|  |
| --- |
| COM667 Computing Systems Project |
| Final Report |
| Food and Fitness Tracker - FFT |

|  |
| --- |
| Christopher Jordan  [Date] |

Contents

[Draft Title 2](#_Toc531715784)

[Project Background 2](#_Toc531715785)

[Project Aim 2](#_Toc531715786)

[Copyright, Intellectual Property Rights or Commercial Sensitivity 2](#_Toc531715787)

[Hardware and Software 2](#_Toc531715788)

[Chapter 2: Planning 3](#_Toc531715789)

[Software Development Life Cycles 3](#_Toc531715790)

[What is a SDLC? 3](#_Toc531715791)

[Stages of a SDLC 4](#_Toc531715792)

[Software Development Models 5](#_Toc531715793)

[Chapter 3: Analysis 9](#_Toc531715794)

[Requirement Gathering 9](#_Toc531715795)

[Focus Group 10](#_Toc531715796)

[Observations 10](#_Toc531715797)

[Requirements 11](#_Toc531715798)

[Functional Requirements 11](#_Toc531715799)

[Non-Functional Requirements 12](#_Toc531715800)

[Requirement Prioritisation Strategy 12](#_Toc531715801)

[References 15](#_Toc531715802)

## Draft Title

FFT- Food & Fitness Tracker

# Project Background

Over the last 6 months I have gained an increased interest in my personal health and wellbeing. To help me keep a record of what I am eating and what exercises I am doing on daily basis, I employ the use of several mobile applications, as I have not been able to find an app which will do the same job as all three apps. These applications are “Apple Health”, “MyFitnessPal” and “Nike Run Club”. The main issue that I have with using these three applications is that whilst each app does record different information, and each have their own advantages such exercise route tracking, calories eaten in day and projected weight loss. They are also inefficient due the amount of crossover information that is stored on each of the application such as the total number of steps taken in a day and calories burned. The second issue I have with these apps is the number of adverts that appear (i.e. MyFitnessPal and Nike Run Club) when they are used. The third and final problem with them is that the information breakdowns tend to be quite complex and therefore not very helpful to a normal user. This app would initially be used by myself and a few of my friends who have encountered similar issues.

Due to the above reasons, I desire to create an application that will keep track of a user’s daily eating and fitness habits. The initial features for the proposed solution will allow the user to enter their daily diet and any exercises they have completed. As well as being able to track the current route a user is taking for an exercise that they are currently completing. A breakdown of the users’ daily diet and fitness habits will also be provided to the user.

# Project Aim

My project aim is to create a cross-platform application that will allow the user to track their daily diet and exercise activities and provide a breakdown of the information in an easily interpreted graphical representation.

# Copyright, Intellectual Property Rights or Commercial Sensitivity

The only area of concern that I will need to ensure that I comply with, when creating the application would be data protection as set by the General Data Protection Regulation (GDPR) by the EU. This law requires me to ensure that all personal data and the privacy of EU citizens is protected.

# Hardware and Software

The only hardware necessary for me to create the app would be my own personal pc. As well as this at this stage of my initial project planning I am intending to create the application using C# programming language and the Xamarin software development tool. Also required will be the Xamarin.Essentials APIs which are download via NuGet. Xamarin.Essentials APIs is what will allow me to access native features (i.e. accelerometer, geocoding and geolocation) on both IOS and Android devices without having to write my own abstractions or finding an open source plugin that could be used. Therefore, a NuGet account will also be required to retrieve the appropriate API libraries. (Montemagno, 2018)

# Chapter 2: Planning

## Software Development Life Cycles

A Software Development Life Cycle (SDLC) also known as a Software Development Process is a process used by people who work in the software industry to help them design, develop and test software to ensure that the finished software is of a high quality. A SDLC aims to create high-quality software that either meets or exceeds the customers’ expectations whilst ensuring that the end software is completed on-time and within cost estimates.

In an attempt to ensure that software is completed to the same standard when utilising a SDLC the international standard ‘ISO/IEC 12207’ was developed. This standard for software development processes aims to be the standard that defines all of the tasks that are required for developing and maintaining software. (Tutorialspoint, 2018)

### What is a SDLC?

A SDLC is a framework which defines the tasks to be performed at each stage of the software development process. It consists of a detail plan describing how to develop, maintain, replace and alter or enhance specific software. The SDLC defines a methodology that can be used to improve the quality of the software and the overall development process. (Tutorialspoint, 2018)

A close up of text on a white background

Description automatically generatedThe following figure is graphical representation of several of the stages in a typical SDLC:

*Figure 1:Stages of a Software Development Life Cycyle* (Wisdomjobs.com, 2018)

### Stages of a SDLC

The typical Software Development Life Cycle will be comprised of the following stages:

#### Planning and Requirement Analysis

Requirement Analysis is the most important stage of the SDLC, as it is this stage that determines if a project should proceed or if it should be cancelled. The Requirement Analysis stage should be carried by senior members of the development team and any key stakeholders such as the customer, sales team and any experts on the project area. The information gathered is the used to help plan how the project will be approached as well as to conduct a feasibility study into the operational, economical and technical areas of the project and if it should go ahead. (Wisdomjobs.com, 2018)

Planning for the quality assurance will also usually be carried out at this stage. This includes quality assurance requirements and the identification risks. This allows for a software project to be implemented with minimum risks. (Wisdomjobs.com, 2018)

#### Defining Requirements

Upon the Requirement Analysis has been completed the next stage in a life cycle is to define and document the product requirements and for them to be approved by either the customer or group who requested the project to be developed. Requirements a documented in a Software Requirement Specification (SRS) which will contain all of the project requirements that are to designed and development as part of the project life cycle.

#### Design

A Document Design Specification (DDS) will then usually be produced, which uses the SRS created earlier as a reference. A DDS can be either a very high-level document or a very low level document. A high-level DDS may only contain screenshots of what the user interface should look and where requirements are implemented. Whereas a low-level DDS will also describe of the software architecture that should be implement such as any databases, APIs and the classes and modules the code will include. Typically, more than one design will be produced for this document. (Tutorialspoint, 2018)

The DDS will then be reviewed by the key stakeholders involved in the project and depending on various the project parameters such as risk, cost and development time the most appropriate design will be chosen for implementation.

#### Building the Project

It is at this point in the SDLC that the actual development of the projects programming code is started. The code that is developed will follow the chosen design laid out in the DDS that was approved for implementation. It is important during the development process that the code written follows the organisations coding guidelines such as their choice of programming language and development tools.

#### Testing

Whilst testing should be carried out during all stages of the SDLC, this stage refers to the testing of the proposed completed project. Due to this stage of testing all bugs/issues should be reported, tracked and retested until the product is deemed to be stable and of a high enough standard as defined in the SRS that it can be released.

#### Deployment to the Market and Maintenance

Upon the completion of the Testing stage the product is then released formally into its intended market. Occasionally, the product will be released in a limited capacity to a User Acceptance Testing (UAT) environment. So that the intended end-user/customer can test the product and decide if they are satisfied with the product or if further development is required before formal release.

### Software Development Models

There are numerous approaches that can be followed for a software development lifecycle, with each methodology having their own advantages and disadvantages. The following software methodologies where reviewed and evaluated as part of my project, so that I could determine the most suitable methodology to follow for my project development life cycle.

* Waterfall Model
* Modified Waterfall Model
* Scrum Model
* Spiral Model

#### Waterfall Model

The Waterfall Model was developed in 1970 by Dr Winston W. Royce, it was the first process model that was introduced. The Waterfall model is very simple to understand and use with it emphasising a logical progression through the various stages of a SDLC. As such it requires that work is completed in a sequential manner which means a new piece of work cannot be started until the previous task has been completed. (TryQA.com, 2013)

Recently the popularity of the Waterfall model has fallen, due to the rise of the agile methodologies. This however does not mean that it is obsolete, as the logical nature of the waterfall model still retains its desirability for many software developers.



Figure 2: Waterfall Model(Citta, 2017)

##### Advantages

* Adapts to shifting teams
* Forces a structured organisation
* Allows for early design changes
* Suited for milestone development

##### Disadvantages

* Nonadaptive design constraints
* Lack of user/client feedback mid-process
* Delayed Testing Period

#### Modified Waterfall.

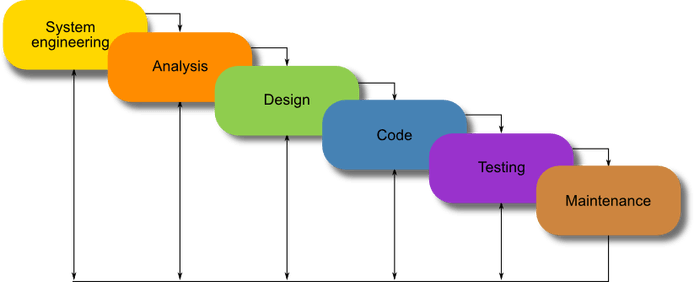
The modified waterfall model is almost identical to the waterfall model which it is based off. The modified waterfall model allows a developer to go back to an earlier task and update it if they release that something needs to be changed instead of going back starting all over again from the start.

Figure 2: Modified Waterfall Model (Powell-Morse, 2016)

##### Advantages

* Adapts to shifting teams
* Forces a structured organisation
* Allows for early design changes
* Suited for milestone development
* Adaptive to Design Constraints

##### Disadvantages

* Lack of user/client feedback mid-process
* Delayed Testing Period

#### Scrum Model

The Scrum model is an agile methodology that suggests that work should be carried out in series of sprints with each sprint lasting for an agreed upon period of time. In the Scrum model at the start of each sprint the team members will determine the number of items they can commit to. The team members will then create a sprint backlog which is a list of all of the tasks that are to be performed during the sprint.

As a part of the Scrum model that are two key roles which support the scrum teams, these the ScrumMaster, who could be thought of as the coach or captain of the team. It is the ScrumMasters’ role to help the team members involved in a sprint to use, the Scrum process, to allow them to perform at the highest level. (Mountain Goat Software, 2018)

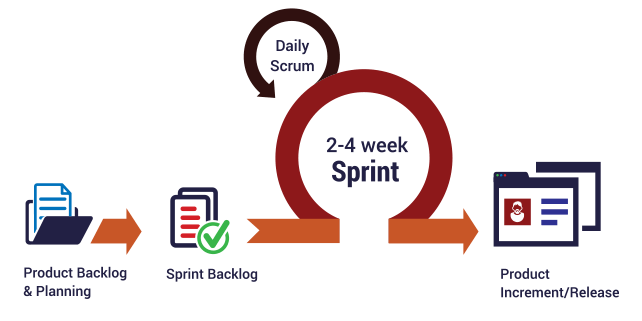


Figure 4: Scrum Model (Intelegain Technologies, 2017)

##### Advantages

* Ensures effective use of time and money
* Projects are divided into easily managed sprints
* Development work is tested in-sprint
* Feedback from customers and stakeholders
* Adaptive to Design Constraints

##### Disadvantages

* Scope creep, due to lack of a definite end-date
* Requires experienced team members
* Team member can become frustrated due to daily team meetings
* Difficult to adopt in large teams

(Chandana, 2018)

#### Spiral Model

The Spiral model is an incremental Software Development Life Cycle which focuses on risk analysis. The Spiral model was first described by Barry Boehm in his 1986 paper "A Spiral Model of Software Development and Enhancement". The spiral model combines combination of iterative development process model and sequential linear development model i.e. the waterfall model with a very high emphasis on risk analysis. This allows for the incremental release of the product or incremental refinement through each iteration around the spiral. In the spiral model the risk determines the amount effort required for each task as well as the amount of details required for a task to completed. (PAL, 2018)

##### Advantages

* Risk handling
* Good for large projects
* Flexibility in requirements
* Client/user feedback
* Adaptive to Design Constraints

##### Disadvantages

* Difficult to implement
* Expensive
* Highly dependent on Risk Analysis
* Difficult to manage time

#### Conclusion

Having reviewed the advantages and disadvantages of each of the four software development life cycles that are detailed in this report, I decided that the best approach for my project would be to follow the modified waterfall model.

The reason that I chose the modified waterfall model is that like I like the logical process that it follows when completing tasks. The reason I did not chose either the Scrum or Spiral models is due to my belief that they are better suited to projects that are being developed by a team and not as a solo project.

During the project it is important that I fellow the project plan that will be created and watch out for slipping into the habit continuously going back and forth between the various stages in the plan to make small incremental changes which could result in a delay in the project’s planned completion date.

## Technical Background

### Chosen Platform

The solution developed for this project should work on both iOS and Andriod. The reason for this is that these two platforms have the largest market share with them having a total market share of over 95%, from November 2017 to November 2018.

A screenshot of a cell phone

Description automatically generated

Source: (http://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-201711-201811-bar)

### Resources Required

* Ionic account
* SQLITE – database (used as the database only needs to be accessed by the user)
* Adobe Experience Design (XD) – UI design
* RapidAPi – API Hub (intending to use the Nutrionix api)
* Katalon Studio – Testing (automated API, mobile testing)
* Git/Github Desktop
* Node.js

### Development Choice

#### Xamarin

* C#
* Visual Studio plug-in
* Difficult for complex applications

#### Ionic

Ionic 3 (latest version) is written in HTML5, Javascript, and CSS. For this project this an advantage due to the fact that I have prior experience in them. Also included in Ionic is Angular.js which whilst I lack experience in at them moment, this is something that is likely to change due to the fact that my employer has asked me to help out on an Ionic project that is due to start in January.

Ionic offers the advantage of being able develop native looking applications for multiple platforms simultaneously.

Produces Native looking apps

* Also uses Angular.js
* Is an HTML5framework which works with the help of Cordova or PhoneGap.

##### Installation Requirements

Node.js

Git/Github Desktop

Cordova

# Chapter 3: Analysis

## Requirement Gathering

As part of any project or project management it is essential that requirements gathering is carried out as part of it. This allows for stakeholders to develop a full understanding of what it will deliver that is critical to its success. It is important that this stage of a project is not overlooked, so that further down the line it is not realised that the customers’ needs have not been adequately met due to the customers requirements not being fully understood.

To give this project a greater chance at success the following rules will be applied to help ensure successful requirement gathering is carried out:

1. Don't assume you know what the customer wants - always ask.
2. Involve the users from the start.
3. Define and agree on the scope of the project.
4. Make sure requirements are SMART - specific, measurable, agreed upon, realistic and time-based.
5. Gain clarity if there is any doubt.
6. Create a clear, concise and thorough requirements document and share it with the customer.
7. Confirm your understanding of the requirements alongside the customer (play them back).
8. Avoid talking technology or solutions until the requirements are fully understood.
9. Get the requirements agreed with the stakeholders before the project starts.
10. Create a prototype, if necessary, to confirm or refine the customer's requirements.

(Haughey, 2014)

During the requirements gathering process, there some commonly found mistakes that should be avoided. These mistakes are:

* Basing a solution on complex or cutting-edge technology and then discovering that it cannot easily be rolled out in the 'real world'.
* Not prioritising the requirements, for example, 'must have', 'should have', 'could have' and 'would have' - known as the MoSCoW principle.
* Insufficient consultation with real users and practitioners.
* Solving the 'problem' before you know what the problem is.
* Lacking a clear understanding and making assumptions rather than asking.

(Haughey, 2014)

Two different methods where utilised when it came to requirements gathering for this project to help ensure that no requirements were missed. The two methods employed for this project was a Focus Group and Observation. Both techniques were carried out over a period of time.

The collated requirements where then separated out into Functional and Non-Functional Requirements.

### Focus Group

Focus Groups are used to help provide a large variety of possible requirements, by getting a group of people together to discuss the problem and think of possible solutions and desirable functionality. This technique is useful in that it allows people to bounce ideas of each other and therefore creating broader range of ideas.

In this case I gathered four people together who had an interest in keeping fit and tracking their diet and briefed them on the project background and my aims for the completed solution. Upon being briefed I then asked them about how they currently keep track of this information and what they would like to see in the proposed solution.

### Observations

Observations is another useful technique for requirement gathering. Observations work by a person observing the environment that the solution will used in or by observing solutions that are already on the market.

Due to the fact my proposed solution is something that I personally would use and have prior experience when comes to solutions already on the market, meant that I had a unique perspective when it came to gathering Requirements. As part of this requirement gather technique I looked at apps that where already on the market and I have used in the past. These apps are:

* Apple Health
* Nike Run Club
* MyFitnessPal
* MapMyRun

Upon review I found Apple health to be the basic in terms of overall design with the app simply keeping track of the user sleep patter, total, steps, total distance and weight. This data was then viewable in a graph that allowed the user to keep track of the daily life.

Nike Run Club had what I feel to be the best user interface out of all the existing solutions that where reviewed, with the app being very easy to use and understand. The app offers guided runs and exercise by professional athletes that was a nice feature as well as an interactive map which your route was tracked on. Also included was the ability to set yourself challenges such as run to a certain distance or reach a certain time. However despite these positive feature there was one big drawback in that it was not able to keep track of your daily diet.

MyFitnessPal and MapMyRun are both made by the same company, in this case ‘Under Armour’. Due to this allowed you to like your accounts on both the apps together, however I felt that this was a complicated process and the apps contain almost too much information, which them quite complex to use and understand. Another downfall for these apps was the number adverts and restricted features which aren’t included unless they are bought by the user.

## Requirements

### Functional Requirements

A functional requirement is basically a requirement that specifies what a system ‘should’ do. A functional requirement will usually identify the behaviour or function of a feature which the end user will require the system to do. Table 1 lists all of the functional requirements defined for this project:

|  |  |
| --- | --- |
| Number | Functional Requirement |
| FR-01 | The application shall work on both iOS and Android platforms |
| FR-02 | A new user shall be asked to set up an account |
| FR-03 | The app shall remember a user after first-time set up |
| FR-04 | The user shall be able to navigate the app via a Navigation menu pane |
| FR-05 | The app shall allow a user to be able to search for meals. |
| FR-06 | A user shall be able to log a meal for breakfast |
| FR-07 | A user shall be able to log a meal for lunch |
| FR-08 | A user shall be able to log a meal for dinner |
| FR-09 | A user shall be able to log a snack |
| FR-10 | A user shall be able to view all the food they have logged in the last 7 days |
| FR-11 | A user shall be able to view a calorie break down for all the food they have eaten for the current day. |
| FR-12 | The app shall allow the user to start a run |
| FR-13 | The app shall allow the user to pause a run |
| FR-14 | The app shall allow the user to continue a run |
| FR-15 | The app shall allow the user to end a run |
| FR-16 | The app shall record the time ran by a user |
| FR-17 | The app shall record the distance covered during a run |
| FR-18 | The app shall record the calories burned during a run |
| FR-19 | Upon a run ending the app shall show the route a user ran on a map |
| FR-20 | Upon a run ending the app shall show the distance a user ran |
| FR-21 | Upon a run ending the app shall show the calories burned by a user |
| FR-22 | Upon a run ending the app shall show the average pace of the user |
| FR-23 | The app shall show statistics on all the runs a user has completed |
| FR-24 | The app shall display the total distance walked/ran in a day |
| FR-25 | The app shall show the average distance walked/ran since the user started using the app |
| FR-26 | A user will be required to enter their date of birth upon initial set up |
| FR-27 | The app shall ask the user to select a gender upon initial set up |
| FR-28 | The app shall ask the user to enter their weight upon initial set up |
| FR-29 | The app shall be able to randomly suggest a meal for breakfast |
| FR-30 | The app shall be able to randomly suggest a meal for lunch |
| FR-31 | The app shall be able to randomly suggest a meal for dinner |
| FR-32 | The app shall display the recipe upon selecting a meal |
| FR-33 | The app shall be provide a nutritional breakdown of any meals/snacks it suggests |

Table 1: Functional Requirements

### Non-Functional Requirements

A non-functional requirement simply describes ‘how’ are system should work upon completion. They help to determine the quality of the software and how it should be behave once developed. Table 2 list all of the non-functional requirements that my completed projects shall meet.

|  |  |
| --- | --- |
| Number | Non-Functional Requirement |
| NFR-01 | The app shall have a consistent theme throughout the application |
| NFR-02 | The app shall be simple to understand and use |
| NFR-03 | The app shall work on devices of different screen sizes |
| NFR-04 | The app shall not crash |
| NFR-05 | The app shall be capable of running in the background |
| NFR-06 | The app shall react appropriately to changes orientation |
| NFR-07 | The app shall be easy to navigate |
| NFR-08 | The app shall conform to best practice and design principles for the chosen platforms |

Table 2: Non-Functional Requirements

### Requirement Prioritisation Strategy

To ensure that this project is successful a requirement prioritisation strategy will be used. A requirement prioritisation strategy is used to determine which requirements should be implemented first and which ones can be delayed until later in the projects development lifecycle. For this project the MOSCOW technique was employed.

The MOSCOW technique categorises, requirements into the following 4 categories:

* Must Do
* Should Do
* Could Do
* Won’t Do

The following tables show the MOSCOW rating for each of the non-functional and functional requirements.

|  |  |  |
| --- | --- | --- |
| Number | Functional Requirement | MosCOW |
| FR-01 | The application shall work on both iOS and Android platforms | Must |
| FR-02 | A new user shall be asked to set up an account | Must |
| FR-03 | The app shall remember a user after first-time set up | Must |
| FR-04 | The user shall be able to navigate the app via a Navigation menu pane | Must |
| FR-05 | The app shall allow a user to be able to search for meals. | Should |
| FR-06 | A user shall be able to log a meal for breakfast | Must |
| FR-07 | A user shall be able to log a meal for lunch | Must |
| FR-08 | A user shall be able to log a meal for dinner | Must |
| FR-09 | A user shall be able to log a snack | Should |
| FR-10 | A user shall be able to view all the food they have logged in the last 7 days | Should |
| FR-11 | A user shall be able to view a calorie break down for all the food they have eaten for the current day. | Must |
| FR-12 | The app shall allow the user to start a run | Must |
| FR-13 | The app shall allow the user to pause a run | Should |
| FR-14 | The app shall allow the user to continue a run | Should |
| FR-15 | The app shall allow the user to end a run | Must |
| FR-16 | The app shall record the time ran by a user | Must |
| FR-17 | The app shall record the distance covered during a run | Must |
| FR-18 | The app shall record the calories burned during a run | Should |
| FR-19 | Upon a run ending the app shall show the route a user ran on a map | Should |
| FR-20 | Upon a run ending the app shall show the distance a user ran | Should |
| FR-21 | Upon a run ending the app shall show the calories burned by a user | Should |
| FR-22 | Upon a run ending the app shall show the average pace of the user | Should |
| FR-23 | The app shall show statistics on all the runs a user has completed | Should |
| FR-24 | The app shall display the total distance walked/ran in a day | Must |
| FR-25 | The app shall show the average distance walked/ran since the user started using the app | Should |
| FR-26 | A user will be required to enter their date of birth upon initial set up | Must |
| FR-27 | The app shall ask the user to select a gender upon initial set up | Must |
| FR-28 | The app shall ask the user to enter their weight upon initial set up | Must |
| FR-29 | The app shall be able to randomly suggest a meal for breakfast | Could |
| FR-30 | The app shall be able to randomly suggest a meal for lunch | Could |
| FR-31 | The app shall be able to randomly suggest a meal for dinner | Could |
| FR-32 | The app shall display the recipe upon selecting a meal | Could |
| FR-33 | The app shall be provide a nutritional breakdown of any meals/snacks it suggests | Could |
| NFR-01 | The app shall have a consistent theme throughout the application | Must |
| NFR-02 | The app shall be simple to understand and use | Must |
| NFR-03 | The app shall work on devices of different screen sizes | Must |
| NFR-04 | The app shall not crash | Must |
| NFR-05 | The app shall be capable of running in the background | Must |
| NFR-06 | The app shall react appropriately to changes orientation | Must |
| NFR-07 | The app shall be easy to navigate | Must |
| NFR-08 | The app shall conform to best practice and design principles for the chosen platforms | Must |

Table 3: MOSCOW Rating for Functional and Non-Functional Requirements

## Risk Analysis

For any project to achieve a successful it is important that all project risks are identified early and managed appropriately to avoid time slippage, increased costs, corruption or even the total loss of the project. A risk can be thought as anything that may be a potential problem that may occur during the software development life cycle.

Table 4 below contains all of the risks identified for this project.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | RISK | IMPACT | RESOLUTION |
| 1 | Developer illness | Reduced productivity. Project falls behind schedule. | Task buffers and trying to work ahead of schedule. |
| 2. | Lack of productivity | Project will fall behind schedule | Creation and following of a project plan will help ensure productivity. |
| 3 | Compromising on Design | Finished software may not follow the same theme and be complex to user. May lead to further coding being needed in later stages. | Designs for all screens will produced before any development commences. 2 weeks has been dedicated to this. |
| 4 | Data Loss | Project may need to be restarted or falls behind schedule due to having redo work. | Multiple copies of all work carried out will be kept. i.e. Cloud, USB, hard drive |
| 5 | Unidentified resources | Time wasted getting the unidentified resources. Fall behind schedule | All required resources will be identified early in the projects planning stage. |
| 6 | Unavailable resources | Resources may be identified which you may need to pay for or are unable to access. Workarounds could compromise project quality. Project falls behind schedule | All resources will gathered in advance and checked that |
| 7 | Unforeseen Circumstances | Both technical and personal issues which may result in work not being completed to schedule. Or having to change project direction. | Buffers are included in the project plan to account for unforeseen circumstances. |
| 8 | Requirement Creep | Constant updating/adding of requirements will result in work not being accounted for in the project plan | Requirements will be finalised and can only be changed should a review carried to determine the feasibility and impact on the projects planned completion date. |

Table 4: Project Risk Analysis

### Risk Mitigation Strategy

Upon all project risks being identified and controls being implemented. All risks where put through a Probabilty-Impact Matrix to ensure that the risks were within acceptable tolerances.

Figures 3.1 and 3.2 show probability-impact matrices for before and after controls were implemented for al risks identified.

#### Pre-Controls



Figure 3.1: Probability-Impact Matrix before controls were implemented

#### Post Controls

Figure 3.1: Probability-Impact Matrix after controls were implemented

# Chapter 4

## UI Design

### Schneiderman’s 8 Golden Rules

1. Strive for consistency.
2. Enable frequent users to use shortcuts.
3. Offer informative feedback.
4. Design dialog to yield closure.
5. Offer simple error handling.
6. Permit easy reversal of actions.
7. Support internal locus of control.
8. Reduce short-term memory load.

(University of Washington, 2018)

# References

Chandana, 2018. *Scrum Project Management – Pros and Cons.* [Online]   
Available at: https://www.simplilearn.com/scrum-project-management-article  
[Accessed 03 November 2018].

Citta, 2017. *Waterfall Model.* [Online]   
Available at: https://www.cittasolutions.com/waterfall-model  
[Accessed 01 November 2018].

Haughey, D., 2014. *REQUIREMENTS GATHERING 101.* [Online]   
Available at: https://www.projectsmart.co.uk/requirements-gathering.php  
[Accessed 14 November 2018].

Intelegain Technologies, 2017. *Scrum Software Development Dallas | Scrum Project Management.* [Online]   
Available at: https://www.intelegain.com/scrum/  
[Accessed 03 November 2018].

Montemagno, J., 2018. *Xamarin.Essentials: Cross-Platform APIs for Mobile Apps.* [Online]   
Available at: https://blog.xamarin.com/xamarin-essentials-cross-platform-apis-mobile-apps/  
[Accessed 03 October 2018].

Mountain Goat Software, 2018. *Scrum Methodology and Project Management.* [Online]   
Available at: http://www.mountaingoatsoftware.com/agile/scrum  
[Accessed 02 November 2018].

PAL, S. K., 2018. *Software Engineering | Spiral Model.* [Online]   
Available at: https://www.geeksforgeeks.org/software-engineering-spiral-model/  
[Accessed 03 November 2018].

Powell-Morse, A., 2016. *Waterfall Model: What Is It and When Should You Use It?.* [Online]   
Available at: https://airbrake.io/blog/sdlc/waterfall-model  
[Accessed 01 November 2018].

TryQA.com, 2013. *What is Waterfall model- Examples, advantages, disadvantages & when to use it?.* [Online]   
Available at: http://tryqa.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/  
[Accessed 30 October 2018].

Tutorialspoint, 2018. *SDLC Overview.* [Online]   
Available at: https://www.tutorialspoint.com/sdlc/sdlc\_overview.htm  
[Accessed 01 November 2018].

University of Washington, 2018. *Shneiderman's "Eight Golden Rules of Interface Design".* [Online]   
Available at: https://faculty.washington.edu/jtenenbg/courses/360/f04/sessions/schneidermanGoldenRules.html  
[Accessed 05 December 2019].

Wisdomjobs.com, 2018. *SDLC OVERVIEW SOFTWARE DEVELOPMENT LIFECYCLE (SDLC).* [Online]   
Available at: https://www.wisdomjobs.com/e-university/software-development-lifecycle-sdlc-tutorial-1513/sdlc-overview-25976.html  
[Accessed 01 November 2018].