NDMA
Interim Report

DT228
BSc in Computer Science

William Carey

C16315253

Ciaran Kelly

School of Computer Science

Technological University, Dublin

09/12/2019

Abstract

A mobile application designed to assist people in making informed decisions about any suitable diet that incorporates all the nutritional values necessary to each individual need. It will also high key macronutrients and micronutrients essential to the person's overall nourishment plan, catering to those whom have conditions, like lactose intolerance, that disables them from having certain products like milk.

Many people in the first world countries are employing diets that do not meet the WHO recommendations, which will have long lasting effects on both their generation and future generations that stem from them.

I will provide a solution that manages that for them through development iterations of the systems' features that will be reviewed in intervals. Once these iterations are completed, I will run ongoing tests on both the front-end and back-end, incorporating both automated and manual.

After the testing's, I will use peer review (mainly from those whom have manually tested the system), appropriate universal evaluation method on the application and comparison with similar technologies to ensure it is up to standard of minimum operation standards and other agreed protocols.

Declaration

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

Signed:

[illegible]

William Carey

09/12/19

Acknowledgements

I would like to take the time to provide thanks to my family, friends and lecturers who took the time to communicate with me on some ideas of the project. I would also like to provide gratitude to my supervisor whom provided assistance, such as understanding the route to take the project and providing advice to ensure the project was feasible, where it was needed.

Table of Contents

Table of figure

1. Introduction

1.1. Project Background

Physical health is one of the fundamental areas in which the people need to take care in order to thrive in life. This includes both the diet every individual undergoes combined with activities to upkeep physical aptitude. Not everyone has the time to exercise through the recommended amount of "**150 minutes [**moderate-intensity]" or "75 minutes [vigorous] of aerobic activity per week" **(2)**. What everyone has time to do is diet effectively, in which I personally had the insight growing up with. However, I know many people who did not get this knowledge.

According to a guardian article in 2018, **(1)** "Almost 20% of deaths worldwide are attributable to an unhealthy diet", while the American Dietic Association supports "appropriately planned vegetarian diets, including total vegetarian or vegan diets" as they are "shown to be healthful, nutritionally adequate, and may be beneficial in the prevention and treatment of certain diseases". By knowing this, the aim is to create the technical resources through in-depth research to provide them the knowledge of why they should undergo a healthy diet when they use such resource.

According to Health Education Research**, (3)** "There is ample evidence that" "computer-tailored nutrition education is a more effective tool for motivating people to change to healthier diets than general nutrition education". The conclusion of their tests implied people would be more likely to return to the computer medium than any other. Combined with the work of Philip Lew, Luis Olsina and Li Zhang, "Web applications (WebApps), a combination of information content, functionality and services are fast becoming the most predominant form of software implementation and delivery". As such, the hybrid mobile should follow minimum operation standards for the development of the software design and UX to improve "the user experience as a whole".

John **(4)**, in his writings, explicit states that "Circadian and diurnal rhythms affect food intake, and earlier research has suggested that meal sizes increase, where the after-meals intervals and satiety ratios decrease over the day". It was found that "when individual subjects ate a larger than the mean proportion of their total intake during the morning, they ate significantly less over the entire day. Conversely, when these same subjects ate a high proportion of their total intake during the evening, they ate significantly more over the entire day". This would indicate every necessary daily breakdown of the diet (Breakfast, lunch and dinner) are important areas of the diet which would be maintained properly.

Overall, all the features of having a good diet planner combined with a way to track the diet daily does not exist or is not user friendly for people on the go, whether it is to work or to go for exercise. Therefore, I decided to create a user-friendly mobile application for such users known as Nutrient and Diet Manager Application.

1.2. Project Description

Nutrient and Diet Manager Application (NaDMA) is an application designed to assist the user in managing their dietary through a simplistic yet intuitive UI allowing them to log their diets in the application into the system and getting advice from the recommender system. This is for those whom want assistance in achieve a specific goal in mind, the office worker whom is doing 60-80 hours and needs to optimise their health or the casual user whom just is looking for generic advice.

The main aspect of the application, which is the ability to log the user diet and getting advice on how to manage it, will be completed through the usage of a smooth, robust, intuitive and easy UI for the user. To ensure this, the application complexity will focus on the UX as the main priority. Most applications, when dealing with information on macronutrients etc, expects the user to know this information themselves and requests it for usage. Many people would not know this accurately. Therefore, the system should be assistance the user through allowing for approximate values and visual elements to show what they look like.

From the beginning of the process of the development of the application, the main business requirements will be outlined using various reviewed iterations of prototypes, which will be set through the collaboration of people of different background alongside myself for ideas regarding the application. This would include people from professional background to the casual user. After each stage of the development of the application, I would return to the same user about the prototype built to get their review, use the review to modify, repeat until both parties are satisfied with the efforts.

The methodology of feature driven development, agile and prototyping will be employed for the duration of the application development. Once the prototypes have been completed and reviewed through tests and evaluation, they will integrate into the system gracefully.

1.3. Project Aims and Objectives

The aim of developing NaDMA is to allow the average user to ensure they are getting enough macronutrients they need to live a comfortable and healthy lifestyle. This would be through a UI designed for a simple yet intuitive UX for the users to grasp easily.

There are a few objectives to the aim. At the start, the requirements must be gathered for the application. Subsections incorporate user requirements, business requirements and technical requirements. User requirements involve getting the users' story, their reason for potential using the application and what they expect from it. This would mean spending between 5-6 hours taking to different people about the application area combined with following up on ensuring it is what they are looking for. Once that is completed, a draft of all the business requirements is produced. This would display the mandatory requirements, optional requirements and out-of-scope requirements. This itself would take about an hour itself to complete.

The technical requirements would be investigated last. This would involve the research of similar solutions to the area I am tackling, the available modern technologies and tools, other projects completed and other areas I would need to research, such as datasets needed etc. This would take approximate 10-12 hours as both the tools and methods need to match the business requirements as basic specification.

After the gathering the requirements, the next objective would be the prototyping process. The first would be the planning of the prototype. An analysis of the different software methodologies is required to understand which match the project itself. This could be a singular methodology or a combination of multiple methodologies. This needs to be completed within 2 – 3 hours itself. Once selected, focus would shift to the overview of the system. The plan for how the application is envisioned is plotted and documented, from the specification technology used to the design pattern and system architecture we are working with. This would take between 2 – 6 hours. Following this would be the full stack prototypes. From use cases, low-fide prototypes etc, of the front-end all the way to ERD, class diagrams of the backend. Flow charts would be included to demonstrate the middleware behaviour (unless a more appropriate method is found proven to be more reliable). This would take at least 10 hours to cover all the aspects of the system.

Once the design has been reviewed and approved, a proper prototype using the specification details will be implemented. This is to ensure the areas work properly, the technology behave as expected and the user agrees with both the design and the UX. If there are any issues with the prototype, the documentation can be reviewed and modified to suit what is mutually agreed on. At least 10 hours would be spent to complete this task.

Following this objective, the decision-making on the testing and evaluation would be made, such as the different software required and the different methodologies. This would take about 2 hours as they must work with the chosen technology, making the scope narrow. Risks with the system design must be understood at this time and catered into the schedule.

The final objective is the system implementation. This must be reviewed in intervals to ensure it matches the requirements and design document. This would take between 2-3 months minimum. It would also need both integrated testing in ongoing phases. After each part and the end of the development phase(s), both manual testing and evaluation from peers would be obtained. This part should be about 30 mins in total per person. Once every objective has been met, the aim should be completed.

1.4. Project Requirements:

User Requirements

As part of the project, several people whom are potential users were spoken to for gathering requirements. A use case of three personalities were drafted up from the user requirements whom, based off their feedback, were the mutually distinct in their requirements of the application. They are as follows:

User 1

Background information

User 1 is a 22-year-old white male student who is currently undergoing a college degree in computer science. Aspiring to be a game engineer, User 1 has been a vegetarian for eight years and switching due to a dislike in taste in processed products. Some of the personal interests involve either individual or social activities associated with art and games.

Expectations from the application

User 1 reason for using the application would ensure the diet is followed through correctly. This would involve the application understanding the diet and filtering out the unnecessary options in a visual aesthetic, graphical and simple user experience. When he is logging his diet or inputting his personal details, it should be easy to complete and access. Among User 1 expectations are accurate information displaying only vegetarian options.

Backstory and reasons for using the application

User 1 would often need assistance and reminders to ensure the diet is optimised for his needs. However, it would be a waste of his time if he is bombarded with meat products inside the application or the application is completely inaccurate.

Due to his disabilities, the application must be polished and smooth, yet easy and adjustable to cater to his needs. Since he is in college, the context must sweet and short. User 1 has requested for a colour scheme as part of the feedback.

User 2

Background information

User 2 is a 48-year-old white female student who is currently a mentor and coacher of people who work with disabilities, resulting in vast experience. She also has kids with disabilities and worked with adults of various disabilities. Her interests lie in the fields of family, meditation and harmony.

Expectations from the application

User 2has advised on several functionality that would allow people with disabilities or carers to efficiently use the application. One is access to the camera to scan in the diets rather than inputting it manually combined with the variety of diet choices. Another is importing a system that allows blind people to use the application. The usability of the application must be simple yet intuitive. Some favourable, optional choice are the ability to download the logged diet and display social events with people of the same interests.

Backstory and reasons for using the application

User 2 reasoning for the application is both personally and universally. Had she gotten access to this application during the time she has worked with adults with disabilities, it would had cut down on both the short term and long term on managing the diet, especially if it came with a scanner. Many careers in this position would find it difficult to manage manually logging the diet and care for the disabled.

She is also aware of people who are blind that be interested in this application if they could use it. Since their interests varies different options should be available to them. She also jointly takes care of a son whom she knows would really benefit from this application if designed to his needs. Ideally getting a soft copy of the logs would be nice.

User 3

<u>Background information</u>

User 3 is a 51-year-old white male whom does part time caretaking at the local GAA club and is primary minder of his kids. He has a personalised omnivore diet, which assists him in his fitness and sporty areas of interest. His previous experiences involve being of club for group of kids with disabilities. User 3 has been diagnosed with diabetes and interested in methods to prevent that happening to his family.

<u>Expectations from the application</u>

User 3 wants to have full control in being able to set his own schedule for the diet of choice, which would assist him greatly in his life goals in the area. The one thing that helps is a way to motivate himself through challenges and awards, which he also wants full control over. The application should provide a description of the products he is consuming, including their meaning.

<u>Backstory and reasoning for using the application</u>

For twenty years, User 3 has been into fitness and proper dieting. This resulted from previously being on a shocking diet leading to the diagnose of diabetes. Since the diagnose, he has going through intervals to understand what the correct way is to diet properly, such as consummation of products and their ingredient list meaning.

Continuing this journey, User 3 wants to ensure he is continually motivated. As a result, he may want to change his diet to cater to the goals, which he expects the application to cater to.

Business Requirements

After phase of gathering the user requirements, the collection of the business requirements are as follows:

Business Feature Requirements	**Description**	**Priority**	**Scope Area**
User login	Allow user to login through username and password	High	Mandatory
User Register Account	Allow the user to register for the system	High	Mandatory
User logout	Allow the user to leave the application gracefully	High	Mandatory
View profile	Allow the user to see their details	High	Mandatory
Modify Account	Allow the user to update their details	High	Mandatory
Import / Login / Register using an external application	Allow the user to login using details from social media and / or Fitness / Nutrition Apps, such as Facebook or Fitbit	Low	Out-of-scope

Simple Navigation UI	Allow the user to access the different parts of the application smoothly	High	Mandatory
Diet Logging System	Allow the user to log their diet into the application through various ways.	High	Mandatory
Download logged diet schedule	Allow the capability for the user to download the diet that has been logged.	Medium	Optional
Simple, intuitive Graphical UI	Simple UI to allow the user to log their diet into the application, using buttons and Imagery as opposed to heavy textual information. This would resort to the user using a search bar to filter out the specific dish they had (such as pepperoni pizza as example). Once this has been selected, the user would view the default options of food and ingredients are used before having the option to accept, accept and modify or return to search. This would allow the user to input either homemade dishes or take-aways. This would incorporate the ability to set the fields from previous inputs (reusing dishes)	High	Mandatory
Template for diet	Allow the user to decide their daily diet as to their needs, such as breakfast, lunch and dinner or brunch, dinner and supper as examples	High	Mandatory
Scanner for recipe input	Using the camera to scan the barcode to get the ingredients	Medium	Optional
Advisor / Recommender System	System to advise the user based off their inputs	High	Mandatory
Optimal UI / UX Experience of the advisor system	Use graphs and imagery to assist in advising the user	Medium	Optional
Daily trends	Display the user their daily input and advise on what to do	High	Mandatory
Display Weekly trends	Display the weekly versions of the daily input and advise on what to do	High	Mandatory
Display Monthly trends	Display the monthly versions of the daily input and advise on what to do	Medium	Optional
Display Yearly trends	Display the yearly versions of the daily input and advise on what to do	Low	Out-of-scope
Breakdown of the nutritional input	Display in lay terms what the user are eating and how it impacts their body	High	Mandatory
Cater to user goals and macronutrients nutrition deficiencies	Implement different categorical solutions depending on both the goal of the user (ie lose weight) and their nutrition deficiencies (ie celiac)	High	Mandatory
Suggest alternatives for diet	Provide graphical solutions in areas where diet could improve	Medium	Optional
Colour Scheme	A colouring scheme to alert the user how their diet is	Low	Out-of-scope
Effective Diet Scheduler Advise	Advise on how to diet effectively, from the periods of when you eat to how much you eat at each interval	Low	Out-of-scope
Notification / Alert System	Notify the user to use the application, whether it is to log their breakfast or to check their dietary analysis	Medium	Optional

Product Label Description UI	Provide a breakdown of how to effectively read the ingredient list of products bought from the shops using Graphical UI	Low	Out-of-scope
Food Pyramid Interactive UI	Allow the user to find out key details about proven healthy diets, such as vegetarian, using an interactive food pyramid. This would incorporate a breakdown of what to eat regularly and examples of each (ie vegetables)	Low	Out-of-scope
Disclaimer within application	Ensure the user understands the application has not been reviewed for ethical standards and therefore cannot be taken seriously	High	Mandatory
Temporary storage and usage	Enable the user to access the features without the need for the internet	Low	Out-of-scope
Challenge and Reward System	Enable the user to either have computer generated with a goal in mind or allow the user to create one themselves. This would be catered with the advisor system to ensure the user reaches their end goal regarding their nutritional and dietary needs.	Low	Out-of-scope
Assistive Technology Systems	This would enable people with various disabilities to be able to use the application, such as the blind people etc. Methods, such as importing the needed system or deriving from them, would be used here	low	Out-of-scope

The requirements labelled “Mandatory” are necessary to complete within the timeframe scope of the application dating from the 16th September to 2nd April. The “Optional” fields are extensions that may be completed within the scope provided there is time to accommodate them. The “Out-of-scope” fields are areas in which we know are impossible to do given the timeframe and so are outside the scope of the application.

1.5. Thesis Roadmap

One sentence explaining what each of the following chapters is about.

Chapter 2 – Literature review

Literature Review Chapter will delve into the research in the area relating to the common person nutrition and diet knowledge, industry technology in the area, student take on the area and technologies which are viable for my approach. If other research outside these criteria are found, they will be included too.

Prototype design

The chapter describes the choices for the application. This includes a high-level abstraction of the architecture for the application and the methodology required. After completing this, design prototypes such as uses cases and class diagrams will also be drafted.

Prototype Development

The description of how the system design choices were implemented as part of the prototyping process combined with some unexpected encounters that were met.

Testing and evaluation

The draft of decision taking that involves the draft up of key ideas and plans to ensure the application behave as we design it to do.

Issues and Future Work

A discussion of areas where it could possibly go wrong with the application and other possible work which could be implemented outside the current scope of the application lifecycle.

2. Literature Review

2.1. Introduction

In this chapter, research into areas related to nutrition and dietary needs is the priority. These areas include the different industry solutions, different types of technology applicable to domain area, other useful strategies or research relating to the project aim and studying technical solutions provided by other college students relating to the domain of interest. How the potential solution would be applicable to the target user would also be explored here too.

2.2. Alternative Existing Solutions to Your Problem

Two alternative solutions researched into were “Fitbit” and “mySugr”. Both were found on the “Play Store” for the Android devices and so are mobile applications. When assessing the applications, a comparison between the goal of the team behind the application and the app’s functionality was conducted combined with evaluation of the “UX”, “Design” and “Ease of use” for overall critical review.

Fitbit

According to Play Store Application specification, Fitbit **(5)** “is dedicated to helping people lead healthier, more active lives”. The application main functionality is logging key information, such as the diet, water intake, exercise tracking and a weight goal. Through all these activities, Fitbit assists the user in monitoring the physical condition for overall improved health, which incorporates sleep, eat, exercise and repeat. Fitbit also monitors the heart rate through syncing with multiple external devices, such as Fitbit watch, designed to track such details.

The overall design of the mobile application is clean and smooth. However, it is not easy to use. Parts of the design interface do not follow conventional operational standards in its attempt to be unique, such as clicking on the user profile picture to access the navigational tool. Another aspect is the logging of the data, designed as part of feature usage specification. Because in-depth knowledge of the input such as calories is part of the requirements, the standard of ease of use would be reduced heavily.

This would make the user experience moderate to a disappointment for the average user, as most would not have the knowledge at their depth. The application is catered to specialists or team for an athlete as the typical user.

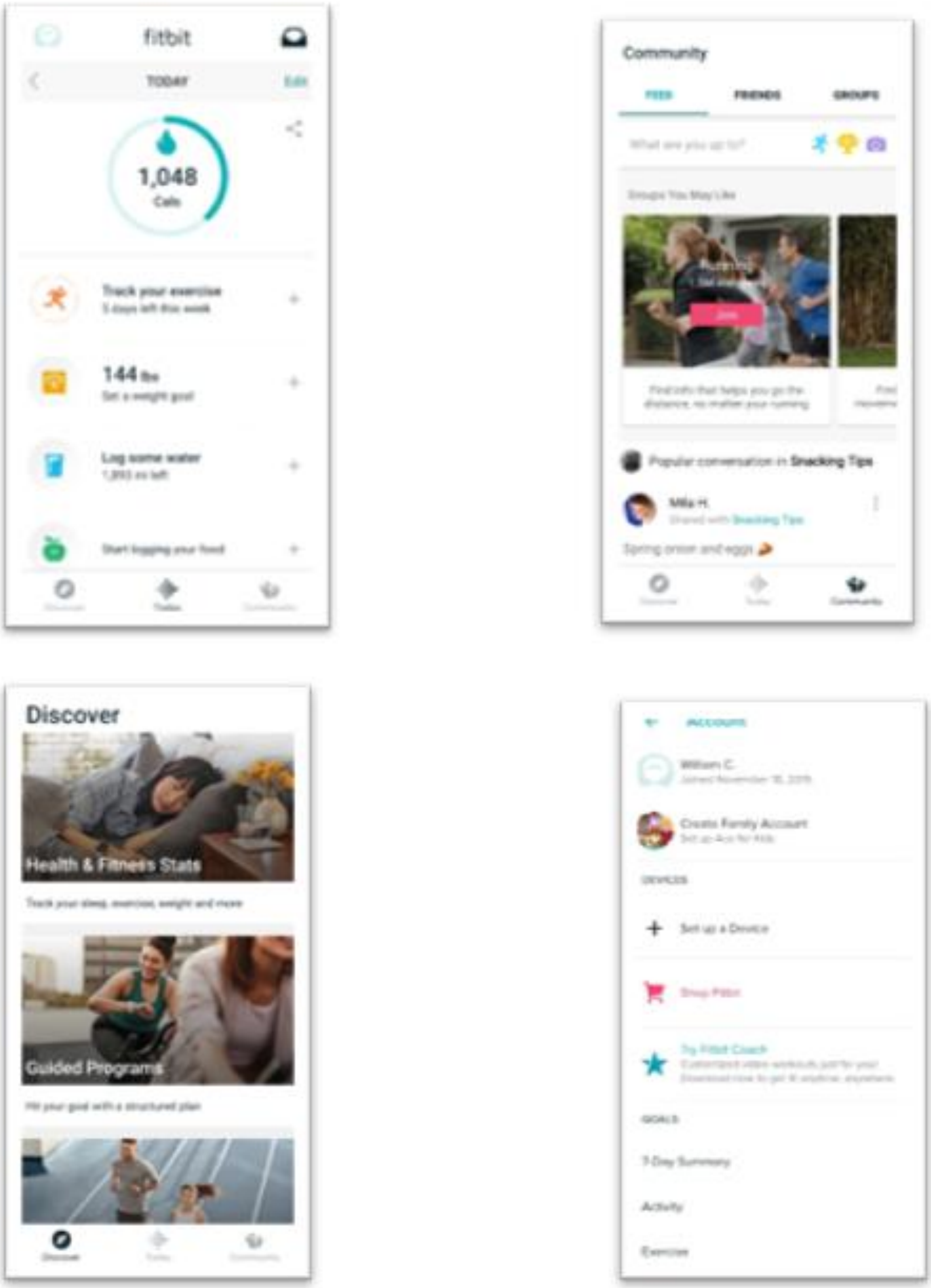


Figure 1 - Fitbit UI

mySugr

An app “to manage your diabetes and HbA1c”. It was ranked **(6)** “the top diabetes app by Healthline 3 times”. The functionalities of the application are access to easy and personalised dashboard (including diets, meds, carb intake, meals, blood, glucose etc), clear blood sugar level graphs, estimate HbA1c, motivating challenges and feedback, medical analysis (daily, weekly and monthly), detailed reports for the doctor and secure data backup, which incorporates regulatory compliance, quality and safety for the user.

The design of the application is simple yet smooth and intuitive. The graphical element grabs the user attention to return and keep using the application, while at the same time using familiar toolkits and standards for minimal learning curve. The ease of use is at medium level, as the complex area of the application is the logging of diabetes. This requires abundant knowledge and insight into the diabetes domain. Otherwise it is simply enough for anyone necessarily needing it.

The overall UX for the application is only a level higher than the expected standard. From a design and ease of use, the UX is pleasant for anyone to use. The complexity area, which is the logging aspect of the application, is the only critic area of the UX and area to improve on.

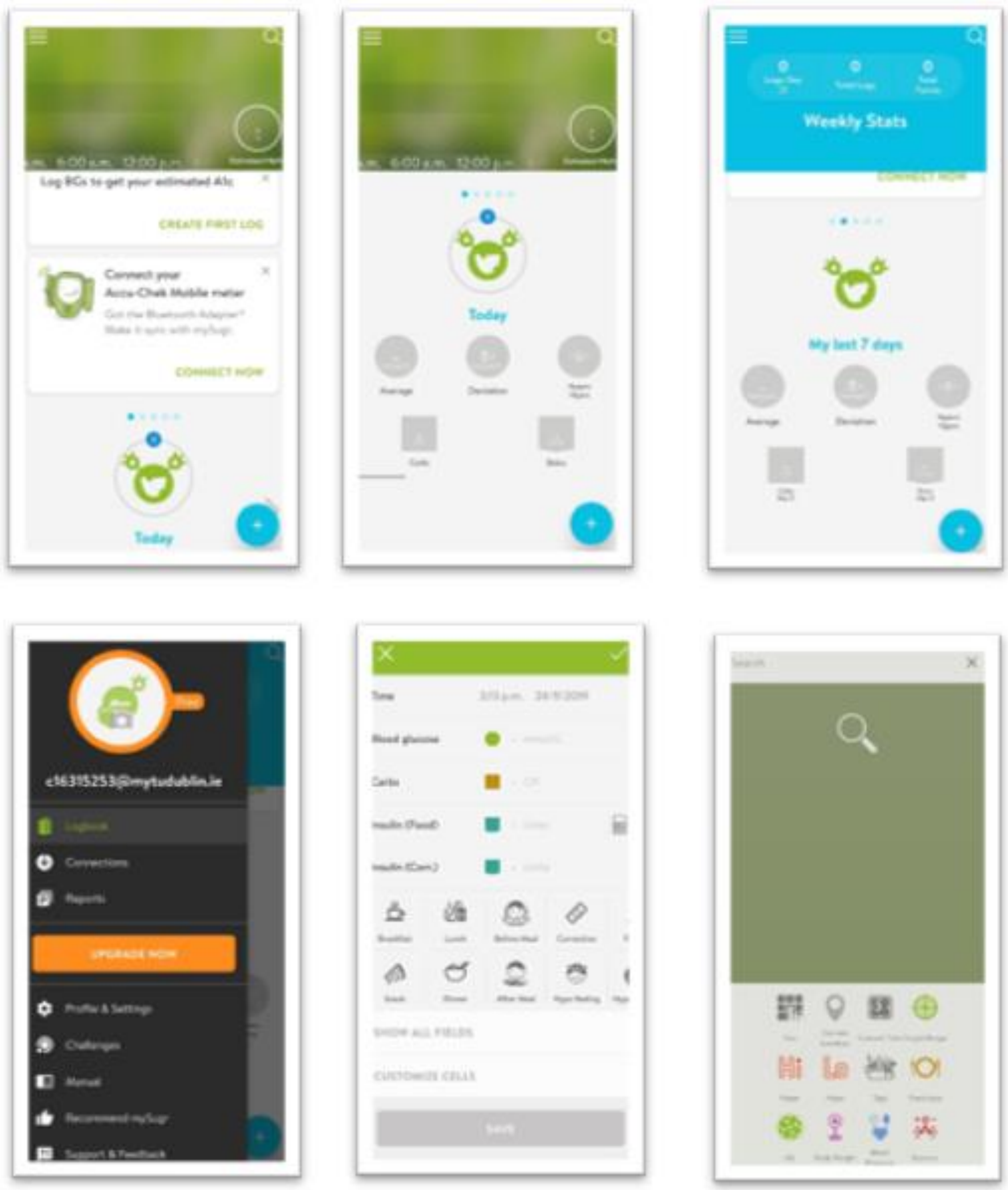


Figure 2 - MySugr Overview Layout

Overall Evaluation of Industry Mobile Applications

The areas to be covered for overall evaluation are UX, functionality, design and ease of use. Regarding mobile applications in the nutritional fields the evaluation varies. The functionality requirements for the application, such as fitness or nutritional deficiencies, have been met for the user.

The design varies. In some applications, conventional standards are followed, which minimises learnability simultaneously with maximising usability. However, the learning curve are not low for someone who never used apps like it before, which may put off non-technical proficient users.

Ease of use, regarding the core aspect of the applications, are moderate to low. This is due to the necessary in-depth knowledge of the area the app was designed for, such as logging nutritional data, which is the heart of the application. These specifications are not something ordinary people would have at hand, which would hinder the overall user experience. All other design decisions are either easily learnable or expected to be known beforehand.

The UX overall, because of the dependency of the logging of the data, is moderate and requires patience as specific measurements are needed for the app optimising.

Conclusion

The application functionality and design are universally catered towards specific user group. They do not expand the UX to generic users. As a result, the focus of the application for the project should be the UI design to allow for generic input rather than specific. This would improve the UX and ease of use combined with giving a feature to make the application stand out.

2.3. Technologies you've researched

Mobile Technologies and their integrated development environment

There were a variety of mobile technologies that were investigated for the purpose of the project. As each were delved into, the questions of how they would fit the requirements all the requirements of the application were the main priority. These were the technical requirements, the businesses requirements and the user requirements simultaneously. By research into the popular hybrid mobile applications by the developing community, an outline of the four most popular technologies, PhoneGap IDE, Ionic, React Native and Xamarin, will be employed.

PhoneGap

By wording of Orion Info Solutions Website**, (7)** "PhoneGap is open source freely available app which can be run on different platforms. There is rapid increase in these types of apps as they are easy to maintain and save both time and money." Because applications created using PhoneGap are developed using modern web technologies (HTML5, CSS3, JS libraries), they provide "easy testing and maintenance" to the community and allow for usage without learning any "additional skills". They also support "multiple platforms such as Android, iOS, Window" through providing software that are "easy to access".

Objectively, downsides are part of the application. Due to knowledge "PhoneGap Apps are very poor in performance, they are not recommended for the gaming technologies as compared to native apps". They also fail "in providing the access and control to the user" due to their "slow processing". The same apps "become inefficient" when working with the same "native apps". As a result of such design decisions, the layout of PhoneGap apps are not "as good as native apps".

PhoneGap Development
HTML
CSS
JS
PhoneGap
orion
infosolutions

Figure 3 - PhoneGap Development

Ionic

Through insight of the Altexsoft Website, Ionic was created **(8)** “2013 as an open-source SDK for hybrid mobile applications” and “now has more than 5 million apps built using it”. The reasoning behind the numbers is the “strong [growing] community” and the “Concise documentation” built with the application. An area where its best known for is “providing platform-specific UI elements … for iOS and Android”, which allows for “quick prototyping”.

Some areas of weakness for Ionic starts with “Absence of hot reloading”, a feature which allows developers to make a change to the system which changes the layout in real time. This combined with the design of being a “Plugin-dependent system” makes any application designed by Ionic have potential “security issues”. The final issue, which had a significant part in the decision making, is the “performance is lacking [when compared] with native applications”.

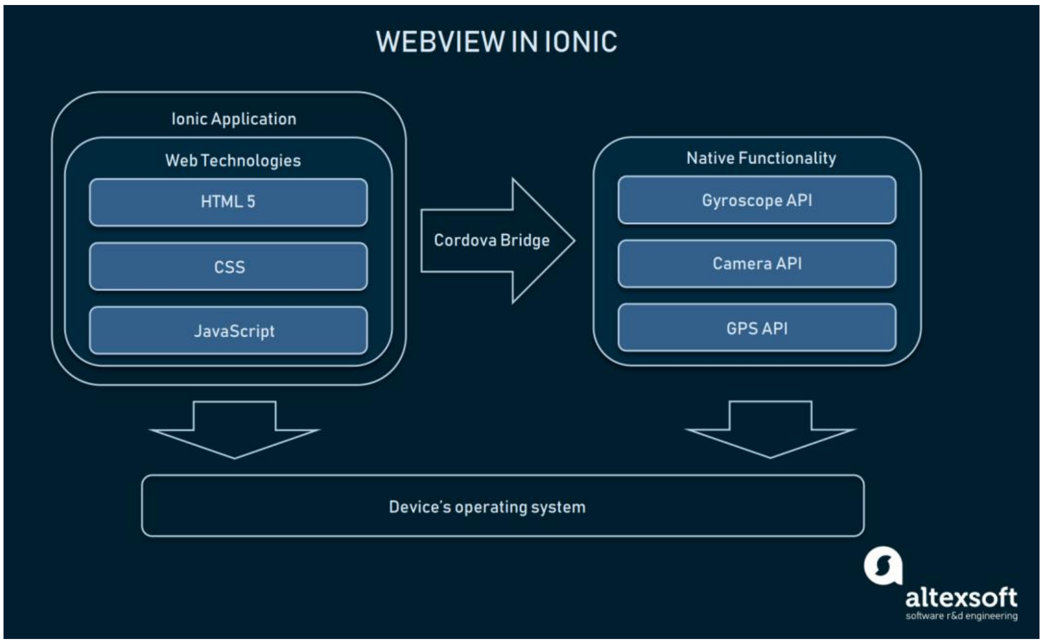


Figure 4 - Ionic Webview

React Native

The Altexsoft Website states React Native is **(9)** “a hybrid mobile-app development framework for iOS and Android.” Built using the tools of ReactJS, “a JavaScript library that [uses] the speed of JavaScript” to make applications “highly dynamic and responsive to user input “, React Native combines native application development with JavaScript UI development. React Native has been created by Facebook and is open source for growth of community developers. The speed of the development comes from the usage of the “Virtual DOM in ReactJS”. This ensures both” user experience” and “developer’s work” are improved simultaneously.

Cons of the application are as follows, the tighting of coupling of business logic is an issue when system’s design find “HTML in [the] JavaScript”, making it difficult for decoupling. When comparing the “High pace of development” vs the pace of updating “Documentation”, it is found to be a major disappointment as it is not completed concurrently. Thus, it is not surprising the last major issue is the “Lagging SDK Updates”.

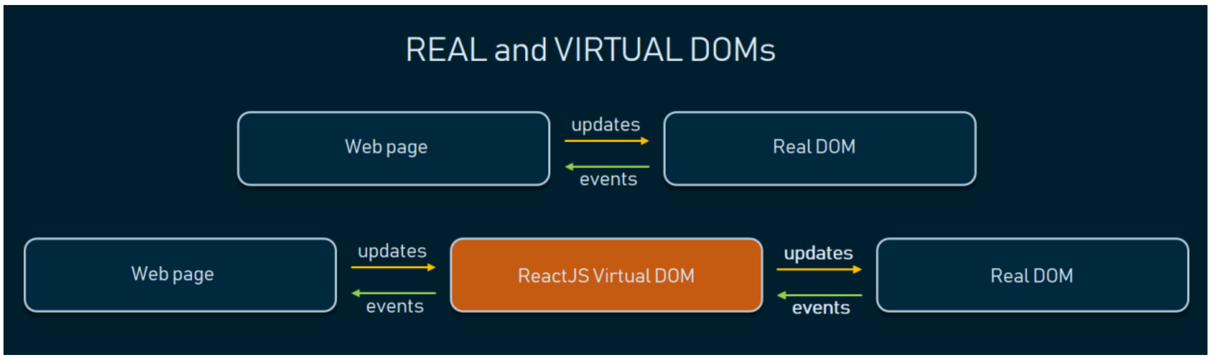


Figure 5 - React Native

Xamarin

The last tool being covered today, also evaluated by the Altexsoft Website, is Xamarin. **(10)** "Xamarin is a tool used for cross-platform mobile app development that allows engineers to share about 90 percent of code across major platforms". "It is based on the Microsoft technology stack and already has a community of over 1.4 million developers". Being "Open Source Technology with Strong Corporate Support" gives Xamarin a competitive advantage when compared with some of the other IDEs, such as receiving "Full Hardware Support" from the developing community and private institutions, which grants "Simplified Maintenance". Such efforts ensure applications developed from Xamarin are provided "Performance Close to Native" applications developed.

Imperfections and Drawbacks from Xamarin can be drawn. Due to the corporate support, there is a "High Cost [of development] for Professional and Enterprise Use" and a strict guideline of the required language needed. Applications derived from Xamarin ide usually are a "Larger App Size". These drawbacks result in "Basic Knowledge of Native Languages" are required to build such applications combining with "Compatibility Issues with Third-Party Libraries and Tools", due to mistrust of larger cooperation's from the potential of hackers.

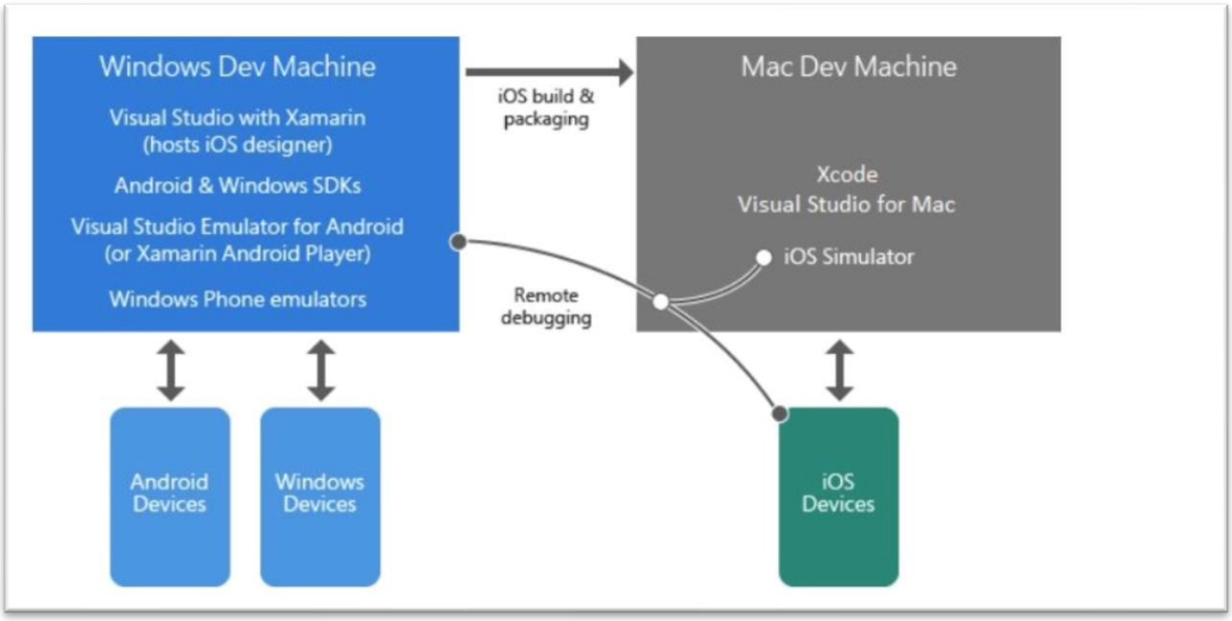


Figure 6 - Xamarin Dev Machine

Middleware technology

Described by Techopedia website, **(11)** “The .NET framework is a software development framework from Microsoft. It provides a controlled programming environment where software can be developed, installed and executed on Windows-based operating systems”.

The main design principles behind the framework are Interoperability (allows programs functionalities to be accessed outside .NET), Common Runtime Engine, Language Independence, Base Class Library, Ease of Deployment (ensure the ease of installing programs without interfering with previously installed applications) and Security. Because of these features, .Net framework has been employed on many applications to allow separation of concerns between the different areas of the applications. These include front-end to backend communication.

Figure 7 - .Net Framewor logo

Cloud services

All cloud services provide solutions including **Infrastructure as a Service** (IaaS), **Platform as a Service** (PaaS), and **Software as a Service** (SaaS) that can be used for services such as analytics, virtual computing, storage, networking etc. Such services allow for remote connections from anywhere in the world, as one main aspect of cloud is it resides on the world wide web. The three competitive public examples in the present moment are Azure Microsoft Cloud services, Amazon Web Services (AWS) and Google Cloud Platform Services. All three would provide the same role for both remote services and storage purposes. The only significant difference between each is the producer of the servicers, with Microsoft hosting Azure, Google with its own and Amazon with its web services.

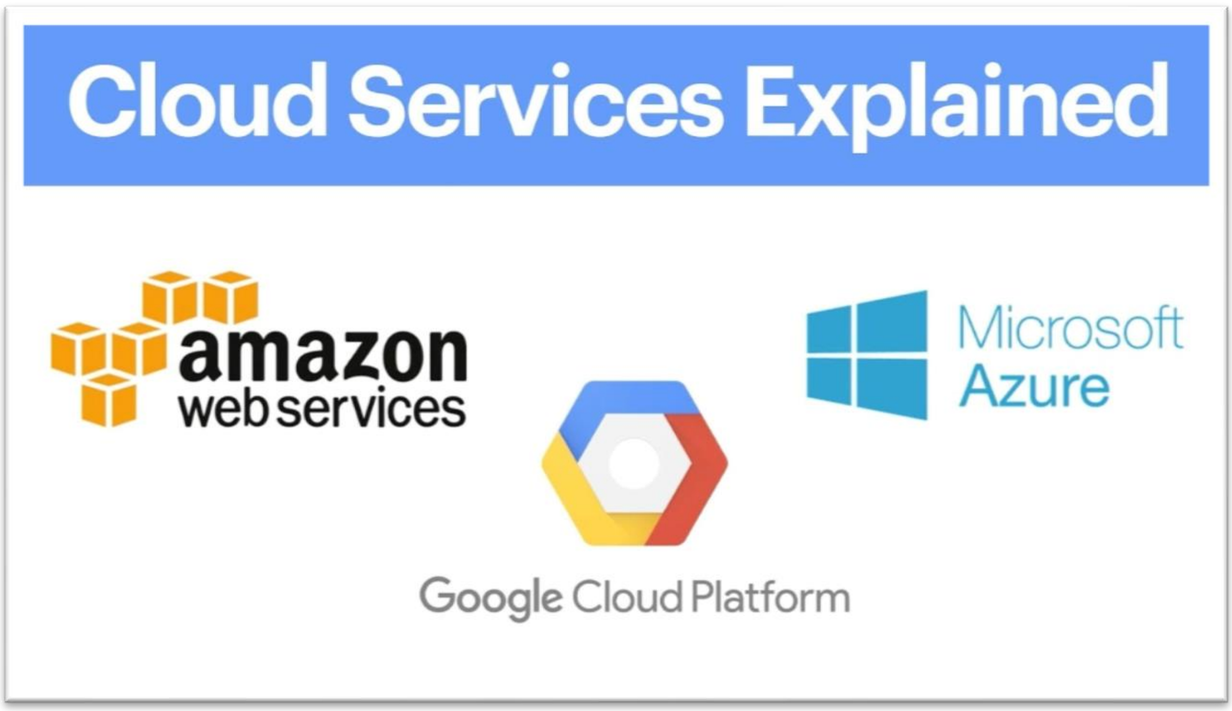


Figure 8 - Cloud Services Explained

Backend Database technology

All the cloud services offer SQL relational database and servers for remote storage, which will be taken with delight and used. For local storage, SQLLite will be used as it is most convenient for mobile application development for minor purposes. The bulk of storage will be completed by the remote storage.

Operating Systems

Android

According to lifeWire **(12)**, “Android is a popular, Linux-based mobile phone operating system developed by Google”. It is an open source project which Google provides to various device manufacturers for free. From Huawei to Samsung, Android is used in each device development and is maintained to adhere to the different specifications. This would allow for diverse use of the phone, providing different phone user experiences. However, as a result is very difficult to keep updated against the different risks against the devices.

Figure 9 - Android Logo

iOS

Recombu states **(13)** “iOS is the mobile operating system that runs on Apple’s mobile devices, i.e. iPhones and iPads. It’s the main software that allows you to interact with your Apple phone or tablet”. While Android is versatile with the devices employed, iOS is restricted to apple products and apple software only. This makes the design more maintainable due to the company able to design the two together simultaneously while making the application safer, when compared to Android, by preventing downloads from Third-libraries sources.

Figure 10 - iOS logo

Programming Languages

C#

When reviewing **(14)** “Geeks for Geeks Website”, information found about C# are describing the language as “a general-purpose, modern and object-oriented programming language”. The language was developed by a Microsoft team within the *.NET* initiative. This team led by Anders Hejlsberg. The language was approved by the European Computer Manufacturers Association (ECMA) and International Standards Organization (ISO). C# is similar to Java syntactically and easy for users who have knowledge of C, C++ or Java.

Scripting Languages

Python

Another language reviewed under **(15)** “Geeks for Geeks”, Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Other Software and Tools

GitHub

GitHub is a git repository hosting service and version control system. By managing different git repositories, it can manage different versions files and systems through individual or collaboratively efforts. The main features are to allow users to push, pull and merge different applications versions, making it a powerful tool in managing any type of projects developed in iterative steps.

Figure 11 - Github logo

2.4. Other Research you've done

Usability

Usability Theory is the concept on human interaction with technology through the means of "effectiveness, efficiency, and satisfaction" **(16)**. This should ensure the system is "easy to learn and remember, efficient, visually pleasing and fun to use; and quick to recover from errors". From the same studies it is shown how people remember usage of systems combined with processing certain images.

Assistive Technology

Assistive Technology is the use of technology to assist people with certain disabilities. These would include vision issues, such as the blind, and providing them with appropriate solution, such as text-to-speech. Another example would be those with mobility issues and providing them with similar solutions like voice recognition. **(17).**

Nielsen's Heuristics

An evaluation method for systems that have heavy usage on interfaces. While on the older side, the 10 evaluation methods are still applicable for modern applications containing heavy GUI elements usage. Such approach will be kept in mind for evaluation of similar technologies and the development of the application. **(18)**.

Modern Mobile usability

As mobile applications availability increases to the people, the application's requirements need to cater to ensure the user is engaged correctly. This is through ensuring the user can complete their tasks in the expected optimal time by providing the following six methods from the source website: Platform Usability, Provide Value Right Away, Simple Navigation, Clear & Concise Content, Minimize the Number of Steps and Reduce Scrolling methodologies. **(19)**

2.5. Existing Final Year Projects

Project 1
Title: Proactive Order Management System

Student: Stephen Fox

Description (brief):
An application that allowed businesses to handle orders processes by analysing its data within the system. This system also provides businesses with information on how and when to process these orders. The order processes are placed into the system remotely through customers using a mobile application that can access and connect to the host web system.

What was complex about this project?
The task Scheduler manager was the most complex part of the system as many uncertain fields that could change that this system needed to calculate in order to be optimised for the overall system.

What technical architecture was used?
A Client – Server application with the tech tools iOS Application, Web Application and AngularJS acting as the client-side while Node.js, NuPIC, Proactive Module - Flask, Google Map Distance Matrix API and MongoDB as the server side

Explain the key strengths and weakness as you see it
To have an algorithm that dynamically creates task handlers based off the tasks requirements is a strength as it allows you to expand your resources and minimise data wastage at the same time.

The data involved does not seem to be protected through encryption or other methods, which could allow nearby people who could steal the data to view the data, is a major weakness of the application.

Project 2
Title: Glucose Coach

Student: Alex Kiernan

Description (brief):
An application designed to track people whom have type 1 diabetes to better manage their overall health by logging their blood sugar levels, their diet and their physical exertion into the coaching system. Once logged, the application will be able to provide the necessary suggestions catered to the individual user to enable them to better manage their glucose levels.

What is complex about this project?
The machine learning part of the overall system was the main complexity as research was needed to be conducted combined with the development of the sound process in order to properly process the user information into the system and export back accurate results to the user.

What technical Architecture was used?
A Client – Server architecture was used with the RESTful service links between the two, a remote relational database, a flask server and the machine learning system scikit-learn as server side while the use of a mobile application as the client.

Explain the key strengths and weaknesses as you see it
A key strength of the application was at the time of development, the constant monitoring of the users progress with the insulin intake was unique to the system design which provides it a competitive edge compared with similar applications

A weakness of the application would be the lack of knowledge provided back to the users whom used this application. If there were graphs that displayed over time the use of insulin intake over days and weeks, it would have made the user understand their blood sugars levels more in depth, which would have allowed them to make more informed decisions.

2.6. Conclusions

The main requirements for the use of the application are designed for the ease of use, the user experience and key polished functionality of the application. As a result of these requirements, along with the user requirements and business requirements, the technologies chosen were using Xamarin for the front-end, the .Net framework for the middleware, Azure Cloud services with for remote service and SQLite for local storage.

These technologies are best suited for the development of the application through their adaptability to the users’ requirements, the developer’s technical needs and the General User Interface requirements.

3. Prototype Design

3.1 Introduction

This chapter provides the insight into abstract layers of the application and the route of its development. An overview of how the technology researched in chapter two will be provided and analysed against the requirements of the applicable. This insight includes specifications sourced from the system's architecture to the full stack development and the applicable chosen methodology.

3.2. Software Methodology

Feature Driven Development

This methodology is one of the few adaptive methodologies which focuses on five short iterations and steps. Each iteration lasts approximately two weeks. The first three covered at the start are to "Develop an Overall Model", "Build a Features List" and "Plan by Feature" **(20)**. The last two covered at the end are to Build by Feature and Design by Feature.

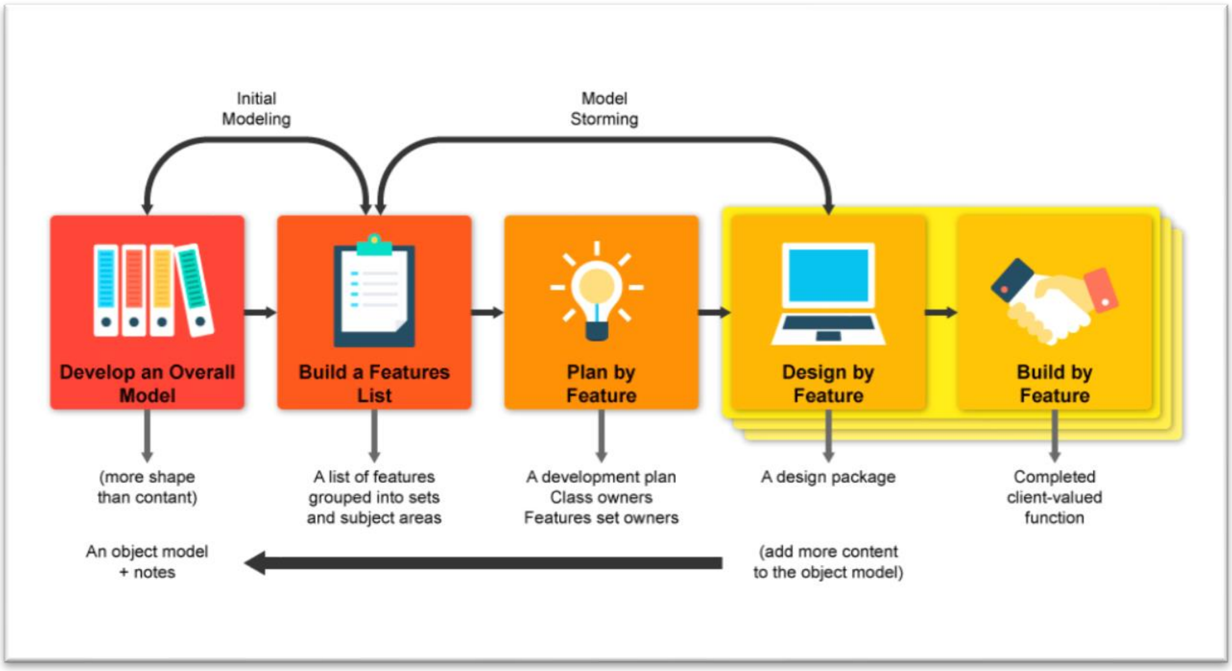

Initial Modeling
Model Storming
Develop an Overall Model
Build a Features List
Plan by Feature
Design by Feature
Build by Feature
(more shape than contant)
A list of features grouped into sets and subject areas
A development plan Class owners Features set owners
A design package
Completed client-valued function
An object model + notes
(add more content to the object model)

Figure 12 - FDD Methodology

Agile

This methodology is another adaptive methodology being looked. Created as the alternative to the waterfall method, this involves both the potential users and the stakeholders in closer range for influence on the project itself. Key aspects of the project include ensuring the user is actively involved in the process, the team is enabled to make independent decisions, allow adaptable requirements in a fixed timeframe, enable the capturing of the requirements at the high level, iteration over small developed releases, frequent delivery of products, completion of the feature before moving on , applying the 80/20 rule **(23)**, provide room for rigours integrated tests and clear comprehensive, collaborative & cooperative approach between all stakeholders. **(21).**

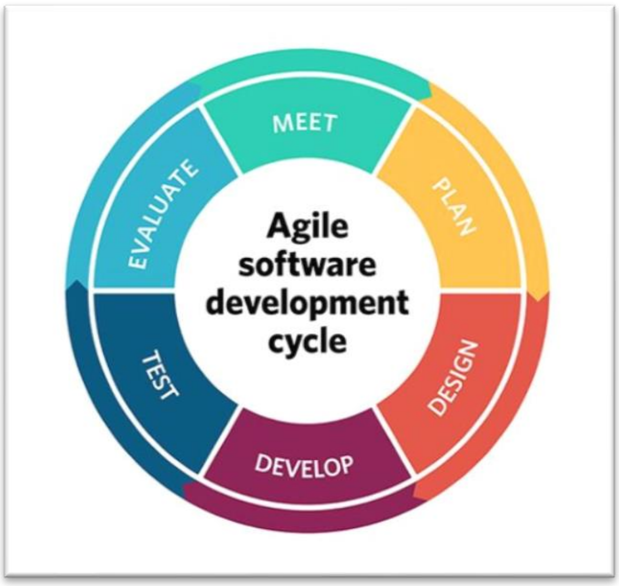


Figure 13 - Agile Methdology

Prototyping

This methodology is used when attempting to create a solution to a proof of concept or a business requirement. It is normally adapted, developed, redefined and refined until the requirements have been gathered and understood to be correct. Once that happens, the final version will expand on the prototype in its implementation. **(22).**

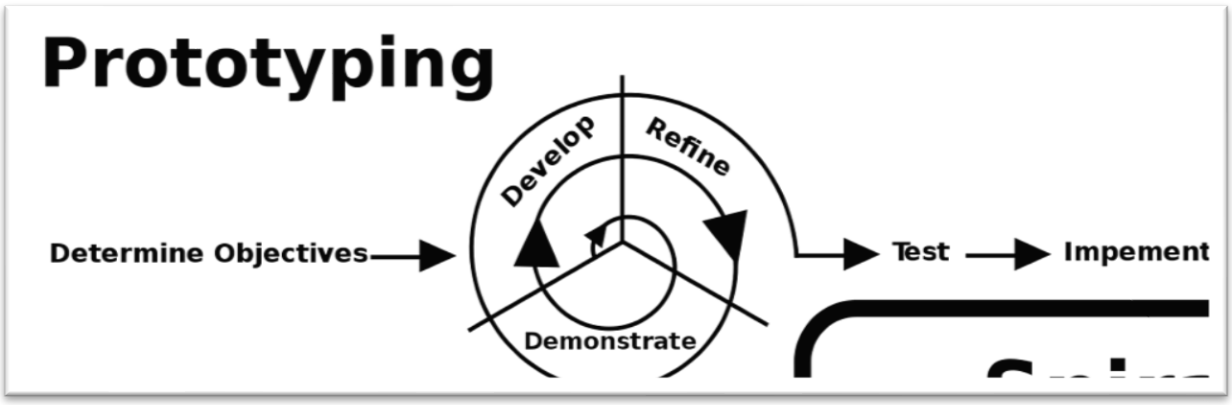


Figure 14 - Prototyping Methodology

Agile Feature Prototyping

This is the methodology chosen and is created from each part of the researched methodology discussed. The methodology starts out like the Feature Driven Development methodology, where it captures the overall model and gathers the features as the first two steps at the start. Once completing that, it goes through iterations of sprints, like the agile methodology. Within each sprint, it would plan out each feature with the user for design purposes, which is followed by prototyping process and then returns to the planning for review. Since this is a circular motion, any part of the step can return to the source it came from for review, which is an aspect of agile during sprints. Since prototyping is included as this step, this is where prototyping methodology is used. The curricular can be ran through a few times before it is satisfied with the feature result before deploying it in increments. The next feature is worked on then.

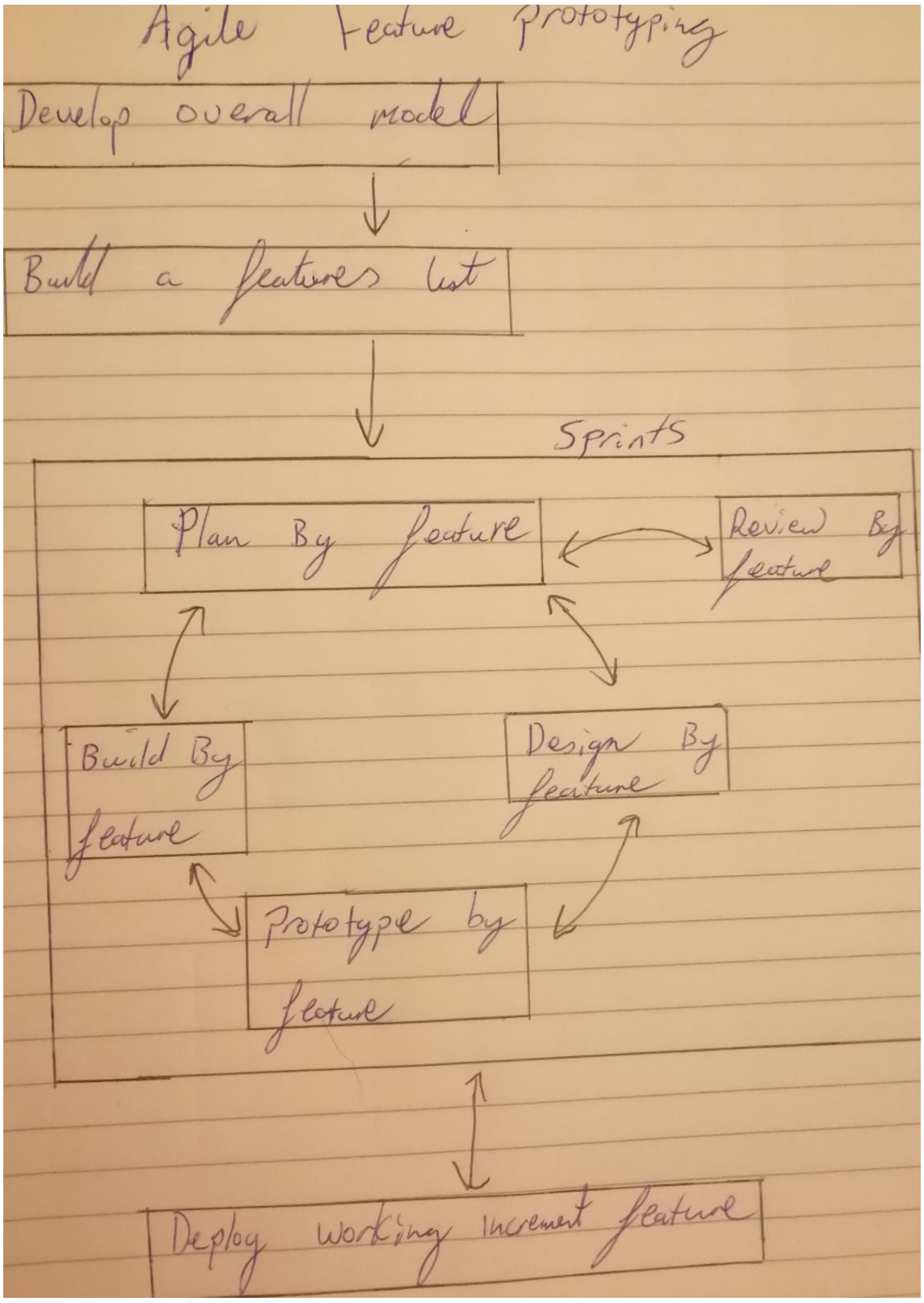
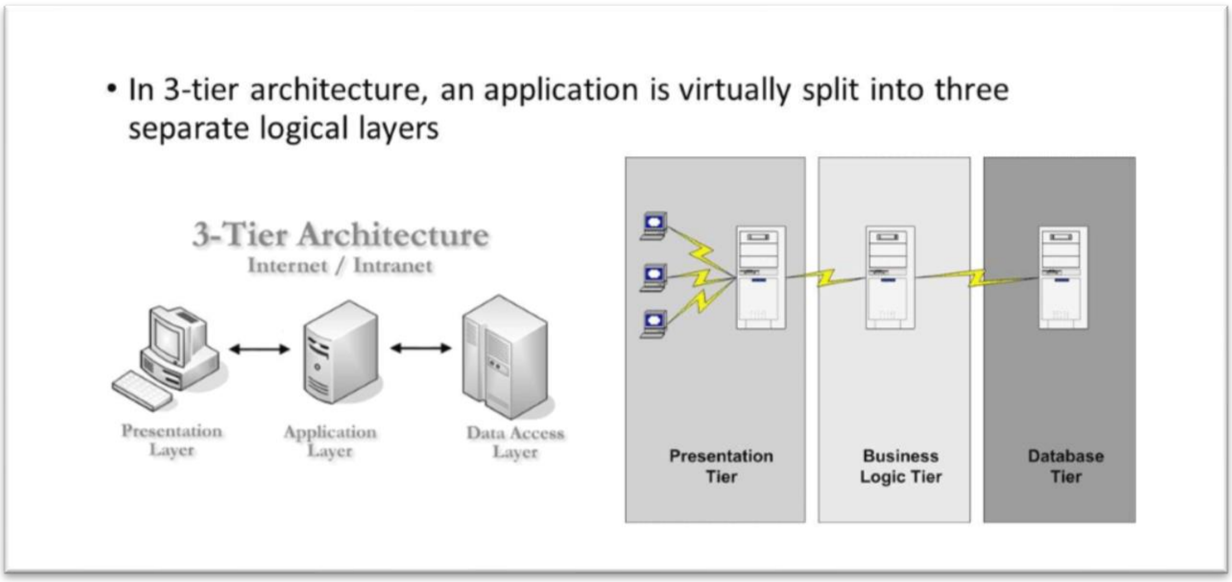


Figure 15 - Agile Feature Methodlogy

3.3. Overview of System

The systems' technical architecture is designed using a Three-Tier Client-Server Architecture. This is to ensure the logic of concerns are decoupled for minimal conflict between the three layers of software, known as the presentation layer, the application layer and the database layer. The presentation layer is managed by the client machine, the application layer is managed by the application server and the database layer is managed by the database server. The presentation layer can also be referenced as the front-view, the application layer as the middleware and the database layer as the backend. The previous mentioned terminologies can interchange. The methodology for building the application is the combination of FDD, Agile and Prototyping (as described previously) **(24).**



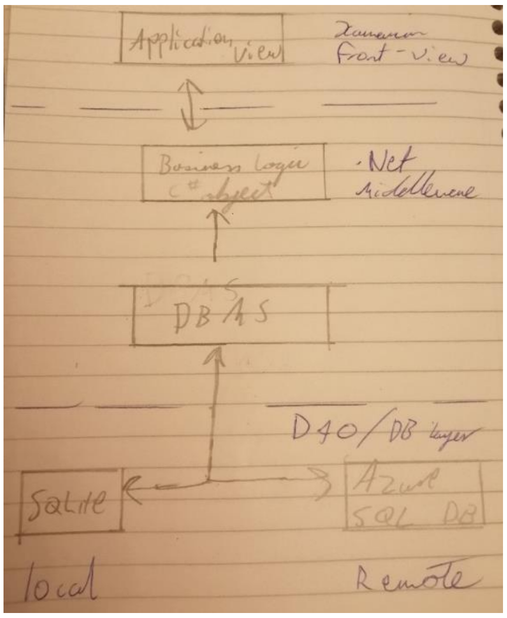


Figure 16 - Client Server Architecture

3.4. Front-End

The front-end aspect of the application is the presentation layer of the system architecture. It displays the information in which the application will look like to the user. This is critical to the system as the complexity of the application is geared towards the user experience, the user interface and design. The functionality of this layer must be polished to properly capture the input of the user in retrospect to their diet.

Low Fide Prototype

Paper prototypes were conjured as the presentation ideas were fleshed out. As part of the first iteration, these low-fide diagrams outlined the first look of the application.

A look at some of the drafts

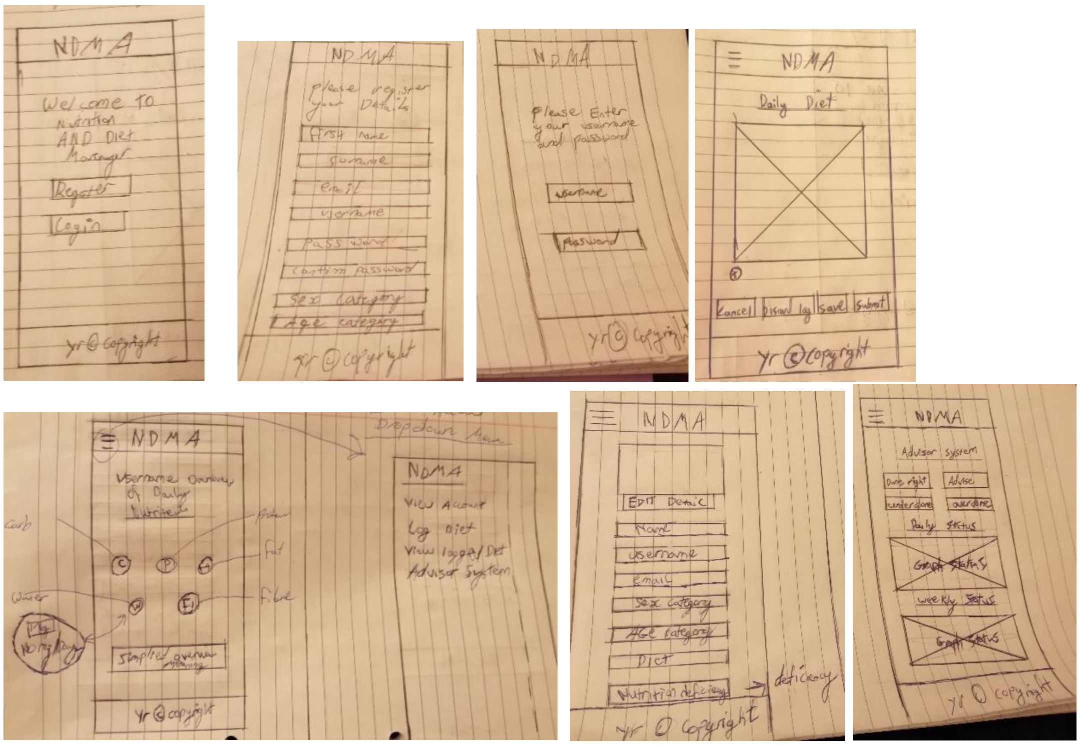


Figure 17 - Low Fide Prototype

Medium Fidelty Prototype

Afterwards the layout of midium fidelty version of the prototype were created, both to display the newer look of the prototype with adding visual aesthic features to the application and the associtaced behaviour of the application. The prototypes and the associcted stroyboards were created from an online software called FluidUI.

First Storyboard Iteration

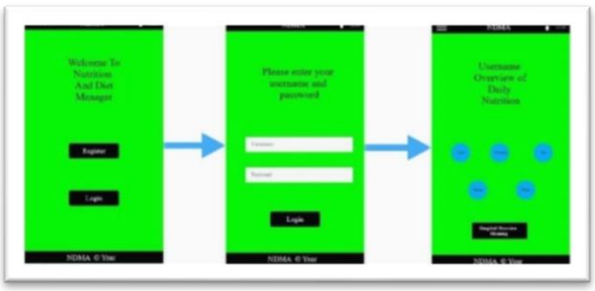
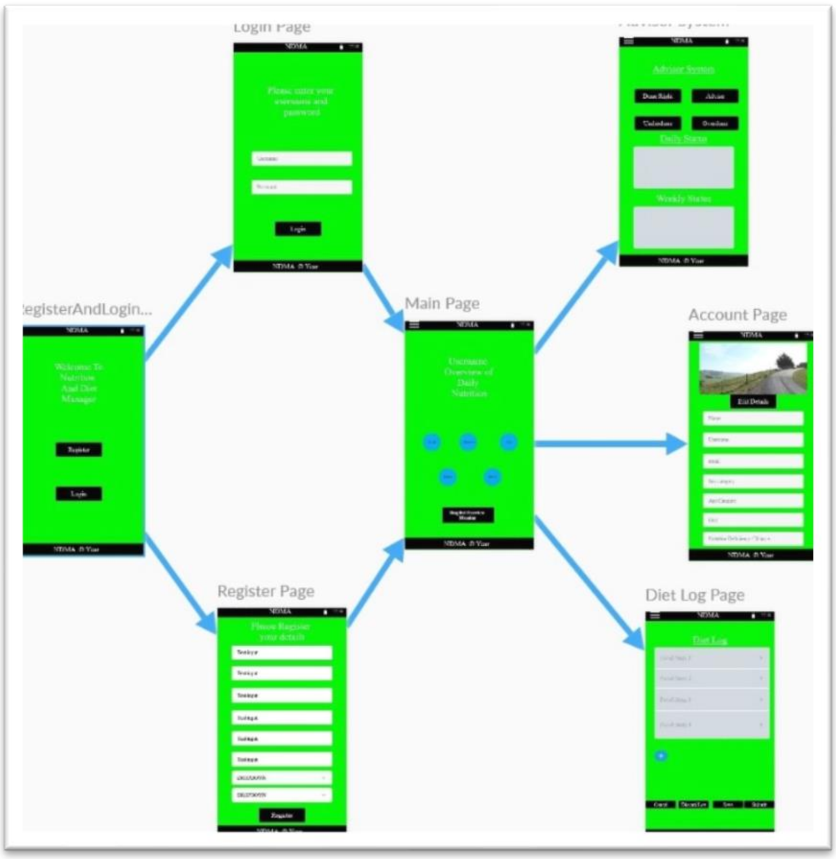
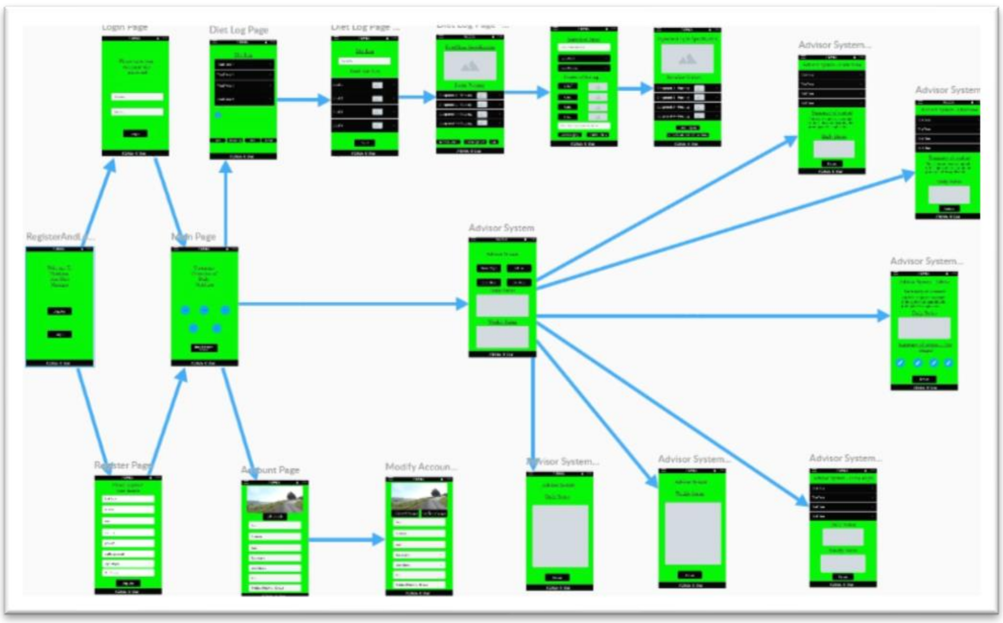

Welcome To Nutrition And Diet Manager
Register
Login
NDMA © Year
Please enter your username and password
Login
NDMA © Year
Username Overview of Daily Nutrition
NDMA © Year

Figure 18 - Storyboard iteration

Second Storyboard Iteration



Third Storyboard Iteration



Some of the Wireframe layout

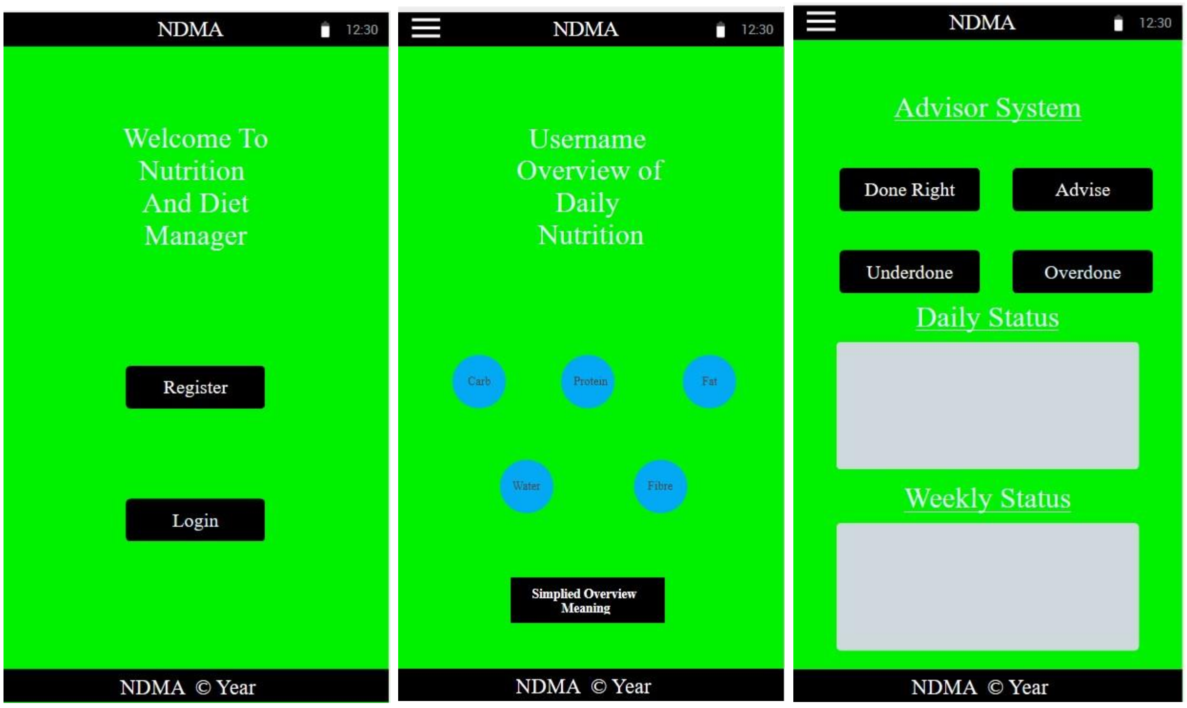


Figure 19 - Wireframe layout

Use Case Diagrams were created alongside the applications prototypes and are used to represent the system behaviour with the user interaction. They also demonstrate both compulsory functionality and extendable ones for each part the user interacts with. This was drafted in three stages.

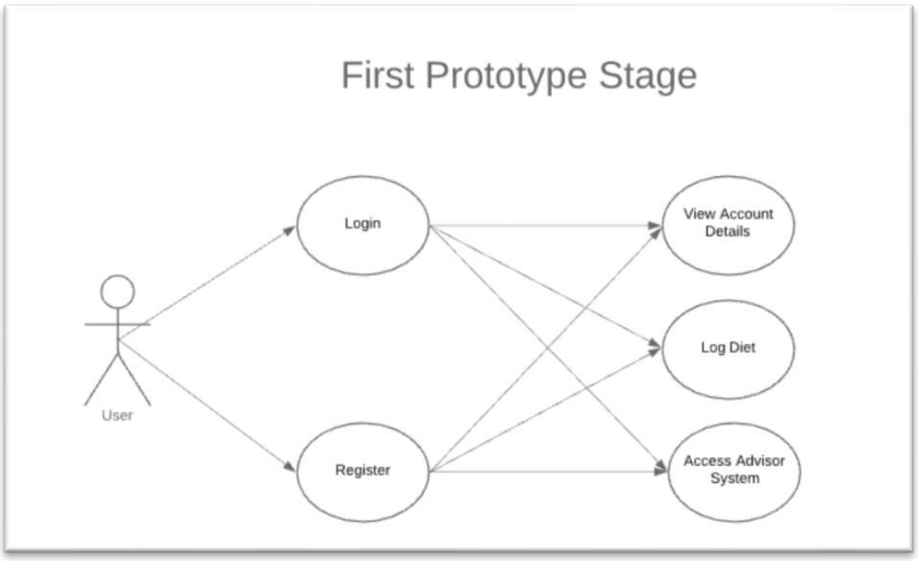
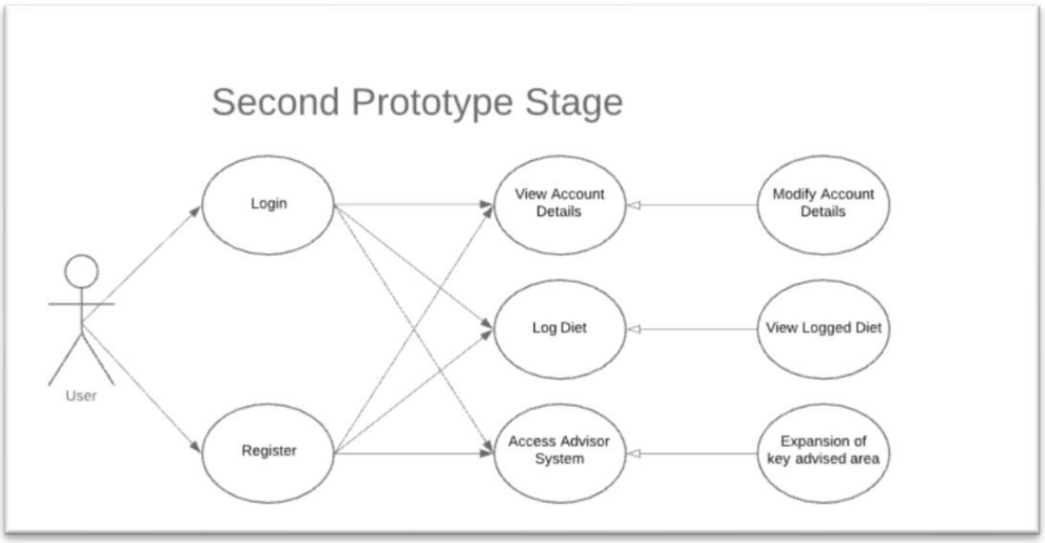
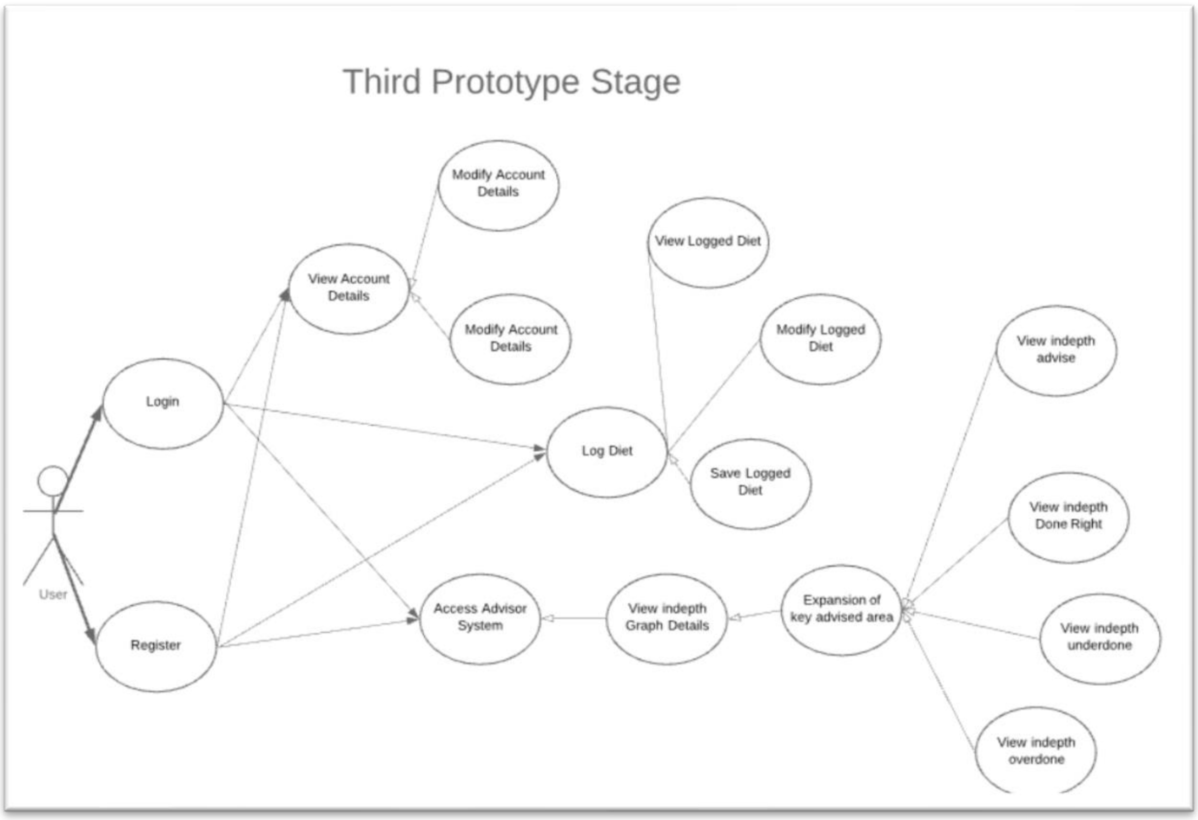


Figure 20 - Use Case Diagrams





3.5. Middle-Tier

The Middle-Tier aspect, also known as the middleware and the application layer, is the logic aspect of the behaviour of the application. It controls the functionality of the application and communicates with the front-end or the backend as necessary. Because the system is a mobile application, the middle-tier provides the directions to get from one screen to the next. It also retrieves the request information from the database to provide to the user for overview, such as their logged diet schedule, or the advice from the system.

In the case of the designated system, the .Net Framework will be the key for the overview of the system.

3.6. Back-End

The backend, also known as the Database layer, is the Data Access Route to all the storage within the system, such as the user credtials and their associated logged diets. This will be accessed from the Database managament system from the middleware section of the system. For local storage, SQLite will be used as the temperaroy storage system, in case the system cannot access the remote storage section. This would be handled by the Azure SQL database as the remote storage.

A few Entity Relational Diagrams have been complied below in different iterations to display the overview of the application backend behaviour.

First Iteration

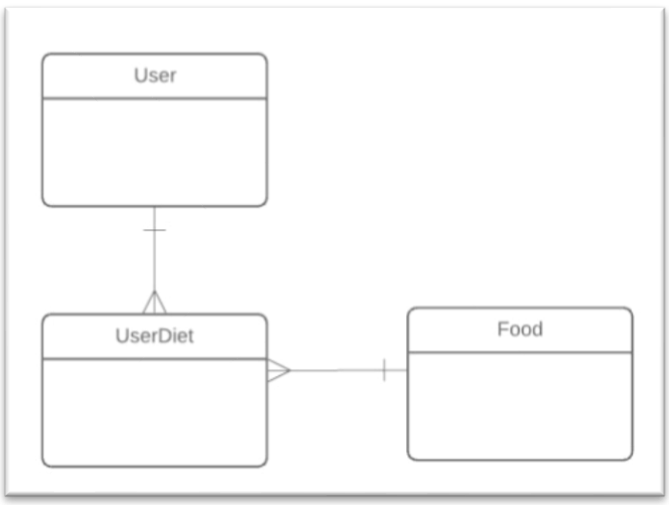
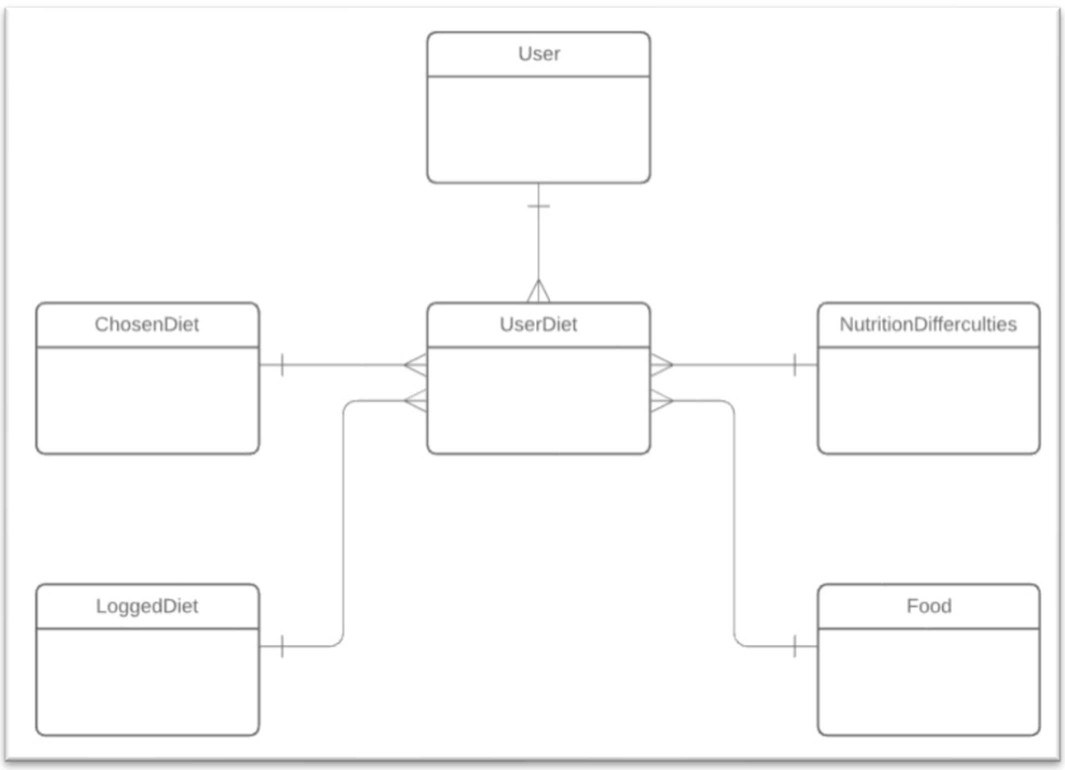
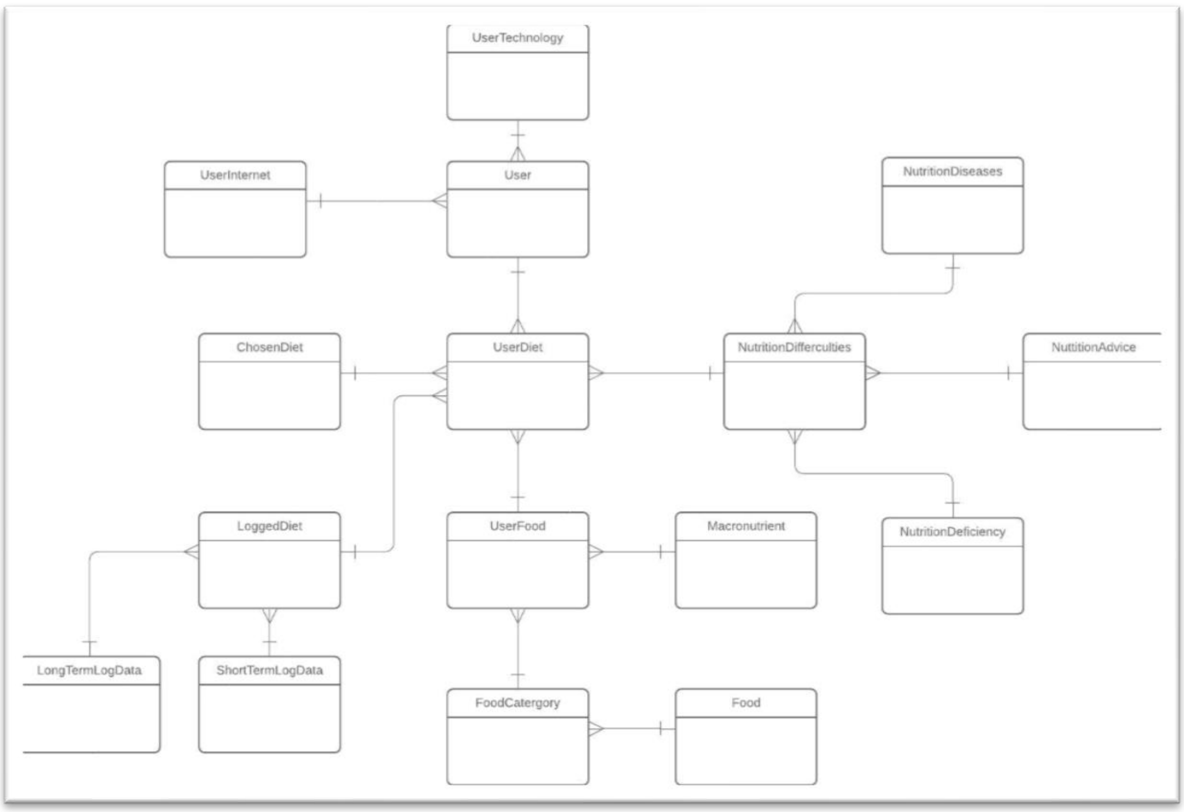


Figure 21 – Entity Relation Diagrams

Second Iteration



Third Iteration



3.7. Conclusions

The overall system was analysed in this chapter in high level. This involves from the system architecture to the full stack development, encompass of the front-end, middleware and the backend, of the system and finally, the chosen methodology for the system. A lower level analysis will be covered in depth in the next chapter which will cover many of the same themes covered in this chapter. They will also cover the problems and potential changes encountered in the development process too.

4. Prototype Development

4.1. Introduction

The description of how the system design choices were implemented as part of the prototyping process combined with some unexpected encounters that were met. The link to the project can be found at the following link: https://github.com/WilliamCareySemiColon/DietaryAssistanceSystem .

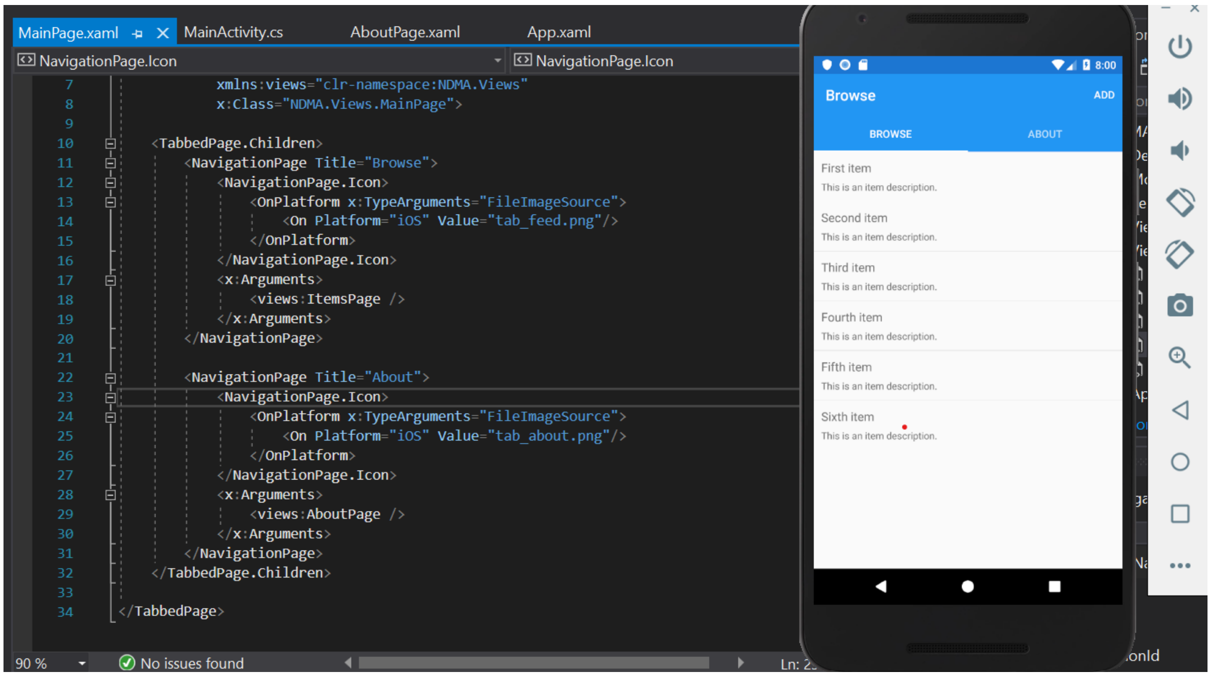
4.2. Prototype Development

There were numerous steps taken for the work on the prototype. The first server was to set up the version control so we can ensure we manage the project properly in iterations. GitHub was the choice to work with due to the familiarity of the technology, a reliable web hosting service for managing projects and combined well with its seamless integration into the visual studio IDE with the workload. After setting up the version control properly, the project could then be started on within the local version of the repository.

The next step was to connect everything together. From ensuring the visual studio knew there was a version control tracking the project to connecting to the Azure Cloud Services from within the environment. Finally, creating a way to store the data locally in case the device at the time does not have internet connection. For the prototype itself, a demonstration with the localhost database would be acceptable as a temporary solution.

4.3. Front-End

For the front-end, eXtensible Markup Language (XML) was used. XML is a universally language that can be used across different environments platforms to display information in the desired way. It is like HTML in this way and is recommended by the World Wide Web Consortium (W3C). The prototype front-view was still in progress at the time of write-up.



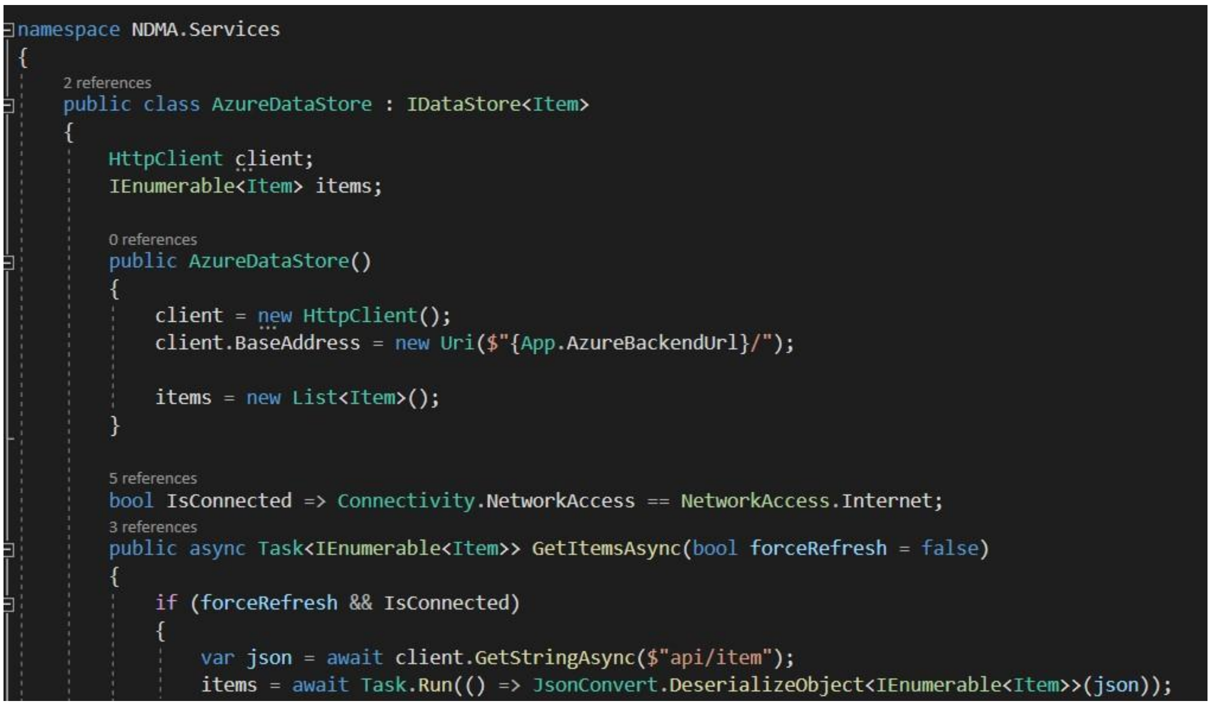
```
             xmlns:views="clr-namespace:NDMA.Views"
             x:Class="NDMA.Views.MainPage">

    <TabbedPage.Children>
        <NavigationPage Title="Browse">
            <NavigationPage.Icon>
                <OnPlatform x:TypeArguments="FileImageSource">
                    <On Platform="iOS" Value="tab_feed.png"/>
                </OnPlatform>
            </NavigationPage.Icon>
            <x:Arguments>
                <views:ItemsPage />
            </x:Arguments>
        </NavigationPage>

        <NavigationPage Title="About">
            <NavigationPage.Icon>
                <OnPlatform x:TypeArguments="FileImageSource">
                    <On Platform="iOS" Value="tab_about.png"/>
                </OnPlatform>
            </NavigationPage.Icon>
            <x:Arguments>
                <views:AboutPage />
            </x:Arguments>
        </NavigationPage>
    </TabbedPage.Children>

</TabbedPage>
```

First item
This is an item description.
Second item
This is an item description.
Third item
This is an item description.
Fourth item
This is an item description.
Fifth item
This is an item description.
Sixth item
This is an item description.

Figure 22 - Front-end Prototype Diagram

4.4. Middle-Tier

For the middleware, the Xamarin, .Net framework and C# language is the driver of the application. The Xamarin framework is the developer for mobile applications, which allows for the cross-platform development. The .Net framework separates the front-end from the backend, which is the primary reason of its existence. The C# programming language is the implementer of the frameworks. It provides the functionality to the front-end xml fields, implements the business logic and captures data provided to and from the backend. The main logic for the proof of concept is currently in progress.

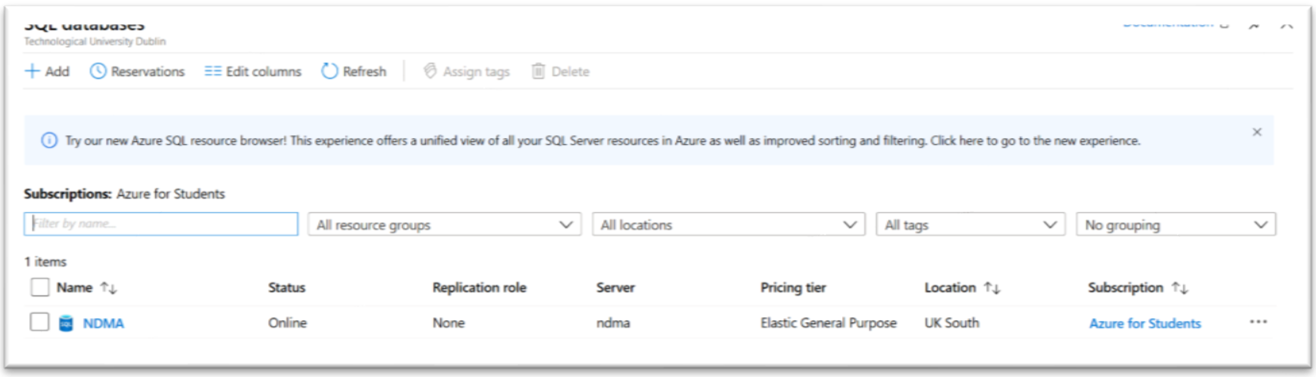


```
namespace NDMA.Services
{
    2 references
    public class AzureDataStore : IDataStore<Item>
    {
        HttpClient client;
        IEnumerable<Item> items;

        0 references
        public AzureDataStore()
        {
            client = new HttpClient();
            client.BaseAddress = new Uri($"{App.AzureBackendUrl}/");

            items = new List<Item>();
        }

        5 references
        bool IsConnected => Connectivity.NetworkAccess == NetworkAccess.Internet;
        3 references
        public async Task<IEnumerable<Item>> GetItemsAsync(bool forceRefresh = false)
        {
            if (forceRefresh && IsConnected)
            {
                var json = await client.GetStringAsync($"api/item");
                items = await Task.Run(() => JsonConvert.DeserializeObject<IEnumerable<Item>>(json));
```

Figure 23 - Middleware Prototype

4.5. Back-End

The database used for storage is both Azure Cloud SQL Database and SQLite database as the localhost. This is to provide both remote and local storage. Both provided issues as they had connectivity issues with the visual studio IDE, which took some time to figure this out.



SQL databases
Technological University Dublin

Try our new Azure SQL resource browser! This experience offers a unified view of all your SQL Server resources in Azure as well as improved sorting and filtering. Click here to go to the new experience.

Subscriptions: Azure for Students

1 items

Name	Status	Replication role	Server	Pricing tier	Location	Subscription
NDMA	Online	None	ndma	Elastic General Purpose	UK South	Azure for Students

Figure 24 - Backend Prototype

4.6. Conclusions

This chapter delved into the implementation of the prototypes. From the time of recording drafts to the connectivity issues and further plans to update the design of the prototype. The plan of action was also decided.

5. Testing and Evaluation

5.1. Introduction

This chapter will delve in the different steps and stages of what the plan is for the testing of the system combined with the evaluation mechanisms drafted up. This will incorporate tools from software as well as human interaction methods too.

5.2. Plan for Testing

NDMA will be tested across all the different layers of the application. The plan for the testing aspect, in a high-level overview, is to combine the usage of using manual / human methods and integrated automated software. The system in its entirety will use Acceptance testing in the full stack development to ensure each layer interact with one another correctly and the state of NDMA is acceptable according to minimum operations standards. Tools, such as Benchmarks or Performance analysis or profiling tools, which are freely provided by Google and other big tech companies, will also be used in conjunction with the debugging process of the application.

For the front-end of the application, I will develop an auto testing application. The technologies used in this development will be the frameworks of Selenium and NUnit. The Selenium framework allows the tests developed to be automate tasks combined with ensuring the testing system itself is decoupled. This will allow the potential of testing certain components or functionalities without affecting other aspects of the NDMA or the testing system. Reusing certain tests for similar functionality is another reason for using Selenium framework. Selenium is a working example of module testing.

NUnit is an open source Unit testing framework released under the MIT license for the .Net framework. Unit testing involves taking certain components of the application, test their performance against certain cases and assert them against expected outcome. This could be testing certain operations of the application and can use self-managed inputs as part of each case. NUnit, when combined with Selenium, can test each component of NDMA in automated session and assert their outcome.

Figure 25 - Selenium Logo

NUnit by Charlie Poole, 7.43M downloads v3.5.0
NUnit is a unit-testing framework for all .NET languages with a strong TDD focus.

NUnit.Runners by Charlie Poole, 1.21M downloads v3.5.0
Deprecated NUnit 3 console runner - use NUnit.Console or NUnit.ConsoleRunner.

NUnit.Console by Charlie Poole, 275K downloads v3.5.0
Console runner for the NUnit 3 unit-testing framework with selected extensions.

NUnit
Version(s) - 0

■	Project ^
☐	DemoProject
✓	DemoProjectTest

Figure 26 - NUnit Logo

For the middleware, NUnit testing will be integrated. Since this area is the key part of NDMA, Black box, White box and Grey box testing will be used here to ensure the performance and efficiency are optimised simultaneously with correct information. This is combined with ensuring the application can withstand faults, whether it is a technical issue or an injection attack from the user from the front-end of the application. Ensuring NDMA is de-coupled as necessary, so Subsystem testing will be incorporated. This is to ensure each application area only communicates in the way they are designed to do, combined with handling items they need to handle with only.

For database testing, DbFit will be used. This framework uses test-driven development as the methodology and encapsulates both unit testing with integration testing as part of its features. This will allow the test cases to capture specific information of the relational database. As a result, we can test the local storage and the remote storage using the same test scenarios with different cases. With the remote storage, Azure offers the services of regression testing with performance analytics, which will be integrated into the database testing. This will allow test cases to be designed for the availability and consistency of the remote application to be completed in ongoing iterations.

Manual testing will also be captured. A set of instructions will be provided to each person whom tests the application in order to provide as much accuracy as possible to the test cases. The instructions will be specific enough for the user to understand what to do combined with the vagueness of allowing the user free roaming of what to test. This will provide an more overview of the application behaviour combined with another opinion on the application design

5.3. Plan for Evaluation

The plan for the evaluation is in three different stages, which is to be completed altogether and separated for a diverse set of detail. The three stages are using the 10 Neison's Heurstics as an evaluation method, comparison with industry technology as another evaluation method and finally getting feedback from the users who have teste the application.

When getting the feedback from the users who have tested the system, two methods will be used. One is gathering the opinion of the users from their experience in terms of the clarity of the application, the design, the user interface, the user experience and the performance of the application. The other method would be questions drafted using the Neison's Heurstics as the guide to gather the other experience of the user. A comparison from other technologies that the user has used is an extendable option.

The evaluation of the system will be completed by Self too. This will be done using the same format as the methods used for the users’ evaluation described above. From comparison with industry technology to using the Neison's Heurstics (can be simultaneously completed), the similarities and contrasts with be documented and outlined in each iteration. Once the evaluation is completed, using both the users' report and the self-documentation, the system will be enhanced and modified to ensure it suits the needs of the users completely.

5.4. Conclusions

This chapter delved into the different test techniques and types planned for the system, as well as the evaluation mechanisms for the optimising of the application design. From the boxes testing (white, grey and black) to combining unit and module testing to combining different technologies to accomplish these goals when developing the system. The evaluation phase will be key to the development of the application through gathering the user assessment to the comparison to the industry technology. Using Neison's Heurstics is also integral to the evaluation phase too.

6. Issues and Future Work

6.1. Introduction

A discussion of areas where it could possibly go wrong with the application and other possible work which could be implemented outside the current scope of the application lifecycle.

6.2. Issues and Risks

Some of the risks associated with the program includes the following:

A security risk associated is the application, due to not being a security application, will not be as safe as other applications. If time allows it, an encryption library could be imported. The medical risk is someone might take the application as in production (ready-made). So, a disclaimer would have to be imputed to prevent this.

If a technical requirement is missing, an appropriate substitution will have to be found quickly while temporary technology would have to be used. The works on the application will use git server control to track its progress, so the data will be backed up as necessary.

The last risks are unforeseen events occur, such as sickness or family matters etc. Should any of the two happen, appropriate measures will follow suit, such as getting in contact with the authorities of the school of computing etc. A schedule was created to minimise the possibility of this hindering the project overall development.

6.3. Plans and Future Work

The plan for the project be found inside the GANTT chart below and the schedule image following it afterwards. The chart and schedule complement one another in detail for both textual purposes and visual purposes. Another version of these descriptors will be drafted up and used to see how the plans for the project have been altered over time.

In the incoming semester, the technical development of the project will be carried out for all the tasks within the scope. Areas outside the scope not previously mentioned are the development of a web application to capture more potential audience, display information in different ways etc, to work with the mobile application. Another area outside the scope is the use of predicative analysis to enhance the user experience and create more accurate analysis for the advisor system.

6.3.1. GANTT Chart

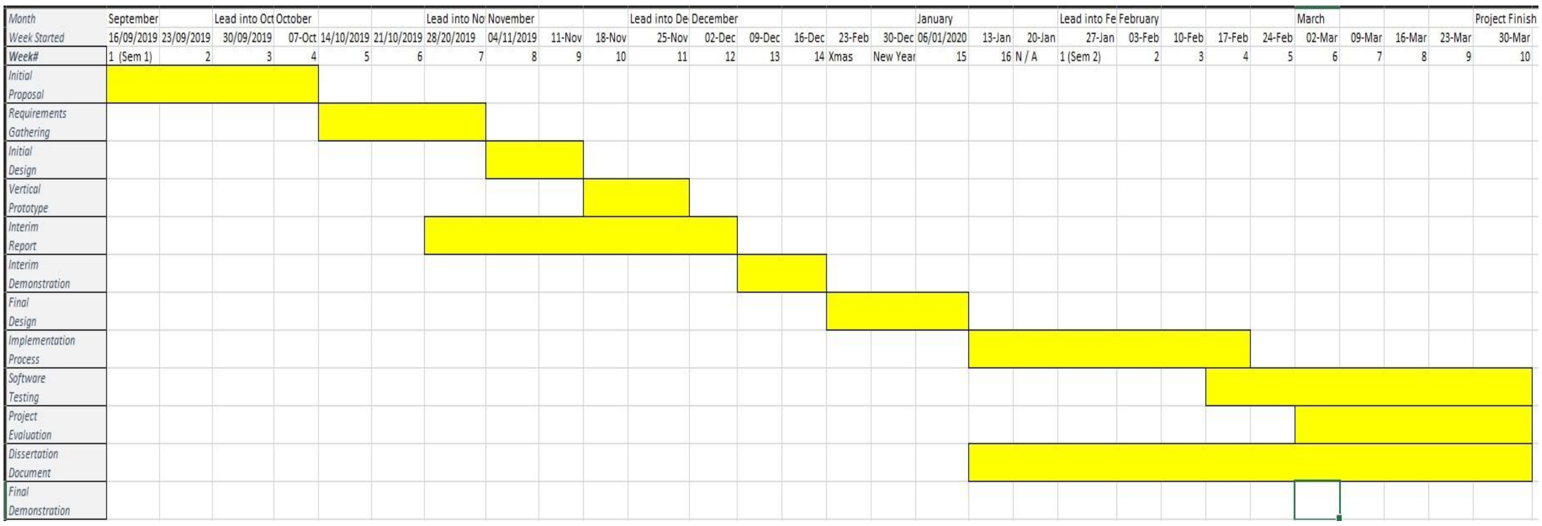
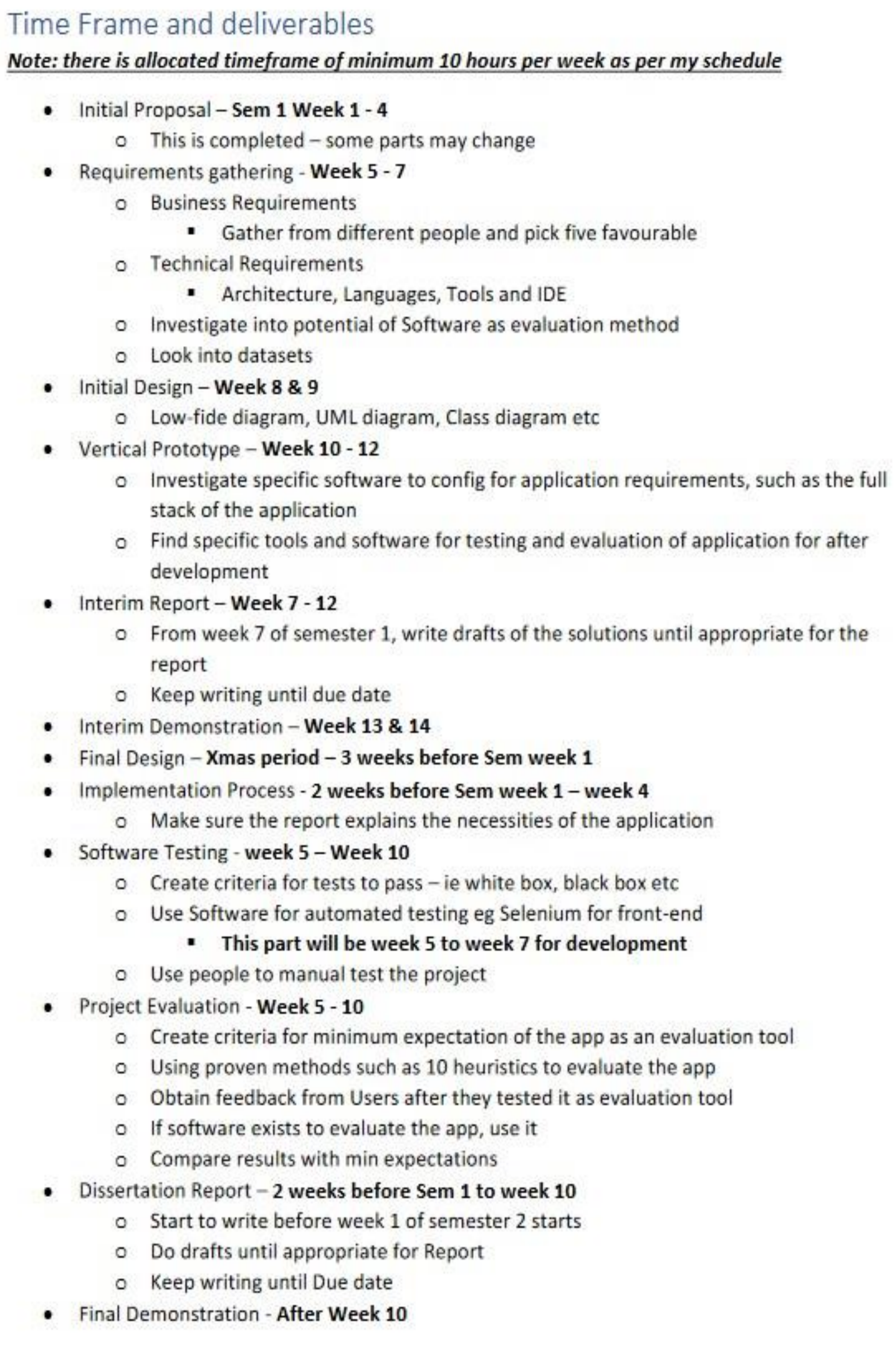


Figure 27 - GANTT Chart



Time Frame and deliverables

Note: there is allocated timeframe of minimum 10 hours per week as per my schedule

- Initial Proposal – **Sem 1 Week 1 - 4**
    - This is completed – some parts may change
- Requirements gathering - **Week 5 - 7**
    - Business Requirements
        - Gather from different people and pick five favourable
    - Technical Requirements
        - Architecture, Languages, Tools and IDE
    - Investigate into potential of Software as evaluation method
    - Look into datasets
- Initial Design – **Week 8 & 9**
    - Low-fide diagram, UML diagram, Class diagram etc
- Vertical Prototype – **Week 10 - 12**
    - Investigate specific software to config for application requirements, such as the full stack of the application
    - Find specific tools and software for testing and evaluation of application for after development
- Interim Report – **Week 7 - 12**
    - From week 7 of semester 1, write drafts of the solutions until appropriate for the report
    - Keep writing until due date
- Interim Demonstration – **Week 13 & 14**
- Final Design – **Xmas period – 3 weeks before Sem week 1**
- Implementation Process - **2 weeks before Sem week 1 – week 4**
    - Make sure the report explains the necessities of the application
- Software Testing - **week 5 – Week 10**
    - Create criteria for tests to pass – ie white box, black box etc
    - Use Software for automated testing eg Selenium for front-end
        - **This part will be week 5 to week 7 for development**
    - Use people to manual test the project
- Project Evaluation - **Week 5 - 10**
    - Create criteria for minimum expectation of the app as an evaluation tool
    - Using proven methods such as 10 heuristics to evaluate the app
    - Obtain feedback from Users after they tested it as evaluation tool
    - If software exists to evaluate the app, use it
    - Compare results with min expectations
- Dissertation Report – **2 weeks before Sem 1 to week 10**
    - Start to write before week 1 of semester 2 starts
    - Do drafts until appropriate for Report
    - Keep writing until Due date
- Final Demonstration - **After Week 10**

Figure 28 - Yearly Schedule

Bibliography

1. American Heart Association Recommendations for Physical Activity in Adults and Kids [Internet]. www.heart.org. 2019 [cited 10 November 2019]. Available from: https://www.heart.org/en/healthy-living/fitness/fitness-basics/aha-recs-for-physical-activity-in-adults
2. Davis N. Poor diet a factor in one-fifth of global deaths in 2017 – study [Internet]. the Guardian. 2019 [cited 10 November 2019]. Available from: https://www.theguardian.com/society/2018/nov/08/poor-diet-a-factor-in-one-fifth-of-global-deaths-in-2017-study
3. Oenema A, Brug J, Lechner L. Web-based tailored nutrition education: results of a randomized controlled trial. Health Education Research [Internet]. 2019 [cited 29 September 2019];16(6):647-660. Available from: https://academic.oup.com/her/article-lookup/doi/10.1093/her/16.6.647
4. de Castro J. The Time of Day of Food Intake Influences Overall Intake in Humans. The Journal of Nutrition [Internet]. 2004 [cited 29 September 2019];134(1):Pages 104–111. Available from: https://academic.oup.com/jn/article/134/1/104/4688191
5. [Internet]. Play.google.com. 2019 [cited 9 December 2019]. Available from: https://play.google.com/store/apps/details?id=com.fitbit.FitbitMobile&gl=IE
6. [Internet]. Play.google.com. 2019 [cited 9 December 2019]. Available from: https://play.google.com/store/apps/details?id=com.mysugr.android.companion&gl=IE
7. PhoneGap: What Is Good and What Is Bad - Orioninfosolutions Blog [Internet]. Orioninfosolutions Blog. 2019 [cited 1 November 2019]. Available from: https://www.orioninfosolutions.com/blog/phonegap-what-is-good-and-what-is-bad/
8. The Good and the Bad of Ionic Mobile Development [Internet]. AltexSoft. 2019 [cited 1 November 2019]. Available from: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-ionic-mobile-development/
9. The Good and the Bad of ReactJS and React Native [Internet]. AltexSoft. 2019 [cited 1 November 2019]. Available from: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/
10. The Good and The Bad of Xamarin Mobile Development [Internet]. AltexSoft. 2019 [cited 1 November 2019]. Available from: https://www.altexsoft.com/blog/mobile/pros-and-cons-of-xamarin-vs-native/
11. What is the .NET Framework (.NET)? - Definition from Techopedia [Internet]. Techopedia.com. 2019 [cited 25 November 2019]. Available from: https://www.techopedia.com/definition/3734/net-framework-net
12. Everything You Need to Know About the Android OS [Internet]. Lifewire. 2019 [cited 9 December 2019]. Available from: https://www.lifewire.com/what-is-google-android-1616887
13. Barraclough C. What Is iOS and What Does iOS Stand For? [Internet]. Recombu. 2019 [cited 9 December 2019]. Available from: https://recombu.com/mobile/article/what-is-ios-and-what-does-ios-stand-for
14. C# Programming Language - GeeksforGeeks [Internet]. GeeksforGeeks. 2019 [cited 9 December 2019]. Available from: https://www.geeksforgeeks.org/csharp-programming-language/
15. Python Language Introduction - GeeksforGeeks [Internet]. GeeksforGeeks. 2019 [cited 9 December 2019]. Available from: https://www.geeksforgeeks.org/python-language-introduction/
16. 4. Usability Theory - Technical Communication Body of Knowledge (TCBOK) [Internet]. Tcbok.org. 2019 [cited 9 December 2019]. Available from: https://www.tcbok.org/wiki/research/bibliography/bibliography-usability/usability-theory/
17. Ahead - Assistive Technology [Internet]. Ahead.ie. 2019 [cited 9 December 2019]. Available from: https://www.ahead.ie/assistivetech-students
18. 10 Heuristics for User Interface Design: Article by Jakob Nielsen [Internet]. Nielsen Norman Group. 2019 [cited 9 December 2019]. Available from: https://www.nngroup.com/articles/ten-usability-heuristics/
19. 7 Best Practices to Overcome Mobile App Usability Issues [Internet]. Clearbridge Mobile. 2019 [cited 9 December 2019]. Available from: https://clearbridgemobile.com/7-best-practices-to-overcome-mobile-app-usability-issues/
20. Feature Driven Development Methodology [Internet]. Newline.tech. 2019 [cited 9 December 2019]. Available from: https://newline.tech/blog/feature-driven-development-methodology/
21. 10 Key Principles of Agile Software Development [Internet]. Project-Management.com. 2019 [cited 9 December 2019]. Available from: https://project-management.com/10-key-principles-of-agile-software-development/
22. Volchko J. Prototyping Methodology: Steps on How to Use It Correctly [Internet]. Lumitex.com. 2019 [cited 9 December 2019]. Available from: https://www.lumitex.com/blog/prototyping-methodology
23. The 80/20 Rule, What Is It and How To Apply It? [Internet]. 2 Meal Day. 2019 [cited 9 December 2019]. Available from: https://2mealday.com/article/the-80-20-rule-what-is-it-and-how-to-apply-it/
24. What is client-server architecture and what are its types? [Internet]. Apachebooster Blog: Showcasing the tech blogs written by our writers. 2019 [cited 9 December 2019]. Available from: https://apachebooster.com/blog/what-is-client-server-architecture-and-what-are-its-types/