = try Data(contentsOf: file)
    } catch {
        fatalError("Couldn't load \(filename) from main bundle:\n\(error)")
    }

    do {
        let decoder = JSONDecoder()
        return try decoder.decode(T.self, from: data)
    } catch {
        fatalError("Couldn't parse \(filename) as \(T.self):\n\(error)")
    }
}
```

Figure 5.7: Unused Code Initializing Recipe List and Loading JSON Data

Once we decided to implement the customization of recipes, we realised it was better to store the files locally on the user's mobile. This way recipes could be stored locally in an writable file and not a json. We used an apple guide to help code the functions to access the phone's storage, which are in the MyRecipeData class. The class has two function load(), which loads the recipes from the phone's documents, when a user launches the app, into a global variable called myrecipes, and save(), which converts the recipe objects in myrecipes into a json and stores it in the phone's documents before a user exits the app, this makes sure any customizations are saved. When the app begins, the simplecookApp is the entry point for the app. From here, two states are initialised; a state called "data", which is of type MyRecipeData(), and "shoppinglist", which is of type ShoppingList. These two state variables act as the source of truth for the recipe data and shopping list data used in the app. They are passed through the views in the app using "bindings" from the entry view of the app to all subsequent views. The above code is of the entry point of the app. The tab bar is the bottom tab that appears throughout the app. CategoryHome() is the home page of the app which is one tab and the other tab is the shopping list ShopList(). The global state variables are intialized at the top of the program and are passed as parameters to other views. .OnAppear is a built in protocol which calls the function load() as soon as the app gets launched. In each view, bindings of the shopping list and recipes are passed from the state values in simplecookApp, the app entry point.

```swift
// method to load data into myrecipes array from the myrecipes.data file
func load() {
    // request global que with background thread service
    DispatchQueue.global(qos: .background).async { [weak self] in
        // asynchronously executing block
        guard let data = try? Data(contentsOf: Self.fileURL) else {      // load contents of
            file into constant 'data'
        //#if DEBUG
        // load sample recipes
        DispatchQueue.main.async {
            self?.myrecipes = MyRecipeModel.data
        }
        //#endif
        return
    }

    // use jsondecoder to decode scrum data
    guard let myRecipes = try? JSONDecoder().decode([MyRecipeModel].self, from: data) else {
        // error decoding file
        fatalError("Unsucessful at decoding recipe file")
    }
    // on main queue set myrecipes equal to folder data
    DispatchQueue.main.async {
        self?.myrecipes = myRecipes
        }
    }
}
```

```swift
// function to save user's recipes
func save(){
    DispatchQueue.global(qos: .background).async { [weak self] in
        // make sure self is in scope
        guard let myrecipes = self?.myrecipes else { fatalError("self is out of scope")}

        // encode recipes using json encoder
        guard let data = try? JSONEncoder().encode(myrecipes) else { fatalError("error
            encoding the recipes in json")}

        // write encoded recipes to file myrecipes.data
        do {
            let outfile = Self.fileURL
            try data.write(to: outfile) // write data to file
        } catch {
            fatalError("cant write to file")    // error
        }
    }
}
```

Figure 5.8: Code for Loading & Saving Data

```swift
// app entry point
@main
struct simplecookApp: App {

   // tab bar struct
   enum Tab {
       case featured
       case list
   }

   // State Variables
   @ObservedObject private var data = MyRecipeData()   // locally stored recipe data
   @StateObject private var modelData = ModelData()    // model data for dictionary
   @State var shoppinglist = ModelData().shoppinglist  // data for shopping list
   @State private var selection: Tab = .featured       // state of tab bar

   // returns scenes
   var body: some Scene {
       WindowGroup {
           // tab bar
           TabView(selection: $selection){
               // search page
               CategoryHome(recipes: $data.myrecipes, shoppinglist: $shoppinglist){
                   data.save()}.environmentObject(ModelData())
               .tabItem
               {
                   Label("Search", systemImage: "magnifyingglass.circle")
               }
               // shopping list
               ShopList(shoppinglist: $shoppinglist).environmentObject(ModelData())
               .tabItem
               {
                   Label("Shopping List", systemImage: "list.bullet")
               }
           }
           .onAppear {
              data.load() // load data
           }
```

Figure 5.9: App Entry Point

```
23      @Binding var shoppinglist : [MyShoppingData]
24      @Binding var recipes: [MyRecipeModel]
```

Figure 5.10: Bindings of Shopping List and Recipes

5.5.1 Adding & Editing Recipes

Once the recipe model, loading data, saving data and passing data was completed, Adding and Editing recipes was simply a matter of creating a new recipe object, assigning values for the attributes (name, cooking time, method etc.) and then storing the new recipe onto the recipe list, which is saved when the app is shut down. To add a recipe, the app

```
@Environment(\.scenePhase) private var scenePhase   // save user data when inactive
@State private var isAddMode = false                // control presentation of view
@State private var newRecipe = MyRecipeModel.Data()
```

Figure 5.11: Temporary Variable newRecipe

enters add mode which displays a form to the user to fill in the recipe data. This data is saved to a temporary variable called newRecipe. After a user submits the form and

```
.sheet(isPresented: $isAddMode){
    NavigationView {
        EditRecipeView(recipeData: $newRecipe)    // pass new recipe data to be filled
            .navigationBarItems(leading: Button("Dismiss") {
                isAddMode = false   // exit add mode
            }, trailing: Button("Add") {
                let new = MyRecipeModel(title: newRecipe.title, ingredients:
                    newRecipe.ingredients, method: newRecipe.method, cost:
                    newRecipe.cost, cooktime: newRecipe.cooktime,
                    imageName:"myrecipe", isCustom: true, categoryName :
                    newRecipe.categoryName, servings: newRecipe.servings,
                    method_images: [""], isSaved: newRecipe.isSaved)  // create new
                    recipe with values from user
                recipes.append(new) // push recipe onto list
                isAddMode = false   // exit add mode
            })
    }
    // triggered when value changes
    .onChange(of: scenePhase) { phase in
        // if unactive call save
        if phase == .inactive { saveAction() }
    }
}
```

Figure 5.12: Add Mode

returns from the page, the newRecipe data is used to copy over and save to the recipe

list. When editing a recipe, the recipe data is passed to the form to be changed by the user. Upon returning, the newly entered data is updated.

Editing a recipe is a similar process, but instead of creating a new variable to hold the entered data, the recipe to be edited is passed and its information is updated. Once the user is finished in edit mode the recipe in data is updated with the new values.

```
.navigationTitle(recipe.title)
.navigationBarItems(
    leading: SaveButton(isSet: $recipe.isSaved).font(.title).padding(10),
    trailing: Button("Edit"){
        data = recipe.data
        isEditMode = true

    }
)
.background(EmptyView().sheet(isPresented: $isEditMode) {
    // present edit mode using entire screen
    NavigationView {
        EditRecipeView(recipeData: $data)
            .navigationTitle(recipe.title)
            .navigationBarItems(leading: Button("Cancel") {
                isEditMode = false          // return from editing
            }, trailing: Button("Done") {
                isEditMode = false          // return
                recipe.update(from: data)   // update values from edit
        })
    }
}
```

Figure 5.13: Edit Mode

5.6 Dictionary

The dictionary implementation was a challenge. It required a local database to be setup to hold all the dictionary information. A dictionary model was created and used to create an object. The object worked as a key value pair, with the key being the word to be defined and the value being its definition. Also locally stored are the photos for the dictionary definitions, they keys of which are the same as the definition. In order to check if a word required underlining to indicate it was a clickable dictionary some computational linguistics was required. Within the system, all text strings, when displayed, are examined to see if they are within the dictionary. If they are, the word is changed to be a button

and is displayed as such. All ingredients are part of the dictionary also, so when they are being displayed they are displayed as dictionary buttons instead of plain text.

5.7 Nutrition API

The API was the most technically challenging component of the system beyond the local database implementation. Initially, we wanted to implement an API that used a supermarket api to retrieve the price of ingredients but unfortunately the only Irish retailer that has an API (Tesco) had to shut down their services temporarily, due to covid. It was decided instead we could implement an API to retireve the nutritional information of the recipe. By implementing this, we could retrieve exact nutritional information to our users who were trying to eat healthier. We discovered *CalorieNinjas*, an API for retrieving nutritional information. A class called API was programmed to handle making the data request to CalorieNinja. The function loadData is called when the user requests

```
class API {
    // load api data
    func loadData(query: String, completion: @escaping (Response) -> ()){
        // request
        let myurl = URL(string: "https://api.calorieninjas.com/v1/nutrition?query=" + query)
        // make request
        if let unwrappedURL = myurl {
            var request = URLRequest(url: unwrappedURL)
            request.addValue("LRooJU3YQJPQMqVaaQjbHQ==xbZoAPJcBDmrAacW", forHTTPHeaderField: "X-Api-Key")   // add key

            // start url session with request
            let dataTask = URLSession.shared.dataTask(with: request){ (data, response, error) in
                    let data = try! JSONDecoder().decode(Response.self, from: data!)
                    // data retrieved
                    DispatchQueue.main.async {
                        completion(data)
                    }

                }
                dataTask.resume()
            }
        }
    }
```

Figure 5.14: API Class

the nutritional information. Its only parameter is the query to be made, which is a string that combines the recipe's ingredients and portion sizes. An example query could be "10 eggs and 3 cloves of garlic".A url variable is created using the API request hyperlink and the query attached. The key is then added to the variable and the request is made. Upon receiving the information, the function returns the data through the Response variable. This all takes place asynchronously, to allow the app to continue running while the request is made. The console log demonstrates an example query and the returned information from the api.

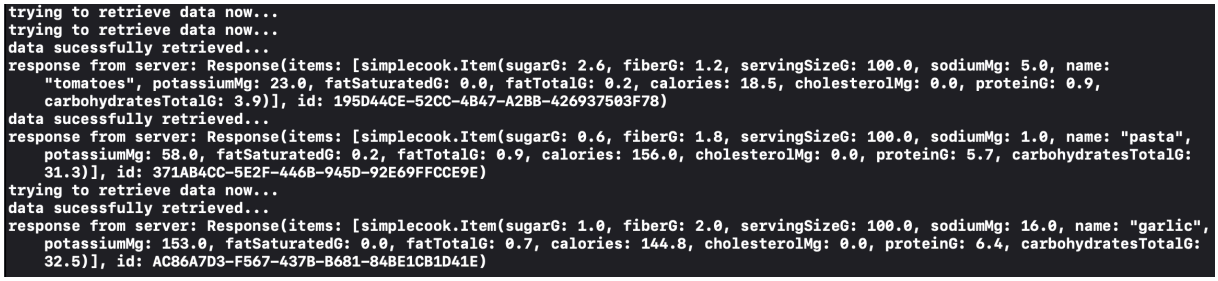


Figure 5.15: Console Log of API

Chapter 6

Evaluation

The evaluation of the system was important for understanding the user's perception of the usability and functionality of the system. As mentioned, evaluation was considered at all stages of the design cycle, by following the iterative design cycle, evaluation took place throughout the entire design process on prototypes and the final implementation. Initially, we used a low fidelity prototype for evaluating and overtime progresses with higher fidelity prototypes. The main goals of evaluation were to asses user's perception of the usability of the system and the effectiveness users had learning recipes from the app. The evaluation methods used to evaluate our system were influenced by our research conducted in chapter two. This chapter describes the testing procedure, the methods used during testing and the results from testing.

6.1 Testing Procedure

The testing procedure describes how each test is executed. It was important to come up with a procedure to be followed for each testing session, to ensure that users were performing the evaluation in similar circumstances with similar knowledge. By carrying out the same procedure with each individual, each evaluation result should be as fair as possible. Due to COVID-19, in-person testing was not an option, which carried with it a lot of complications which we go into further detail. Each testing session followed the same procedure ; Planning, Introduction, Testing, Post-Testing and Analysis.

6.1.1 Planning

It was decided that each testing session would involve asking users to complete tasks using the system. Afterwards, the users would be asked to answer a questionnaire. In preparation for an evaluation session, it was necessary to clearly define what problems/areas

the testing would focus on, this is known as the testing plan. Once the goal of the testing was defined, the tasks that the users would be asked to perform were defined. For example, in one instance we wanted to test the user's perception of the navigation of the application, so testing would focus on tasks that require the tester to navigate through the app and perform action, for example;

"Select a recipe by searching for recipes that use garlic, examine a recipe in detail, read it's instructions and add it to your shopping list"

A simple task like this could allow us to fully understand the effectiveness in which a user could navigate and understand the interface. In this task alone users are required to process the home screen, find the search bar, type the correct ingredient, examine the search results and navigate into a recipe and examine it in more detail, navigate the recipe and find the instructions, return from the instructions and find the button to add it to the shopping list and then navigate out of the recipe and to the shopping list page.

After each testing session, users would be asked to fill out a questionnaire based on their experiences using the app. The questionnaire would either be the SUS or the CE-CAM Questionnaire, depending on what the goals of the testing session were. For example, if we were testing the usability of the system the questionnaire would be the SUS, if we were testing the mobile learning aspects of the system, we would ask users to fill in the CE-CAM.

6.1.2 Introduction

Before beginning the test, it was necessary to provide each user with an introduction to the project and the goal of the testing. It was important to ensure that testers understood that it was the system that was being examined and judged, not them, in order for testers to feel comfortable and act as they would without any bias. Part of the ethics of the project was to prepare an information sheet for testers to read before testing. This sheet provided each tester with all the necessary information such as

- Background Context of Research
- Who are the Researchers
- Ideal Participants for Research
- Procedure of Evaluation

- What data will collected and stored
- Voluntary Nature of Evaluation

It was important that testers felt comfortable with the situation and the use of their data. It was made extremely clear that users could withdraw at any time, without penalty. It was also made clear that each part of the evaluation was optional. After reading the information sheet, testers we requested to sign the participants consent form, which gave their consent to take part in the study and to the data processing. The participants consent form was also part of the research ethics application, which was approved by the SCSS Research Ethics Committee.

6.1.3 Testing Session

After the planning and introduction was performed, an evaluation session would begin. Evaluations took place online, the tester would be asked to download the prototype and launch the app on their phone. The evaluations were taken place over video call on Microsoft Teams. Testers were asked to perform a number of tasks using the system. On performance of a task, code in the program recorded the following; duration of time on screen from launch, number and type of errors. The observer (person in charge of testing) would record the time taken to complete each task. Users would also be speaking aloud their general thoughts and we would record anything of particular interesting nature. After completion of the tasks, the testers were sent the SUS or CE-CAM Questionnaire to be filled in. After all testing sessions had been performed, the data would be compiled into an excel spreadsheet and analysed.

6.2 Usability Testing

So far we have described the procedure each testing session followed; definition of testing goals, informing testers, testing system and analysing results. The following section discusses the observation methods used in testing sessions to evaluate the usability of the system. In order to diversify the data, multiple methods of data gathering were utilised.

6.2.1 SUS

The main method selected to measure the usability of the system was the System Usability Scale. The main reasons for this decision were

- SUS is Easy to Administrate
- Small Sample Size with Reliable Results
- Recognized Validity

(Usability.gov, 2021, Brooke 1996, Bangor et al 2008)

The ease of ability to deploy the SUS along with the small sample size were important because of the restrictions of testing due to the covid pandemic (see Challenges & Limitations of Work). The following sus questionnaire was given to testers at the end of each evaluation.

	The System Usability Scale Standard Version	Strongly Disagree	1	2	3	4	5	Strongly Agree
1	I think that I would like to use this system frequently.		o	o	o	o	o	
2	I found the system unnecessarily complex.		o	o	o	o	o	
3	I thought the system was easy to use.		o	o	o	o	o	
4	I think that I would need the support of a technical person to be able to use this system.		o	o	o	o	o	
5	I found the various functions in this system were well integrated.		o	o	o	o	o	
6	I thought there was too much inconsistency in this system.		o	o	o	o	o	
7	I would imagine that most people would learn to use this system very quickly.		o	o	o	o	o	
8	I found the system very awkward to use.		o	o	o	o	o	
9	I felt very confident using the system.		o	o	o	o	o	
10	I needed to learn a lot of things before I could get going with this system.		o	o	o	o	o	

Figure 6.1: Lewis et. al, 2018

The results were then compiled and analysed. These results were used to reflect the overall usability of the functions tested in that session. The results would also be used by the designer to identify potential issues with the design that required attention.

6.2.2 Testing Observation Methods; Think-Aloud, Task Analysis, Data Recording

During the evaluation sessions, when users were using the system, multiple methods of observation were used to gather as much data as possible on the performance of the system. The observation methods deployed were influenced by Weichbroth (2020) due to the wholeness of his study and its contemporary relevance to mobile usability. The evaluation measures Wiechbroth proposed for testing the attributes of usability; *efficiency, satisfaction, effectiveness, learn-ability, memorability and cognitive load* were all used.

Users were simply asked to describe what they were doing, why they were doing it and what they think is happening on the system.This is known as "Think-Aloud" assessment and according to Weichbroth it is useful for measuring cognitve load. This information was then recorded into data sheets, used for analysis.

Users were asked to complete certain tasks using the app, a simple example would be to navigate through the recipes, select one and add the ingredients to the shopping list. The following were recorded during while a user was performing an instructed task:

- Time Taken to complete task
- Duration of Time on Screen
- Number and type of errors per task
- Number of users making a particular error
- Number of users completing a task successfully

This is known as Task Observation and Data Recording. It is primarily used for measuring efficiency, effectiveness, learn-ability and cognitive load of a system (Wechbroth 2020). Additionally, it was useful for testing the functionality of newly implemented designs.

The usability testing methods used depended on the particular testing session goals. The following graph, adapted from [Figure 8.] is a summarizing of the usability measurement, the attribute it would be testing, how it was performed and the context in which it would be used

Usability Measurement	Usability Attribute	Performed	Context
SUS	Overall Usability	Questionnaire filled after user completed tasks using system	Used after every usability testing session
Think-Aloud	Cognitive Load	User's encouraged to speak during task performance	Used during all usability testing sessions
Time Taken to Complete Task / Duration of Time Spent on Screen	Efficiency & Cognitive Load	Users asked to complete tasks with system, each task time was recorded. The overall time using the system was also recorded.	Used in most usability testing sessions, sometimes users were encouraged to free roam and explore the app, in these cases task time was not recorded
Number and Types of Errors per Task	Learnability, Effectiveness and Efficiency	While users were interacting with system the evaluator would record the errors made	Used in most testing sessions when a task was being performed
Number of Users Making Same Error	Efficiency	After testing, results of errors were compiled and errors which occurred commonly were analysed	Performed after testing sessions were completed
Number of Tasks Successfully Completed	Effectiveness	Users were asked to perform tasks and each successfully completed task without assistance was marked by the evaluator	Used in usability sessions testing effectiveness of the system

Figure 6.2: Summmarization of Usability Testing Methods

6.3 M-Learning Testing

The goal of the M-Learning testing was to evaluate how effective the lessons in the cooking app were at teaching users. From our research from Chapter 2, it was found that the CE-CAM Questionnaire would be the most applicable m-learning evaluation method for the system. It was found that the Diary method proposed by Vavoula (2005) would not be effective because of the limited recipes available on the prototypes and the effort that would be required by testers, who were voluntary. The other method used to evaluate the mobile learning qualities of the system was an adapted version of Sarrab's Quality Model [Figure 5.], which is not intended for use by participant's in testing sessions. Instead, the quality model evaluation was performed by the author and is discussed in Chapter 7: Design Analysis. The following section goes into detail about how the CE-CAM Questionnaire was adapted and used for the purpose of this project.

6.3.1 CE-CAM Questionnaire

As discussed in Chapter 2, the CE-CAM Questionnaire is split into a number of tables, each table focuses on a particular mobile learning aspect to evaluate. Each question within a table is scored between 1 and 5, where 1 represents *"Strongly Disagree"* and 5 represents *"Strongly Agree"*. We decided to utilise the tables used by Collantes et al (2020) because, similar to our project, they were evaluating the mobile learning utility of a mobile phone app, in their case a visual literacy app. Although one difference between our project and Huilcapi-Collantes' study is that the CE-CAM Questionnaire was used to evaluate both pedagogical usability and user interface usability, we wanted to only use

it for mobile learning evaluation purposes. Also, some questions were asked in the tables which would not be relevant to our app and because of this we decided to adapt their constructd to better suit our project. The following questionnaire tables were constructed with guidance from Cota (2016) and Huilcapi-Collantes et al. (2020)

	Educational Activities	Score	Comments
1.	Lessons aid the understanding of the content		
2.	Lessons help improve cooking skills		
3.	Lessons allows users to integrate previous learning with new lessons		
4.	Lessons reflect real life		
5.	Lessons are not to difficult		
6.	Activities are available to evaluate a user's understanding		
7.	Lessons take advantage of mobile devices (pictures, recording audio, augmented reality etc.)		

Figure 6.3: Educational Activites Questionnaire

	Educational Content	Score	Comments
1.	Lesson content is micro-sized		
2.	The learning objective is clearly defined before each lesson		
3.	Required knowledge for lesson is displayed before the lesson		
4.	Lessons are presented clearly and concisely		

Figure 6.4: Educational Content Questionnaire

	Personalization	Score	Comments
1.	App allows users to choose their learning paths		
2.	App allows users to choose lessons of varying levels of complexity		
3.	App allows users to adjust information to aid their learning		
4.	Users can add their own lessons to the app		

Figure 6.5: Personalization Questionnaire

	Multimedia Resources	Score	Comments
1.	Varied multimedia resources are presented		
2.	Multimedia resources aid learning		
3.	Multimedia lessons have a duration less than 7 minutes		
4.	The multimedia is of good quality		
5.	Multimedia can be downloaded onto the mobile		
6.	Multimedia is of appropriate memory size for a mobile device		
7.	An appropriate proportion of multimedia resources is presented in the system		

Figure 6.6: Multimedia Resources Questionnaire

6.4 Evaluation Results of Final Implementation

Throughout the project, evaluation was used to aid the design process but the final step of the evaluation process was to perform user testing on the finished implementation of our cooking app and compare the results with the same tests performed on the already available apps. We could then make a comparison of the effectiveness and usability of an app which used m-learning research and apps that are already available which do not incorporate m learning design principles.

In section 4.4.2 and 4.47 the adapted Technical Quality Model (Sarrab et al 2016) and mobile learning design principles (Grant M.M 2019) were used to demonstrate that

the apps available did not follow mobile learning principles, according to the research. Using these results, we performed user tests on each of the three available cooking apps (BBC Good Food, Recipe World) using our adapted CE-CAM questionnaire and the SUS questionnaire. The results of the questionnaires were then compiled to form overall statistics about users' opinion of the systems.

We then performed the same process but with our system; an analysis using the technical quality model and the mobile learning design principles. We performed the same user tests using the same CE-CAM questionnaire and SUS questionnaire and again, compiled the results. In this section we show the different results from testing.

6.4.1 CE-CAM Questionnaire

Following the process discussed in sections 6.1 and 6.3 we conducted testing on the available applications BBC Good Food, Recipe World and Yummly using the adapted CE-CAM questionnaire. The questionnaire report from each tester (sample of 5) was then compiled to reflect an overall score of each system. The results are as follows

BBC Good Food resulted in an overall score of; 51.4/110

Tasty resulted in an overall score of; 44.7/110

Recipe World resulted in an overall score of; 41.2/110

The questionnaire results from these evaluations are below

	Educational Content	Mean Score
1.	Lesson content is micro-sized	3
2.	The learning objective is clearly defined before each lesson	3
3.	Required knowledge for lesson is displayed before the lesson	2.5
4.	Lessons are presented clearly and concisely	3
	Overall Score	11.5/20

	Multimedia Resources	Mean Score
1.	Varied multimedia resources are presented	2
2.	Multimedia resources aid learning	1
3.	Multimedia lessons have a duration less than 7 minutes	3.5
4.	The multimedia is of good quality	2
5.	Multimedia can be downloaded onto the mobile	2
6.	Multimedia is of appropriate memory size for a mobile device	2
7.	An appropriate proportion of multimedia resources is presented in the system	2.5
	Overall Score	15/35

o	*Educational Activities*	Mean Score
1.	Lessons aid the understanding of the content	3.2
2.	Lessons help improve cooking skills	2.5
3.	Lessons allows users to integrate previous learning with new lessons	1.5
4.	Lessons reflect real life	3.0
5.	Lessons are not to difficult	3.2
6.	Activities are available to evaluate a user's understanding	1
7.	Lessons take advantage of mobile devices (pictures, recording audio, augmented reality etc.)	1
	Overall Score	**15.4/35**

	Personalization	Mean Score
1.	App allows users to choose their learning paths	3.8
2.	App allows users to choose lessons of varying levels of complexity	3.2
3.	App allows users to adjust information to aid their learning	1.5
4.	Users can add their own lessons to the app	1
	Overall Score	9.5/20

Figure 6.7: BBC Good Food CE-CAM Results

	Educational Content	Mean Score
1.	Lesson content is micro-sized	2.75
2.	The learning objective is clearly defined before each lesson	2.75
3.	Required knowledge for lesson is displayed before the lesson	3.1
4.	Lessons are presented clearly and concisely	2.5
	Overall Score	11.1/20

	Multimedia Resources	Mean Score
1.	Varied multimedia resources are presented	1.5
2.	Multimedia resources aid learning	1.2
3.	Multimedia lessons have a duration less than 7 minutes	3.7
4.	The multimedia is of good quality	1.5
5.	Multimedia can be downloaded onto the mobile	2
6.	Multimedia is of appropriate memory size for a mobile device	1
7.	An appropriate proportion of multimedia resources is presented in the system	1
	Overall Score	11.9/35

	Educational Activities	Mean Score
1.	Lessons aid the understanding of the content	3.1
2.	Lessons help improve cooking skills	2.2
3.	Lessons allows users to integrate previous learning with new lessons	1.5
4.	Lessons reflect real life	2.75
5.	Lessons are not to difficult	2.4
6.	Activities are available to evaluate a user's understanding	1
7.	Lessons take advantage of mobile devices (pictures, recording audio, augmented reality etc.)	1
	Overall Score	**13.95/35**

	Personalization	Mean Score
1.	App allows users to choose their learning paths	3
2.	App allows users to choose lessons of varying levels of complexity	2.75
3.	App allows users to adjust information to aid their learning	1
4.	Users can add their own lessons to the app	1
	Overall Score	7.75/20

Figure 6.8: Tasty CE-CAM Results

	Educational Content	Mean Score
1.	Lesson content is micro-sized	2
2.	The learning objective is clearly defined before each lesson	2.3
3.	Required knowledge for lesson is displayed before the lesson	3
4.	Lessons are presented clearly and concisely	1
	Overall Score	8.3/20

	Multimedia Resources	Mean Score
1.	Varied multimedia resources are presented	2.5
2.	Multimedia resources aid learning	2.5
3.	Multimedia lessons have a duration less than 7 minutes	3.5
4.	The multimedia is of good quality	2
5.	Multimedia can be downloaded onto the mobile	1.5
6.	Multimedia is of appropriate memory size for a mobile device	1.5
7.	An appropriate proportion of multimedia resources is presented in the system	2.2
	Overall Score	14.7/35

	Educational Activities	Mean Score
1.	Lessons aid the understanding of the content	2
2.	Lessons help improve cooking skills	2
3.	Lessons allows users to integrate previous learning with new lessons	2
4.	Lessons reflect real life	2.5
5.	Lessons are not to difficult	2.5
6.	Activities are available to evaluate a user's understanding	0
7.	Lessons take advantage of mobile devices (pictures, recording audio, augmented reality etc.)	0
	Overall Score	**11/35**

	Personalization	Mean Score
1.	App allows users to choose their learning paths	2.5
2.	App allows users to choose lessons of varying levels of complexity	2
3.	App allows users to adjust information to aid their learning	1.5
4.	Users can add their own lessons to the app	1.2
	Overall Score	7.2/20

Figure 6.9: Recipe World CE-CAM Results

6.4.2 M Learning App Questionnaire Comparison

After completing the tests on bbc good food, recipe world and tasty we ran the same evaluation with the same judges on our system. The results are as follows resulted in an

	Educational Content	Mean Score
1.	Lesson content is micro-sized	4
2.	The learning objective is clearly defined before each lesson	4.3
3.	Required knowledge for lesson is displayed before the lesson	4.2
4.	Lessons are presented clearly and concisely	4
	Overall Score	16.5/20

	Multimedia Resources	Mean Score
1.	Varied multimedia resources are presented	3.9
2.	Multimedia resources aid learning	4
3.	Multimedia lessons have a duration less than 7 minutes	4.5
4.	The multimedia is of good quality	4
5.	Multimedia can be downloaded onto the mobile	3
6.	Multimedia is of appropriate memory size for a mobile device	4
7.	An appropriate proportion of multimedia resources is presented in the system	4
	Overall Score	27.4/35

	Educational Activities	Mean Score
1.	Lessons aid the understanding of the content	4
2.	Lessons help improve cooking skills	4.3
3.	Lessons allows users to integrate previous learning with new lessons	3.5
4.	Lessons reflect real life	4.2
5.	Lessons are not to difficult	4.1
6.	Activities are available to evaluate a user's understanding	0
7.	Lessons take advantage of mobile devices (pictures, recording audio, augmented reality etc.)	3.9
	Overall Score	**24/35**

	Personalization	Mean Score
1.	App allows users to choose their learning paths	3.6
2.	App allows users to choose lessons of varying levels of complexity	4
3.	App allows users to adjust information to aid their learning	5
4.	Users can add their own lessons to the app	5
	Overall Score	17.6/20

Figure 6.10: M-Learning Cooking App

overall score of; 85.5.

6.4.3 Technical Quality Model

Using our results from 2.4, we could perform the same evaluation of our app and compare the results. Below is a summary of the existing application's quality model score resulted

Technical Aspect	Description	BBC Good Food	Recipe World	Tasty	M-Learning System
Availability	Accessibility associated with m learning	2	2	2	5
Quick Response	Fast system	4	3	3	4.2
Flexibility	Customization options	0	0	0	5
Usability	Ease of use	3	2	3	4.5
Functionality	Good functionality of system based on user needs	2	2	3	4
Reliability	reliable functionality of system	4	2	2	4
Connectivity	Connection between user and device	1	1	3	4.2
User Interface	Clear simple user interface	3	2	3	5
Overall Result	scores / total	19/40	14/40	19/40	35.9/40

Figure 6.11: Quality Model Comparison Results

in an overall score of; 35.9 out of 40.

6.4.4 Comparison Conclusions

From the evaluations performed on the available apps and our app, it is clear that our m-learning app has been judged to score much higher results in both tests. Our m-learning app scored 85.5/110 in the CE-CAM Questionnaires and 35.9 in the technical quality model evaluation, almost doubleing the scores of the already available apps. The evaluations represent the user's perception of the respective system, each evaluation demonstrated that our app was more effective at teaching lessons and scored higher in all constructs of the ce-cam questionnaire and the technical quality model. One of our project goals were to demonstrate that the implementation of m-learning design principles into our app would result in better evaluation results and thus a better learning experience for users. The evaluations carried out and described in this chapter have demonstrated that our app successfully achieved this goal.

Chapter 7

Conclusions & Future Work

7.1 Brief Review

Reviewing the project, we initially set out with a goal to research mobile learning design principles, implement them in an app for learning how to cook and then evaluating the effectiveness of the newly designed system. In our background research, we examined mobile learning, design principles of usability and mobile learning, the evaluation methods commonly used and the existing mobile applications. We also reviewed what technologies could be used in the project and what ethical considerations should be made. Next, we began the design process. We decided on taking the iterative design approach, we gathered the requirements of the system using various methods and we developed prototypes, which allowed us to gather feedback from users. We also discussed the final design and went gave a broad overview on each part of the mobile app. Following the design process section, the design analysis section examined the design choices of the system and how the usability and mobile learning requirements were implemented in the app. After discussing the design process and the design analysis, we discussed the evaluation of the system. This section explained how evaluations influence the design, the testing procedures taken and the approach taken to evaluate the usability and mobile learning aspects of the system. This chapter also discusses the evaluation results of the final implementation and how they compared to the applications already available. The final chapter discussed this implementation, focusing on explaining exactly how certain aspects of the project were implemented in code.

7.2 Brief Results

From our analysis, our project was successful in implementing an app which was more effective at teaching users cooking lessons. Our results show that from evaluations on apps already available that aimed to teach users, less than 50% of the technical aspects of mobile learning were actually implemented in these apps. To further this point, the apps scored badly in a ce-cam evaluation, each scoring below 20/40 points which demonstrates the apps lacking in ability to teach users. In comparison, our m-learning design focused app had better scoring in evaluations of mobile learning and usability tests, with an 85.5/110 in the ce-cam questionnaire and a 35.9/40 in the technical quality model comparison. Our evaluations demonstrates that by implementing mobile learning design fundamentals into an app the result will be a better learning experience for users.

7.3 Challenges & Limitations

The scope of the project was a constant challenge. Unfortunately, we were forced to limit ourselves on what we could accomplish. Components of the app such as the voice command and the social network could not be implemented due to the time and effort that would be required, resulting in certain design choices being made, limiting the features of the app.

Covid-19 also effected the project. For the evaluation, we adapted and overcame any of the limitations due to covid. Because we were not allowed to meet in person, testing took place online. This may have not been quite the experience as in-person laboratory testing, but we took it as a challenge to be overcome.

We tried our best to evaluate our system in the most whole way possible. We used multiple methods of testing where possible, triangulated scores of multiple testers and stuck to strict testing procedures with the goal of gathering unbiased scores. We would like to have done much further testing, employing long term testing, different methods and larger testing audiences. It also would have been beneficial to meet with experts in mobile learning and usability to discuss our design, proposals and evaluations. Unfortunately, the time frame and scope of this project limited the possibilities and perhaps in the future this work could be carried out.

7.4 Future Work

This project also clearly outlines an approach that could be applied to any app available. Simply take an app already available that is not successful at teaching, use the same usability and m-learning design fundamentals outline in section 2 and create a system using the design process outline in section 3. Many apps already available could be improved by following the process outlined in this paper, using the same approach and evaluation methods we employed. It would be interesting to see future work following the approach of our project.

In terms of academia, this research project has proved the utility of evaluation methods created in research. The project also demonstrated that by implementing the design principles of usability and m-learning, according to literature, a better learning experience can be created, thus validating the current mobile learning design research.

In terms of the app, work could always be continued on it. Development of voice command technology and a social media aspect of the app, which could not be accomplished due to the scope of the project, could be designed, implemented and tested. This project also only focused on the technical side of implementing the recipes, without much attention being paid to how the instructions of recipes may effect learning, In this regard, future work could be performed on specifically the pedagogical design aspects of lessons on m-learning systems.

Bibliography

Adam M., Young-Wolff K.C., K. E. (2015). Massive open online nutrition and cooking course for improved eating behaviors and meal composition. *The international journal of behavioral nutrition and physical activity*, 12:143.

Bangor, A., Kortum, P. T., and Miller, J. T. (2008). An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction*, 24(6):574–594.

Brooke, J. (1995). Sus: A quick and dirty usability scale. *Usability Eval. Ind.*, 189.

Cena, H. and Calder, P. (2020). Defining a healthy diet: Evidence for the role of contemporary dietary patterns in health and disease. *Nutrients 2020*, 12:334.

Cota, C. X. N. (2016). *framework para evaluar la usabilidad de sistemas m-learning: un enfoque tecnológico y pedagógico*. PhD thesis, Universidad de Castilla-La Mancha.

Flaherty, S. J., McCarthy, M., Collins, A., and Mcauliffe, F. (2017). Can existing mobile apps support healthier food purchasing behaviour? content analysis of nutrition content, behaviour change theory and user quality integration. *Public Health Nutrition*, 21:1–11.

Geddes, S. (2004). Mobile learning in the 21st century: benefit for learners. *Knowledge Tree: An E-journal of Flexible Learning in VET*, 6:13.

Grant, M. M. (2019). Difficulties in defining mobile learning: analysis, design characteristics, and implications. *Educational Technology Research and Development*, 67:361–388.

Hashemi M., Azizinezhad M., N. V. N. A. (2011). What is mobile learning? challenges and capabilities. *Procedia-Social and Behavioral Sciences*, 30:2477–2481.

Hruby A., Manson J., L. Q. M. V. R. E. S. Q. W. W. C. H. F. (2016). Determinants and consequences of obesity. *American journal of public health*, 106(9), 1656–1662.

Huilcapi-Collantes, C., Hernández Martín, A., and Hernández-Ramos, J. P. (2020). Pedagogical and user interface usability evaluation of an educational mobile app that promotes visual literacy. In *Eighth International Conference on Technological Ecosystems for Enhancing Multiculturality*, TEEM'20, page 315–321, New York, NY, USA. Association for Computing Machinery.

ISO (2018). Ergonomics of human-system interaction — part 11: Usability: Definitions and concepts. *ISO*, 9241.

Jahnke, I., Lee, Y.-M., Pham, M., He, H., and Austin, L. (2020). Unpacking the inherent design principles of mobile microlearning. *Technology, Knowledge and Learning*, 25.

Krug, S. (2005). *Don't Make Me Think: A Common Sense Approach to the Web (2nd Edition)*. New Riders Publishing, USA.

König, A., Gulbahar, Y., and Jacobs, C. (2015). *Chapter 4: Mobile learning in higher education: Current status and future possibilities*, pages 33–42.

Lewis, J. and Sauro, J. (2018). Item benchmarks for the system usability scale. 13:158–167.

Nielsen, J. (1994). *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Nielsen, J. (2005). Ten usability heuristics. 24.

Nielsen, J. and Molich, R. (1990). Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, page 249–256, New York, NY, USA. Association for Computing Machinery.

of Health, I. D. (2019). Healthy ireland survey 2019. *Department of Health*, 36p.

Sarrab, M., Elbasir, M., and Alnaeli, S. (2016). Towards a quality model of technical aspects for mobile learning services: An empirical investigation. *Computers in Human Behavior*, 55:100–112.

Shackel, B. (2009). Usability - context, framework, definition, design and evaluation. *Interact. Comput.*, 21(5–6):339–346.

Vavoula, G. (2005). A study of mobile learning practices. *WP4*.

Weichbroth, P. (2020). Usability of mobile applications: A systematic literature study. *IEEE Access*, PP:1–1.